Интернет-технологии и распределённая обработка данных

ЛЕКЦИЯ 1

Вопросы занятия

Модель OSI.

Понятие сетевого протокола.

Стек протоколов TCP/IP.

Коммуникация в TCP/IP

Протокол IP

Протоколы UDP и TCP

URI, URL, URN

Протокол HTTP

Компьютерная сеть – определение

Коммуникационная сеть – система каналов связи и коммутационного оборудования для передача информации с минимальным количеством ошибок и искажений (примеры: телефонная сеть, кабельное, сотовые сети).

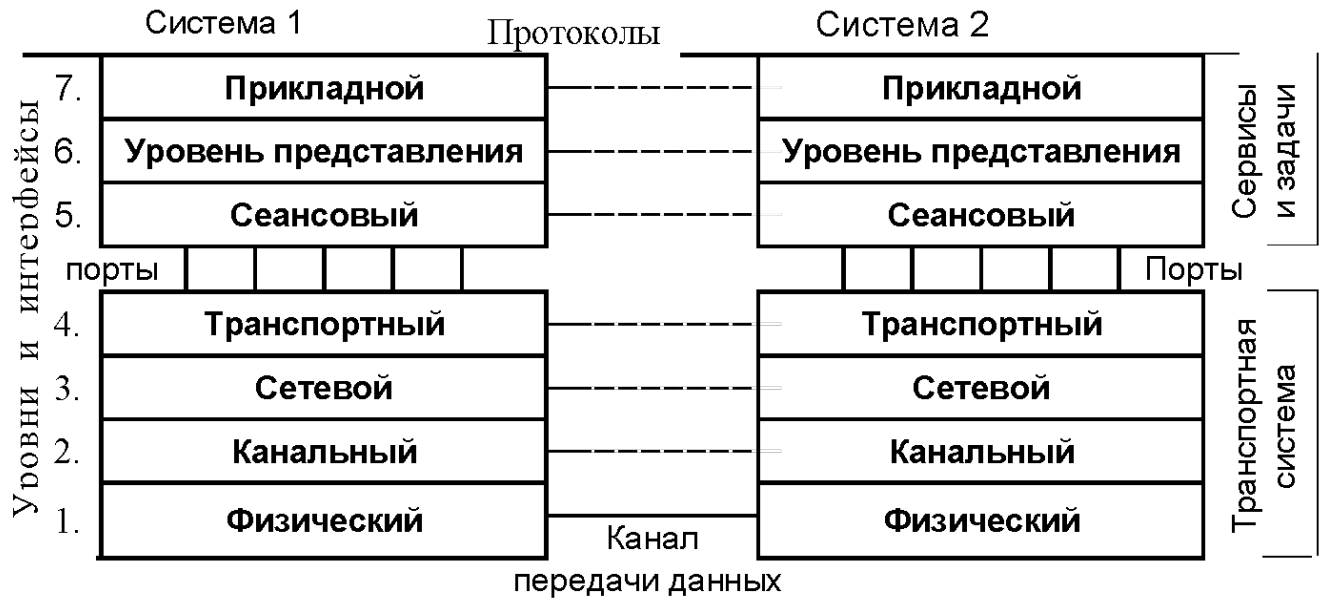
Компьютерная сеть – вид коммуникационной сети для обмена данными между вычислительными устройствами.

Сети как открытые системы

Сети поддерживают концепцию *открытой системы* (построены на основе открытых спецификаций, открыты для расширения и взаимодействия).

Модель *взаимодействия открытых систем* (Open System Interconnection, OSI) принята Международной организацией по стандартизации (ISO) в 1983 г.

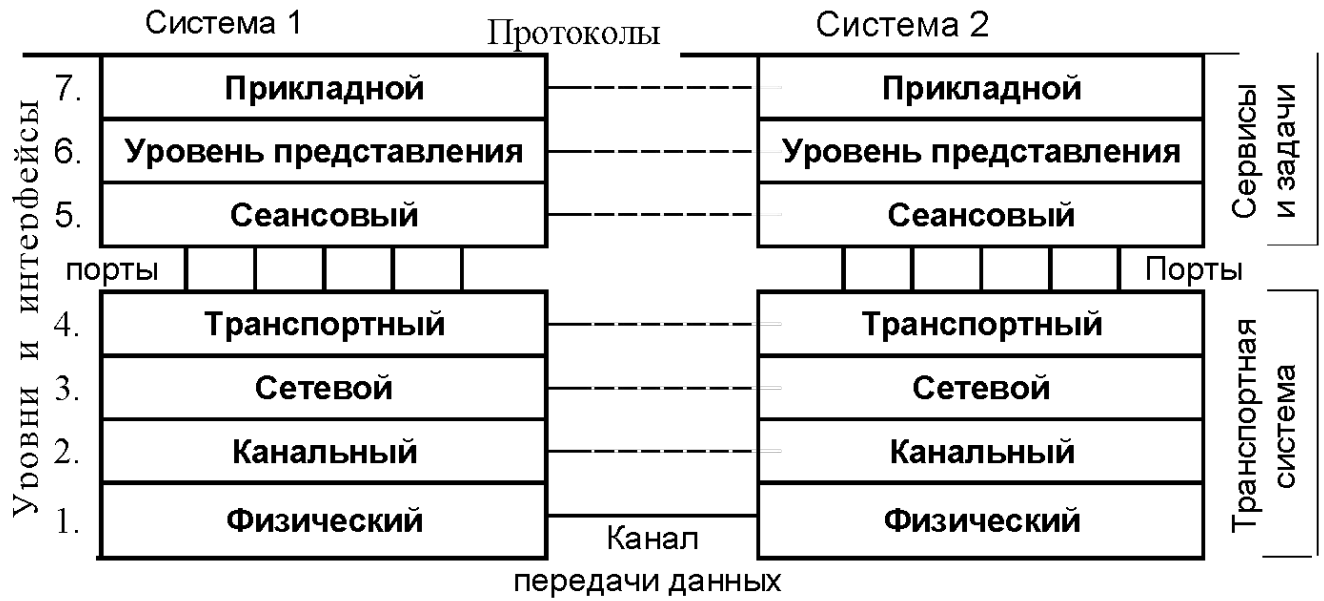
Модель OSI



Определяет набор уровней взаимодействия двух систем и правила организации уровней.

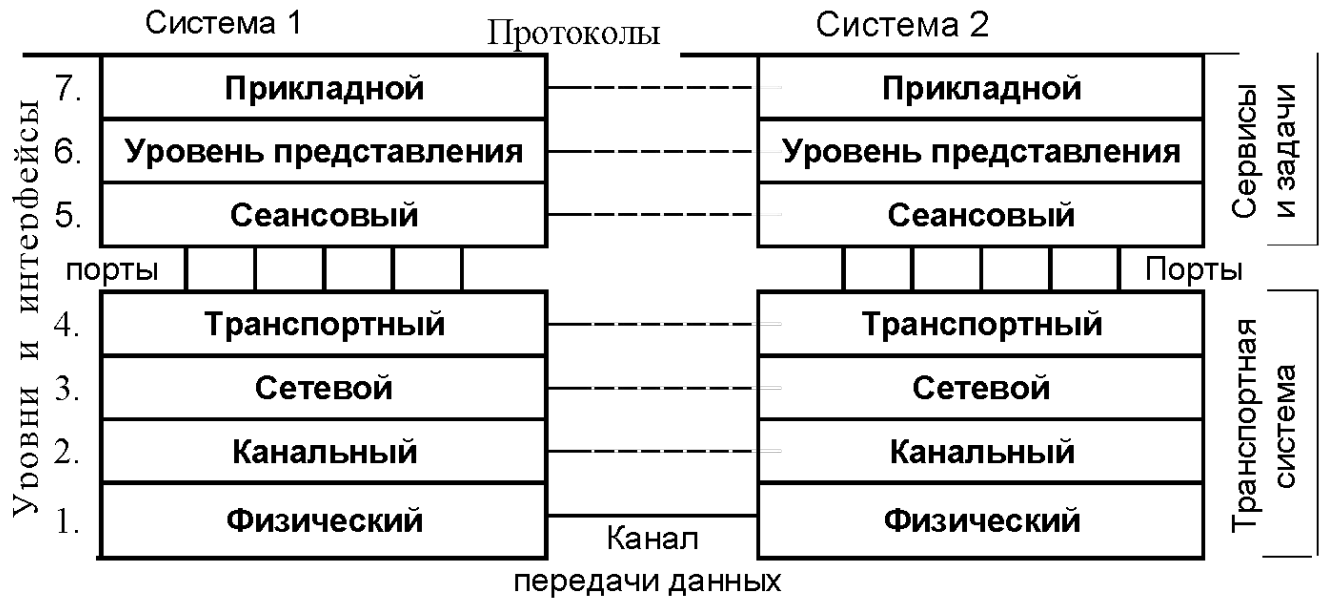
Организует сетевые протоколы.

Модель OSI



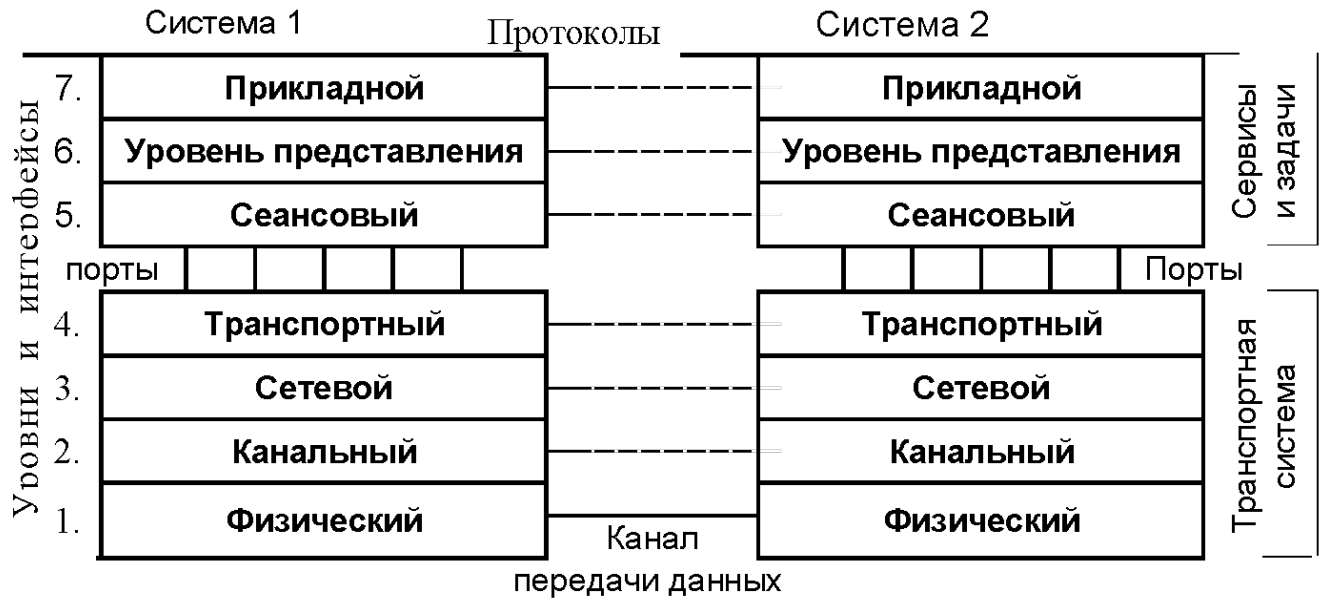
Протоколы работают друг с другом в стеке – протокол, располагающийся на уровне выше, работает «поверх» нижнего, используя механизмы инкапсуляции.

Модель OSI – уровень 1



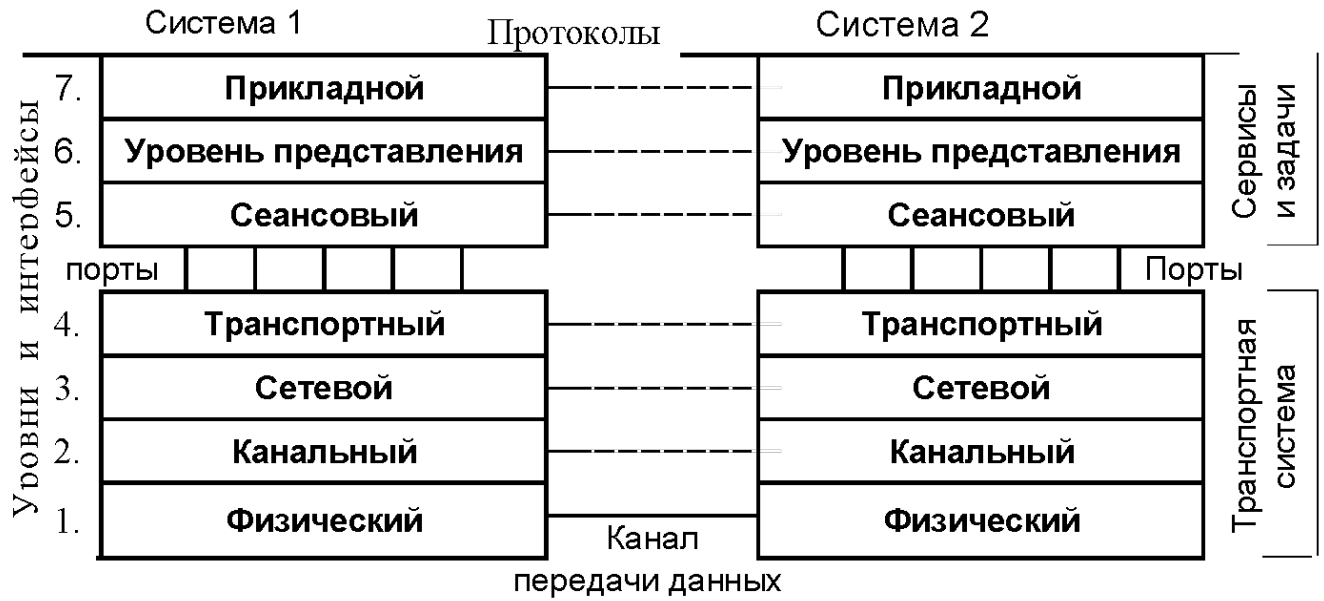
Физический уровень – аппаратура подключения к сети. Этот уровень обеспечивает взаимодействие со средой передачи данных на уровне сигналов **(ADSL, USB)**

Модель OSI – уровень 2



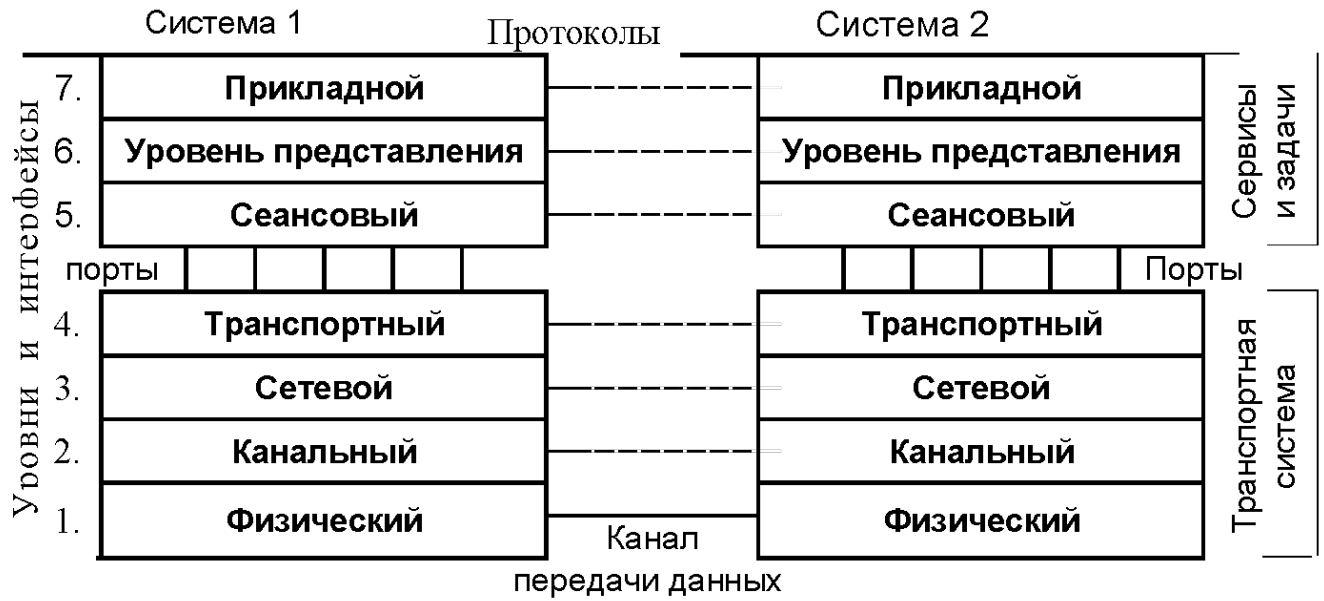
Канальный уровень осуществляет логическое управление физическими устройствами и повышение достоверности передачи – контроль и исправление ошибок **(PPP, IEEE 802.3)**

Модель OSI – уровень 3



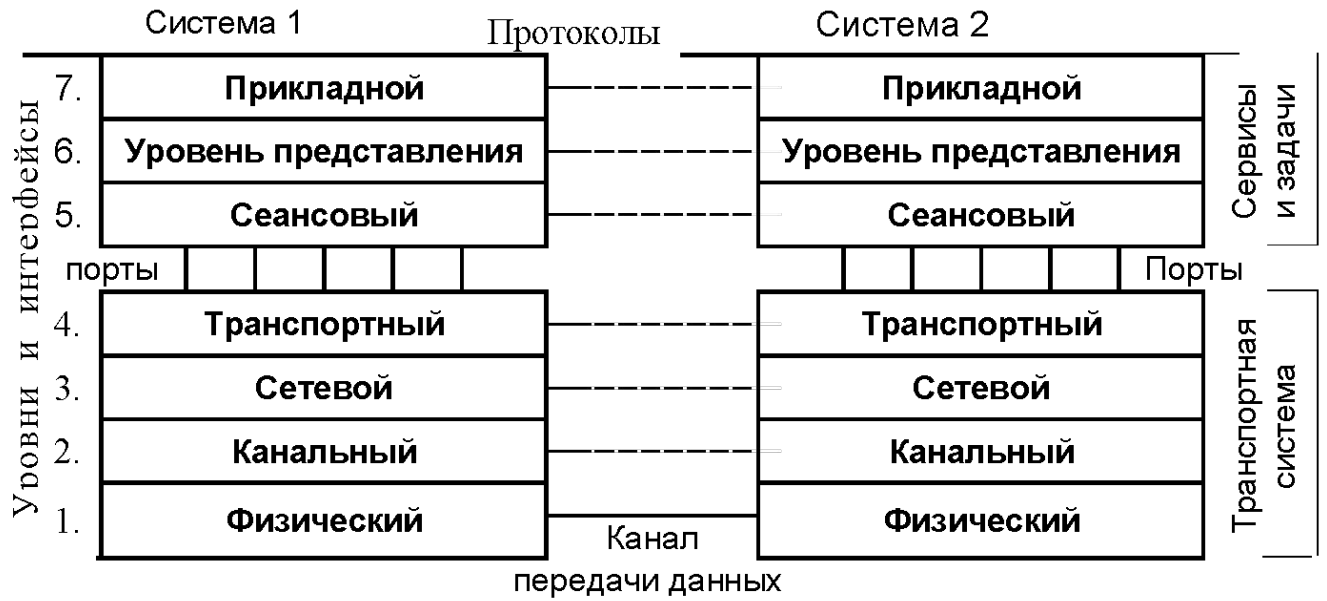
Сетевой уровень организует поиск адресов в сети и перенаправление передаваемых адресованных данных (*маршрутизация*) **(IPv4 или просто IP, IPv6)**

Модель OSI – уровень 4



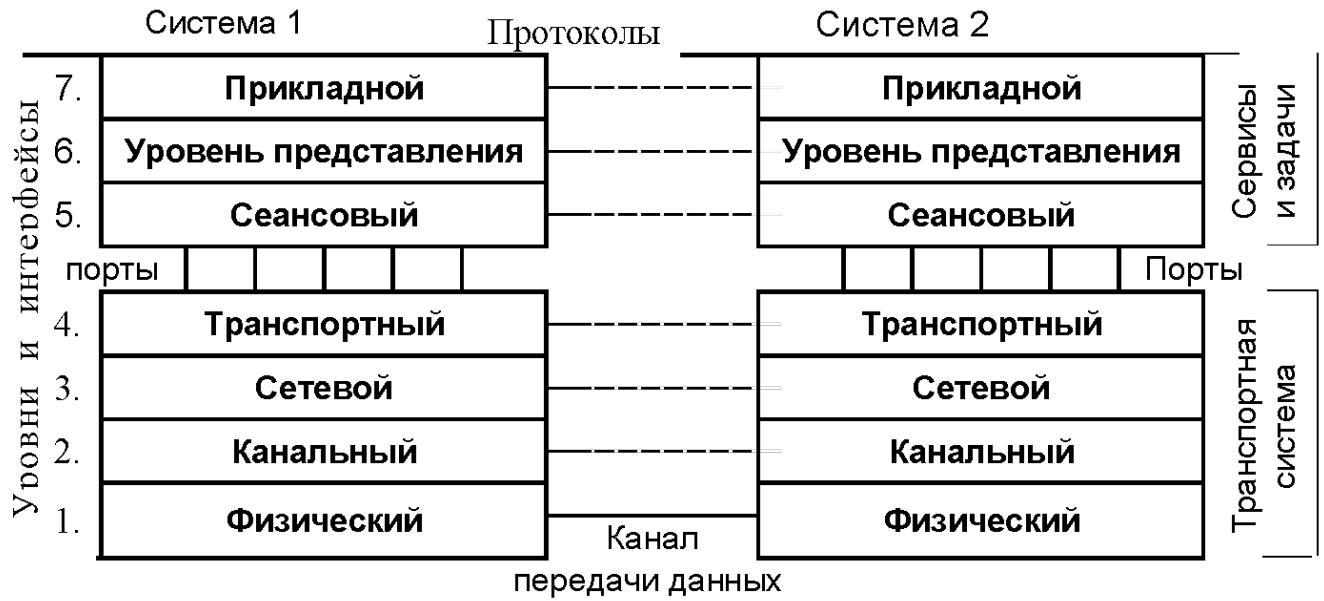
Транспортный уровень выполняет передачу от одной точки (адреса) к другой с необходимым контролем и (возможно) дополнительным сервисом **(TCP, UDP)**

Модель OSI – уровень 5



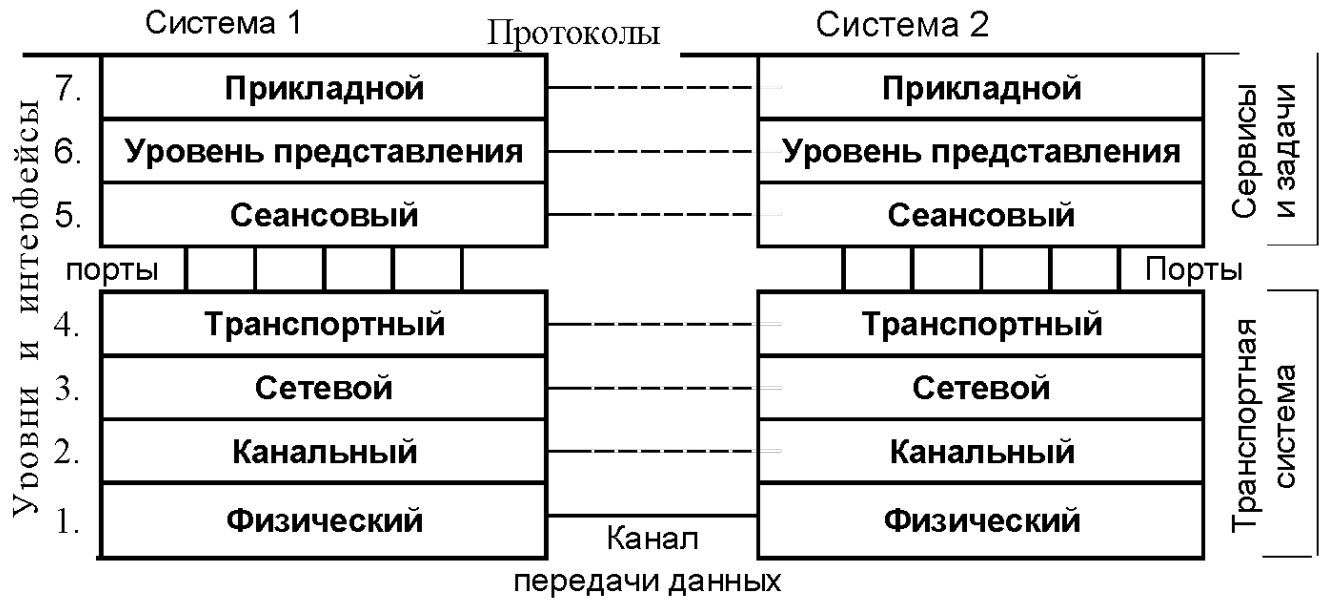
Сеансовый уровень обеспечивает установление соединений (*сеансов*) между взаимодействующими системами (процессами) и управление сеансами **(RPC)**

Модель OSI – уровень 6



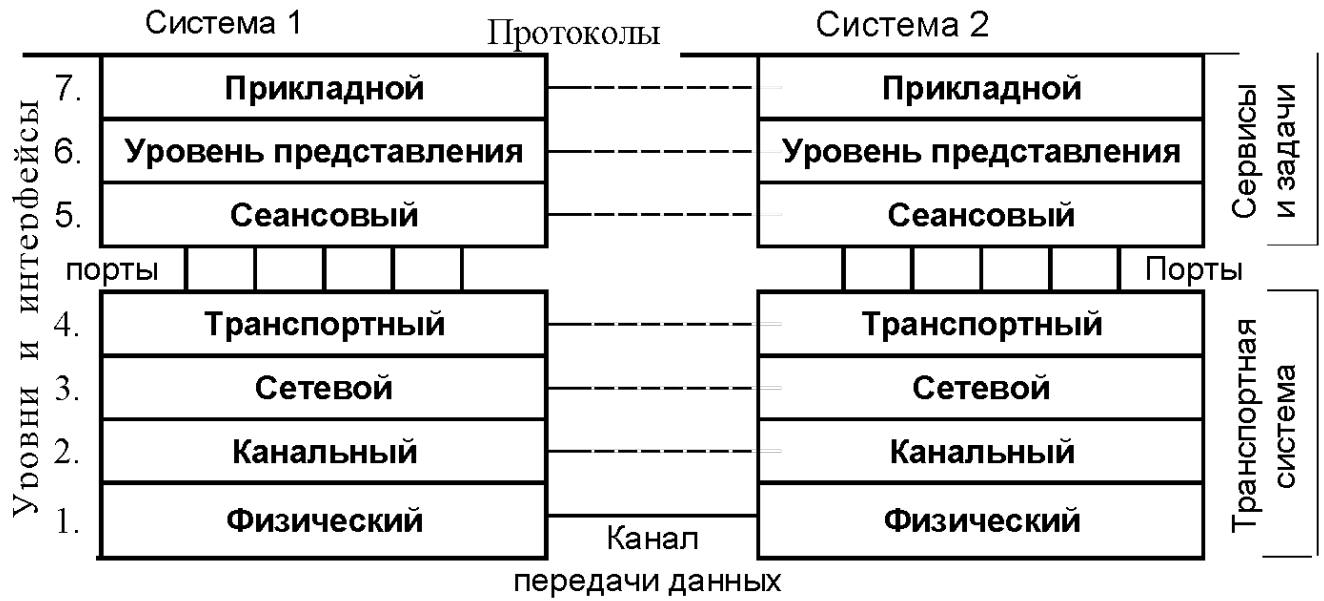
Уровень представления служит для преобразования форматов данных (например, вид кодировки) в соответствии с правилами ПО следующего 7-го уровня **(ASCII, JPEG)**

Модель OSI – уровень 7



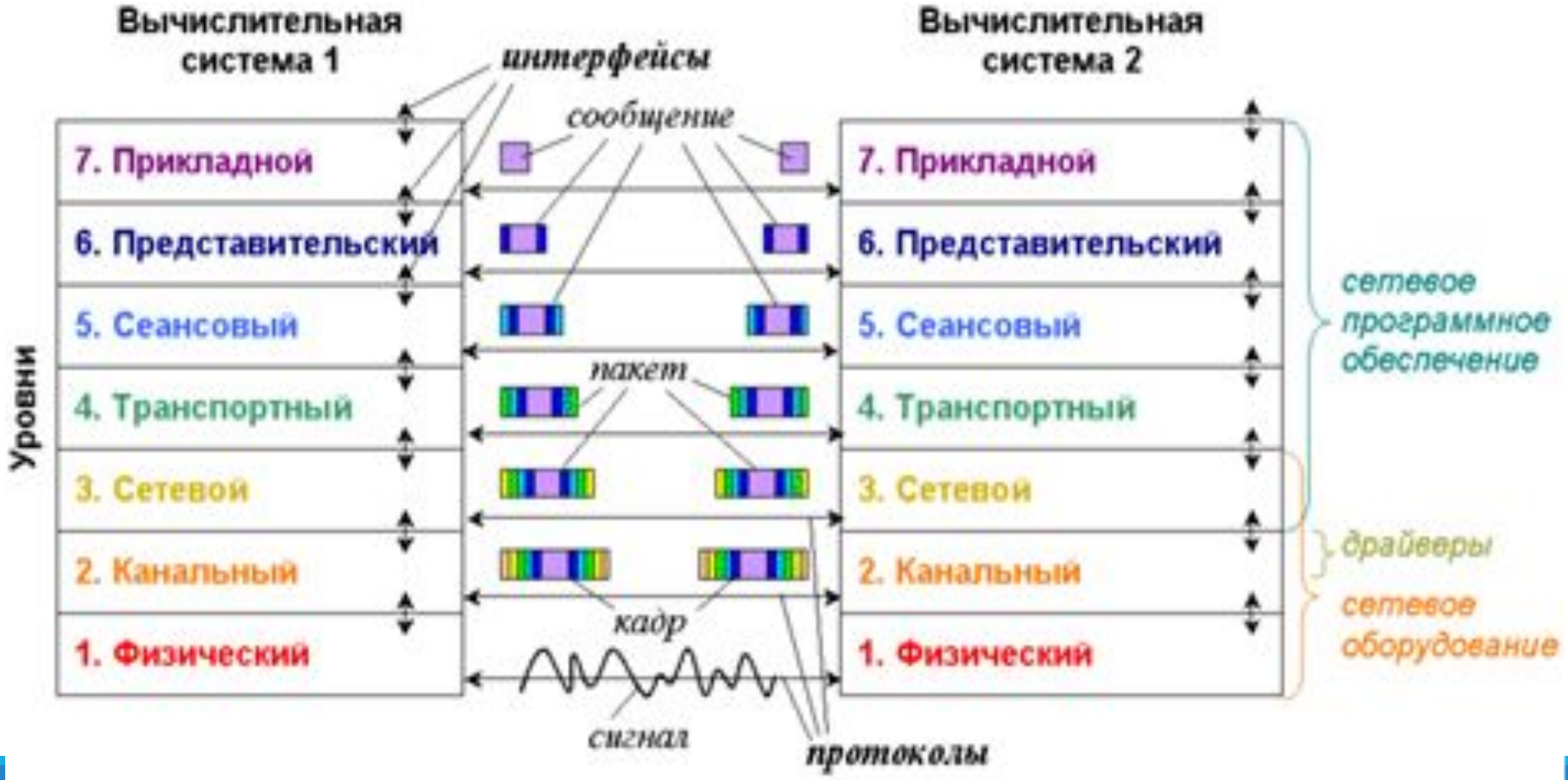
Прикладной уровень – конечные приложения, как чисто прикладные (программы пользователя), так и служебные (т. н. *службы* или *сервисы*) **(HTTP, FTP, SMTP, POP3)**

Модель OSI



Уровни 1-4 это *транспортная система*. Для вышестоящих уровней транспортный уровень создает так называемые *порты* – точки доступа к функциям транспортной системы.

Модель OSI


Вычислительная система 1
интерфейсы
Вычислительная система 2
сообщение
7. Прикладной
6. Представительский
5. Сеансовый
4. Транспортный
3. Сетевой
2. Канальный
1. Физический
Уровни
пакет
кадр
сигнал
протоколы
сетевое программное обеспечение
драйверы
сетевое оборудование

Понятие сетевого протокола

Сетевой протокол – набор правил и действий для соединения и обмена данными между двумя и более включёнными в сеть устройствами.

TCP, HTTP, POP3, FTP, IP – примеры сетевых протоколов.

http://www.wikiwand.com/en/Lists_of_network_protocols

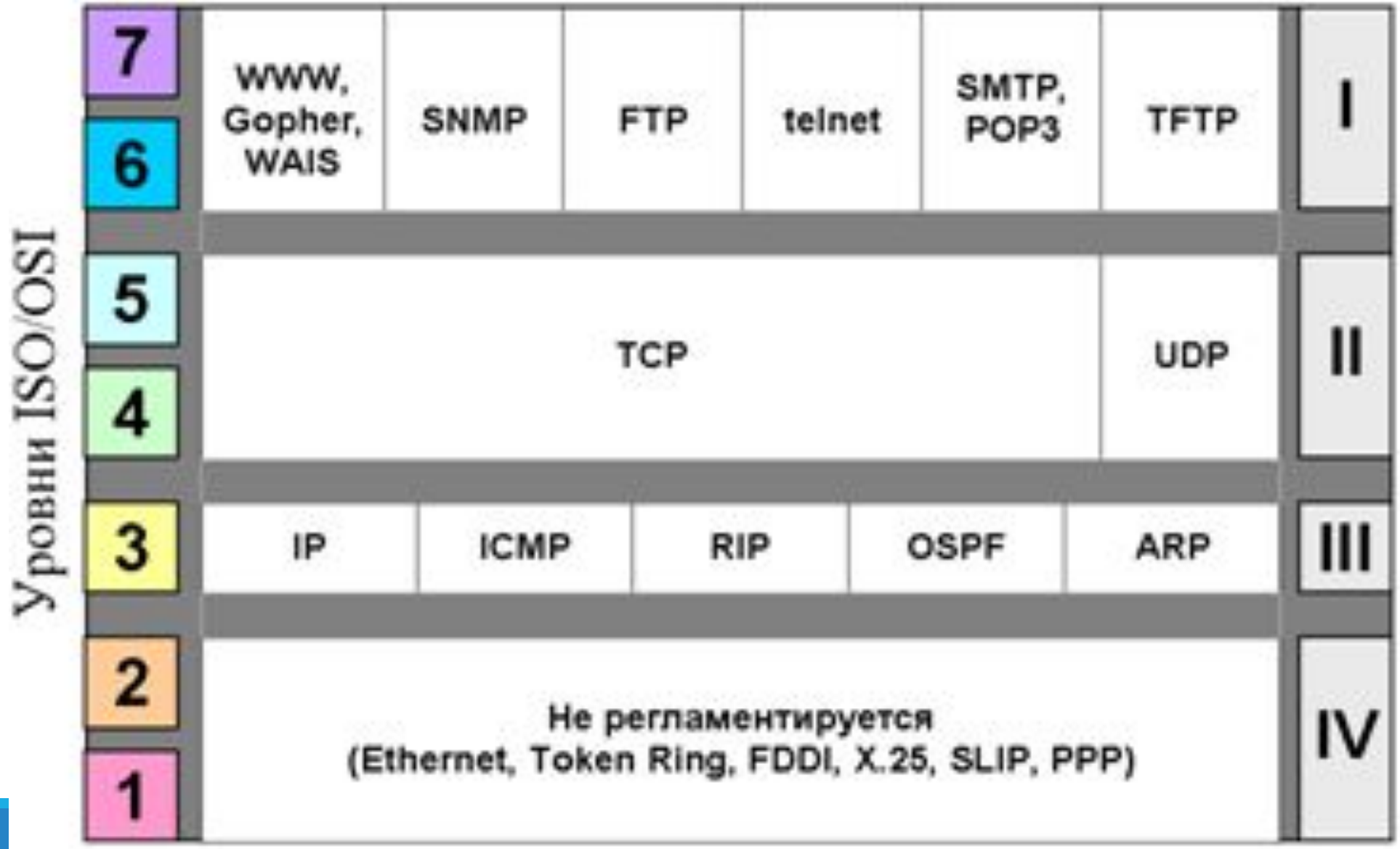
TCP/IP

Стек протоколов TCP/IP – набор сетевых протоколов передачи данных, используемых в сетях, включая сеть Интернет.

Реализует модель OSI (**не точно**, ибо появился раньше её).

Реализован в большинстве операционных систем.

Уровни стека TCP/IP



Коммуникация в TCP/IP

Сетевое взаимодействие подразумевает минимум двух участников: *отправителя* информации по сети и *получателя* этой информации (например, браузер и веб-сервер; почтовый клиент и почтовый сервер).

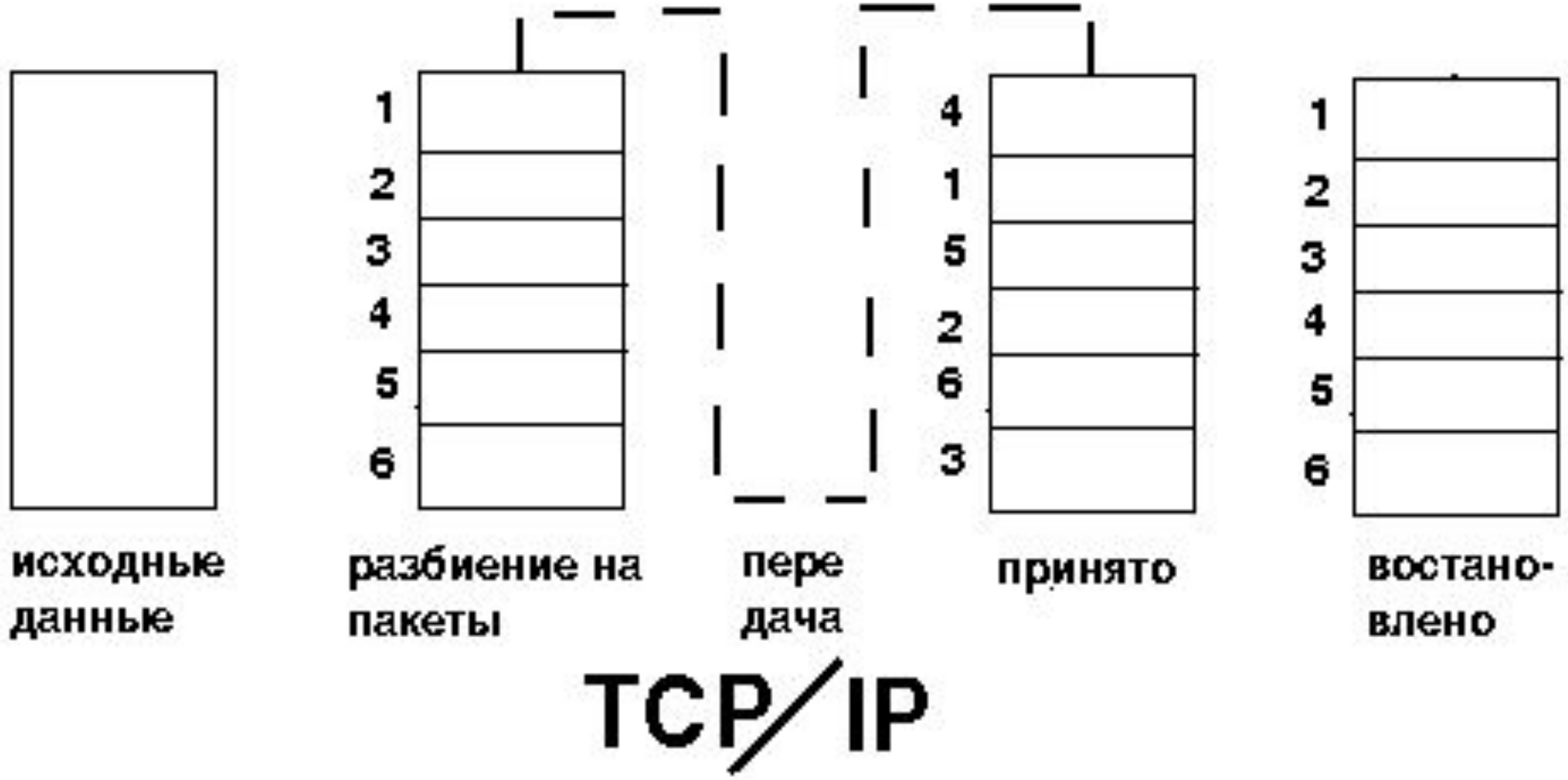
О чём надо бы задуматься: как одна часть сетевой программы найдет свою вторую «половинку»?

Коммуникация в TCP/IP

Хорошая новость: протоколы транспортной системы обычно реализованы на уровне ОС. Их функции доступны через соответствующее API.

(Для того, кто пишет драйверы сетевого оборудования или саму операционную систему, не всё так хорошо)

Передача информации по протоколу TCP/IP



Идентификация участников

Идентификатор участника сетевого взаимодействия состоит из двух частей:

1. Идентификатора (*адреса*) узла.
2. Идентификатора программы на этом узле.

Адрес узла

Протокол сетевого уровня (например, IP) использует *логические адреса* (сетевые адреса).

На уровне оборудования используются *физические адреса* (MAC-адреса). Преобразование физических адресов в логические и обратно называют *разрешением адресов*. Оно прозрачно для прикладных программ.

Сетевые имена

Для удобства человека и повышения гибкости системы узлы могут получать символические обозначения – *сетевые имена*.

Преобразование сетевых имен в адреса выполняется обычно специальными службами (DNS, NAT).

Идентификатор программы

В этом качестве принято использовать **номер порта**, который предоставлен программе.

Транспортные протоколы создают порты независимо друг от друга, поэтому необходим также и идентификатор транспортного протокола.

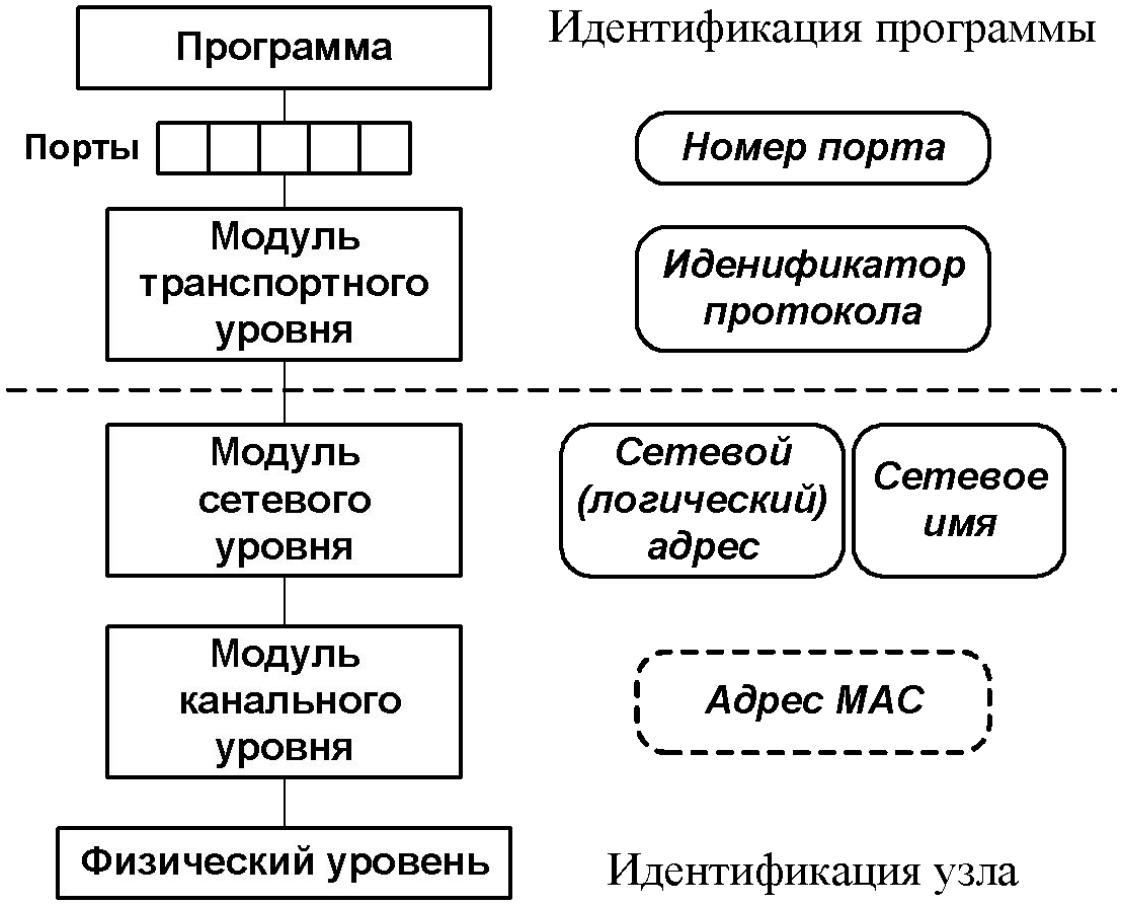
Идентификация участников – выводы

Полный идентификатор включает три звена:

<сетевой адрес>:<транспортный протокол>:<порт>

Например: 192.168.0.99:TCP:8080

Идентификация участников – схема



Протокол IP (Internet Protocol)

Принадлежит стеку TCP/IP (*сетевой уровень*).

Решает две задачи:

- 1) *адресация* узлов сети;
- 2) *передача данных* между адресованными узлами.

Система адресации позволила *объединять* сети.

Сеть Internet построена на протоколе IP.

Протокол появился в 1981 году.

Передача данных по протоколу IP

Информация передаётся порциями (*датаграмма,*

Длина макс. 65535 байт, но обычно короче.

Адресация в IP-сетях

IP-сеть – сеть, использующая протокол IP для работы.

Каждому узлу IP-сети назначается *сетевой адрес* (*IP-адрес, «айпишник»*).

В протоколе IP версии 4 (IPv4) адреса 32-разрядные.

Принятая форма записи адреса: байты как десятичные числа, разделённые точками, старший байт в начале:

192.168.0.99

IPv4

Класс A	Сеть	Узел		
Октет	1	2	3	4
Маска подсети по умолчанию	255	0	0	0
Диапазон IP-адресов	1.0.0.0 – 126.255.255.255			
Число узлов в сети	2^24-2=16 777 214			

Класс B	Сеть		Узел	
Октет	1	2	3	4
Маска подсети по умолчанию	255	255	0	0
Диапазон IP-адресов	128.0.0.0 – 191.255.255.255			
Число узлов в сети	2^16-2=65 534			

Класс C	Сеть			Узел
Октет	1	2	3	4
Маска подсети по умолчанию	255	255	255	0
Диапазон IP-адресов	192.0.0.0 – 223.255.255.255			
Число узлов в сети	2^8-2=254			

Специальные адреса

0.0.0.0 – узел-источник пакета данных или узел с неопределенным адресом (например, для задач маршрутизации).

Специальные адреса

255.255.255.255 – *«все узлы»*, т. е. любой узел сети.

Это позволяет организовать широковещательную (broadcast) рассылку (правда только в пределах локальной сети – не можем засорять Internet!).

Специальные адреса

127.x.x.x (семейство адресов) – *локальный адрес*; это тот узел, на котором выполняется программа.

Обычно используется 127.0.0.1, сетевое имя: **localhost**

Специальные адреса

10.x.x.x, 172.16.x.x - 172.31.x.x, 192.168.x.x – *«внутренние» адреса*: пакеты данных, направленные на эти адреса, действуют только в локальной сети, никогда не отсылаются из неё в глобальную и соответственно никогда из нее не приходят.

Уникальность таких адресов необходимо поддерживать только в пределах локальной сети, но при этом, очевидно, **узел не может иметь прямого выхода в глобальную сеть**.

Протоколы UDP и TCP

UDP – User Datagram Protocol (протокол пользовательских датаграмм)

TCP – Transmission Control Protocol (протокол управления передачей)

Оба протокола – из стека TCP/IP (*транспортный уровень*). Решают задачи доставки сообщений.

Протокол UDP

Простой транспортный протокол.

По сути, берёт порцию данных, снабжает её коротким служебным заголовком и передаёт всё это протоколу IP для дальнейшей передачи.

Факт получения или пропажи данных никак не отслеживается.

Протокол UDP

+ Быстрее передаёт данные, так как не выполняет процедуры установления соединения между узлами

+ Позволяет вести широковещательную рассылку (в рамках локальной сети)

+ Имеет короткий служебный заголовок (8 байт)

– Не гарантирует доставки данных

– Сообщения имеют ограниченную длину (65 528 байт). Значит, много данных нужно передавать несколькими сообщениями

Протокол TCP

Основная задача – **надёжная** доставка данных.

Ориентирован на соединение: два приложения перед обменом данными должны выполнить определенные вспомогательные действия, называемые *установлением соединения*.

Следствие: в обмене данными всегда участвуют **две** оконечные точки. Организовать широковещательную рассылку при помощи протокола TCP нельзя!

Протокол TCP

Для обеспечения надёжности:

1. Разбивает передаваемые данные на сегменты оптимальной длины, которые приёмник собирается в правильном порядке.

2. При пересылке сегмента использует таймер для ожидания подтверждения от принимающей стороны (квитанция). Если по истечении определенного времени квитанция не приходит, выполняется повторная передача сегмента.

3. После получения сегмента данных принимающая сторона проверяет контрольную сумму сегмента.

Протокол TCP

+ Обеспечивает надёжную доставку информации

– Работает медленнее UDP

– Нет широковещательной рассылки

– Медленнее «стартует» (установление соединения)

Использование UDP и TCP

Приложение	Прикладной протокол	Транспорт
Электронная почта	SMTP	TCP
Доступ с удалённого терминала	Telnet	TCP
Web	HTTP	TCP
Передача файлов	FTP	TCP
Трансляция имён	DNS	UDP
Оптимальная маршрутизация	RIP	UDP
Интернет-телефония	SIP	UDP

Символьные имена узлов сети

IP-адрес человеку трудно воспринимать и запоминать.

Символьная система: отдельному узлу сети назначается определенное символьное имя.

Преимущества:

+ символьные читаемые имена для человека более удобны, чем числовые значения;

+ при изменении IP-адресов имена можно не менять, или, наоборот, произвольно изменить имена при неизменных адресах путем перенастройки системы разрешения имен.

Символьные имена узлов сети

Первые системы символьных имён были «плоские»: администратор локальной сети поддерживал таблицу, в которой сопоставлялось имя и IP-адрес.

– Как гарантировать уникальность в глобальной сети?

– Где хранить таблицы соответствия?

– Кто будет администрировать таблицы в Internet?

Система доменных имён

Современная схема – *система доменных имен* (Domain Name System, DNS).

Имена в DNS образуют иерархически организованное пространство, которое можно представить в виде дерева.

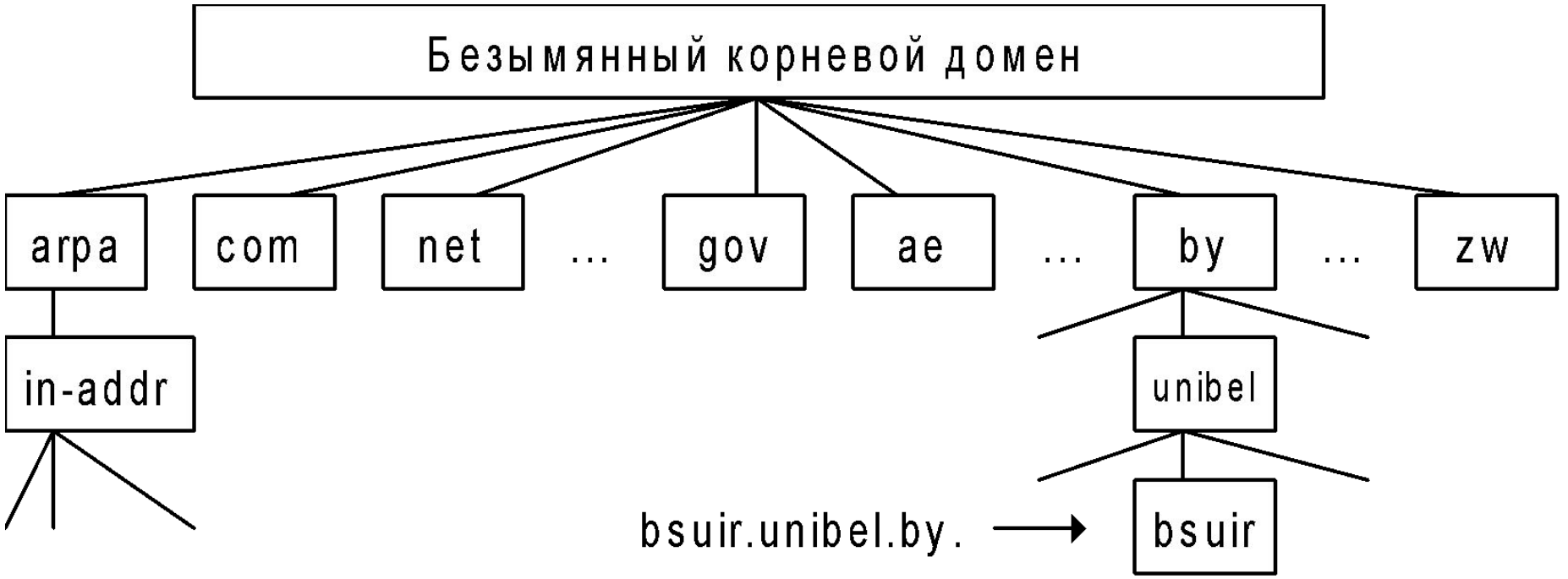
Система доменных имён

Каждому узлу дерева сопоставлена метка длиной до 63 символов.

Прописные и заглавные буквы в метках не различаются.

Доменное имя каждого узла представляет собой список меток, разделенных точками. Список начинается слева, с метки текущего узла, в нем перечислены все узлы расположенные вверх по иерархии, вплоть до корня.

Система доменных имён



Трансляция имён

DNS обеспечивает ещё и задачу трансляции символьного имени в IP-адрес (и наоборот).

Глобально для этого используются *сервера имён*.

На компьютере этим занимается специальная *служба DNS*.

Алгоритм трансляции имён

1. Поиск соответствия в локальной таблице, хранящейся на компьютере (файл **%SystemRoot%\system32\drivers\etc\hosts**).

2. Если разрешить имя локально не удалось, выполняются запросы к одному или нескольким серверам DNS, IP-адреса которых указаны в настройках стека протоков TCP/IP на данной машине. Данные на запросы возвращаются в виде ответов.

3. Сервер может дать точные ответы об именах из своей зоны ответственности либо переадресовать запрос к соседнему серверу.

Uniform Resource Identifier (URI)

Универсальный идентификатор ресурса (Uniform Resource Identifier) – строка символов для идентификации (обозначения) произвольного ресурса в глобальной сети.

http://example.org/wiki/Main_Page

URL и URN

URI может быть представлен в двух формах:

1. Uniform Resource Locator, **URL** – это URI, который, помимо идентификации ресурса, предоставляет ещё и информацию о местонахождении ресурса (URL называют *веб-адрес*):

http://example.org/wiki/Main_Page

2. Uniform Resource Name, **URN** – это URI, который только идентифицирует ресурс в определённом контексте, но не указывает его местонахождения:

urn:isbn:0-486-27557-4

Структура абсолютного URI

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

1. *Схема* и : (двоеточие).
2. *Специфичная для схемы часть.*

Для URL это будет

1) // (два слэша)
2) имя хоста (символьное имя компьютера)
3) : порт (опционально)
4) путь на хосте к ресурсу
5) ? параметры запроса к ресурсу (опционально, разделяются &)

http://example.com #запрос стартовой страницы по умолчанию

http://www.example.com/site/map.html #запрос страницы в указанном каталоге

http://example.com:81/script.php #подключение на нестандартный порт

http://example.org/script.php?key=value #передача параметров скрипту

ftp://user:pass@ftp.example.org #авторизация на ftp-сервере

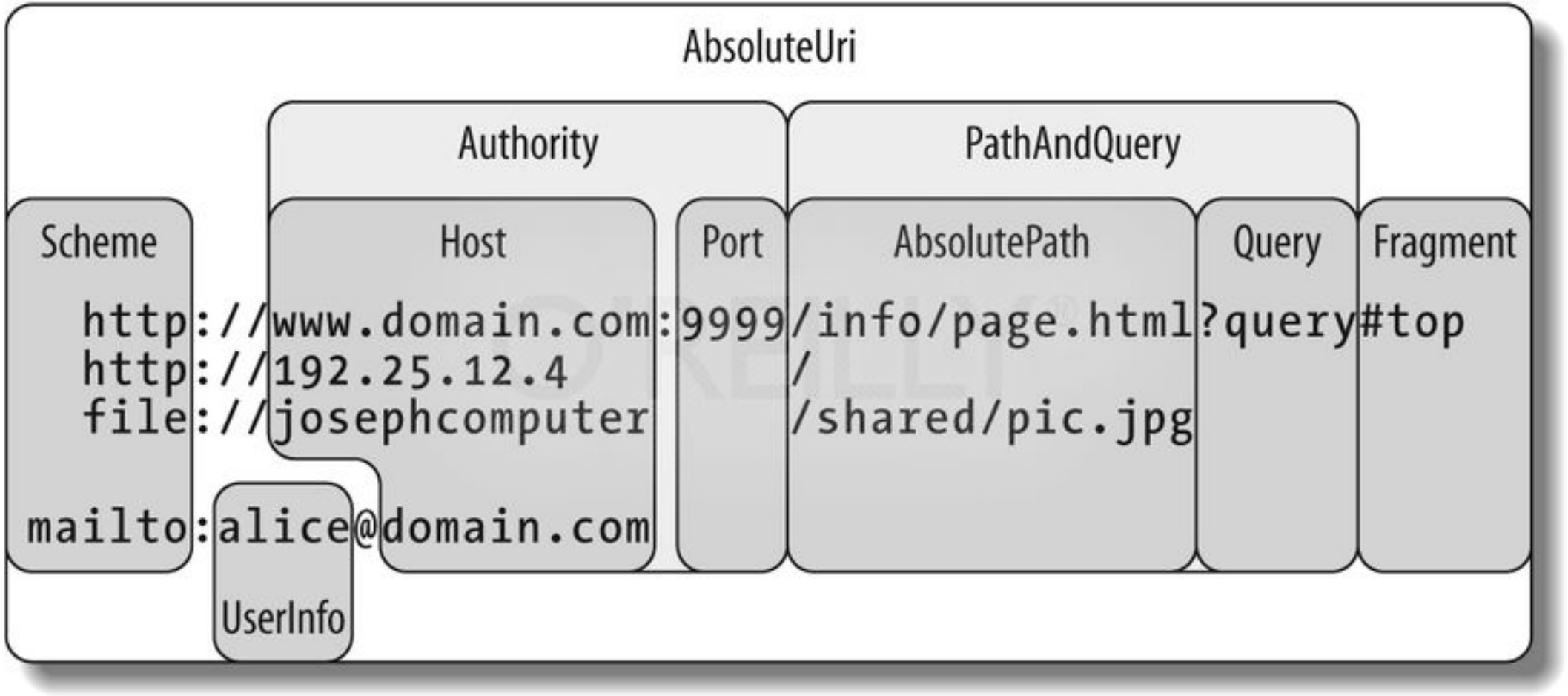
http://192.168.0.1/example/www #подключение по ip-адресу

file:///srv/www/htdocs/index.html #открытие локального файла

gopher://example.com/1 #подключение к серверу gopher

mailto://user@example.org #ссылка на адрес эл.почты

Структура абсолютного URI



Протокол HTTP

Протокол передачи гипертекста (Hypertext Transfer Protocol) – протокол прикладного уровня для передачи данных (изначально – в виде HTML-документов, сейчас – для передачи произвольных данных).

Протокол HTTP – факты

1. Создан в 1992 году
2. Версии: HTTP/0.9 (устарела), HTTP/1.0, **HTTP/1.1** , HTTP/2 , HTTP/3
3. Описан в RFC 1945, RFC 2616
4. Работает поверх протокола TCP, стандартный порт: 80
5. Ориентирован на сети «клиент-сервер»
6. Работа по протоколу – это обмен текстовыми сообщениями между клиентом и сервером

HTTPS (аббр. от англ. *HyperText Transfer Protocol Secure*) — расширение протокола HTTP для поддержки шифрования в целях повышения безопасности. Данные в протоколе HTTPS передаются поверх криптографических протоколов TLS или устаревшего в 2015 году - SSL .

В отличие от HTTP с TCP-портом 80, для HTTPS по умолчанию используется TCP-порт 443.

По данным W3Techs 42,1 % из 10 млн самых популярных интернет-сайтов поддерживают протокол HTTP/2:

1) бинарный (формирование пакетов легче и проще)

2)изменены способы разбиения данных на фрагменты и транспортирования их между сервером и клиентом.

3) сервер имеет право послать то содержимое, которое ещё не было запрошено клиентом. Это позволит серверу сразу выслать дополнительные файлы, которые потребуются браузеру для отображения страниц, без необходимости анализа браузером основной страницы и запрашивания необходимых дополнений.

4) мультиплексирование запросов и ответов для преодоления проблемы «head-of-line blocking» протоколов HTTP 1;

5) сжатие передаваемых заголовков и введения явной приоритизации запросов.

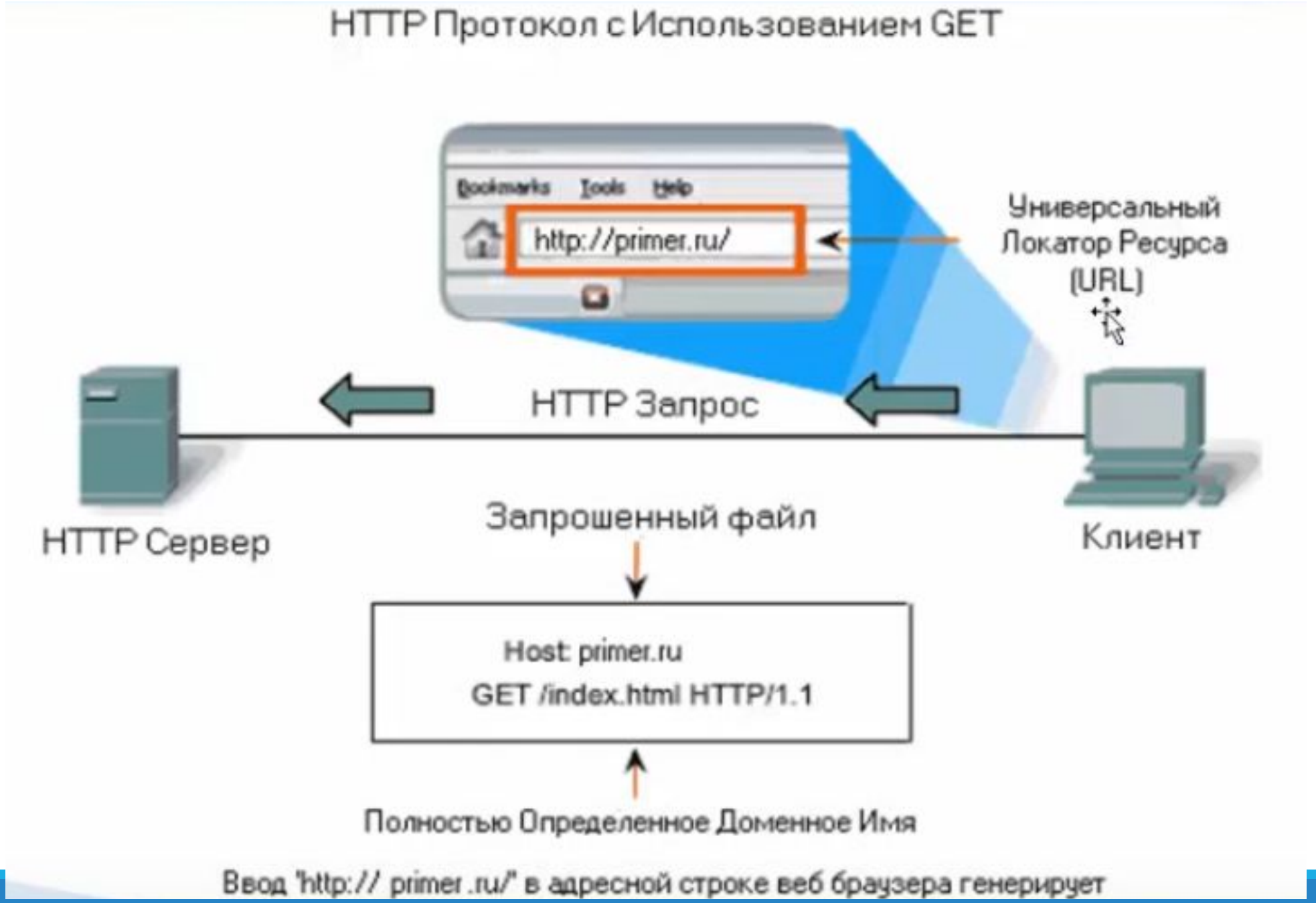
HTTP-протокол

Структура:

I. Стартовая строка (starting line) – обязательно!

II. Заголовки (headers) – опционально

III. Тело сообщения (message body) – опционально, отделяется пустой строкой


HTTP Протокол с Использованием GET
Bookmarks Tools Help
http://primer.ru/
Универсальный
Локатор Ресурса
(URL)
HTTP Запрос
HTTP Сервер
Клиент
Запрошенный файл
Host: primer.ru
GET /index.html HTTP/1.1
Полностью Определенное Доменное Имя
Ввод "http:// primer .ru/" в адресной строке веб браузера генерирует

Стартовая строка запроса

Метод *URI* HTTP/*Версия*

Метод – название запроса (определяет действие), одно слово из стандартного списка, заглавными буквами

URI определяет путь к запрашиваемому ресурсу.

Версия – пара разделённых точкой цифр. Например: 1.0

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

Стартовая строка ответа `HTTP/1.1 200 Ok`

Методы (глаголы) HTTP

Метод	Что делает
GET	Запрашивает ресурс, расположенный по указанному URI.
POST	Используется для передачи на сервер данных, которые должны быть обработаны ресурсом, указанным в URI. Данные передаются в теле запроса.
PUT	Позволяет сохранить (или перезаписать) передаваемый на сервер ресурс с указанным URI.
DELETE	Используется для удаления ресурса, указанного в URI.
HEAD	Аналогичен GET, но клиенту возвращается только заголовок сообщения ответа. Этот метод можно использовать для проверки доступа к ресурсам.
OPTIONS	Запрашивает характеристики соединения между клиентом и сервером, характеристики сервера, требования для запроса данного ресурса и т. п. Если вместо идентификатора URI стоит символ «*», то запрашивается вся доступная информация о сервере.

Примеры стартовых строк

GET /index.html HTTP/1.1

GET /index.html?x=1&y=%D0%9A%D0%BE HTTP/1.0

POST /login.php HTTP/1.1

Стартовая строка ответа

HTTP/*Версия* *КодСостояния* *Пояснение*

Версия – пара разделённых точкой цифр. Например: 1.1

Код состояния – три цифры. По коду состояния определяется дальнейшее содержимое сообщения и поведение клиента.

Пояснение – текстовое короткое пояснение кода для пользователя. Не является обязательным.

```
HTTP/1.1 200 Ok
```

Код состояния

Коды состояния (Status Codes)

Класс	Описание класса
1xx	Информационный: запрос принят, процесс обработки продолжается
2xx	Успешное завершение: запрос был успешно принят и обработан
3xx	Переназначение: следующее действие должно быть обработано, чтобы завершить запрос
4xx	Ошибка клиента: запрос содержит неверный синтаксис или не может быть выполнен
5xx	Ошибка сервера: сервер не может выполнить требуемый запрос

Коды состояния – примеры

100 Continue (Продолжать)

101 Switching Protocols (Переключение протоколов)

102 Processing (Идёт обработка)

200 OK (Успешно)

201 Created (Создано)

202 Accepted (Принято)

204 No Content (Нет содержимого)

206 Partial Content (Частичное содержимое)

Коды состояния – примеры

300 Multiple Choices (Множественный выбор)

301 Moved Permanently (Перемещено навсегда)

304 Not Modified (Не изменялось)

Коды состояния – примеры

401 Unauthorized (Неавторизован)

402 Payment Required (Требуется оплата)

403 Forbidden (Запрещено)

404 Not Found (Не найдено)

405 Method Not Allowed (Метод не поддерживается)

406 Not Acceptable (Не приемлемо)

407 Proxy Authentication Required (Требуется аутентификация прокси)

Коды состояния – примеры

500 Internal Server Error (Внутренняя ошибка сервера)

502 Bad Gateway (Плохой шлюз)

503 Service Unavailable (Сервис недоступен)

504 Gateway Timeout (Шлюз не отвечает)

HTTP-заголовки

Используются для настройки параметров передачи, описания тела сообщения (если оно есть) и прочих сведений

Записываются в виде *ИмяЗаголовка: Значение*

Есть заголовки, специфичные только для запросов или только для ответов. А есть универсальные заголовки.

HTTP-заголовки

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

HTTP-заголовки – примеры

http://en.wikipedia.org/wiki/List_of_HTTP_header_fields

```
Host: en.wikipedia.org
Accept-Language: en-US
User-Agent: Mozilla/5.0
Content-Length: 348
Content-Type: text/html; charset=utf-8
```

HTTP-запрос – пример 1

```
GET /wiki/List_of_HTTP_header_fields HTTP/1.1
Host: en.wikipedia.org
Accept-Language: en-US
User-Agent: Mozilla/5.0
```

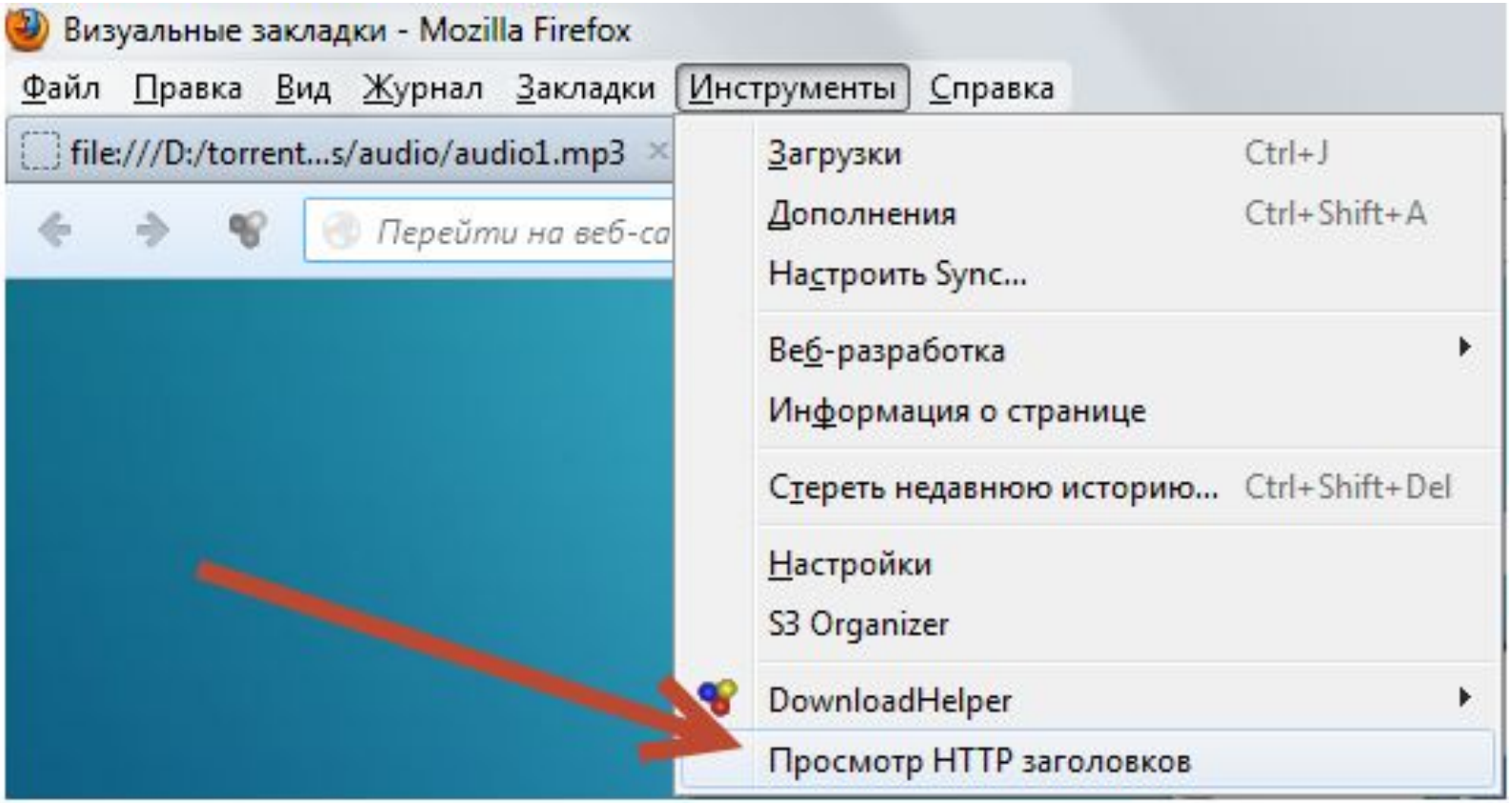
HTTP-запрос – пример 2

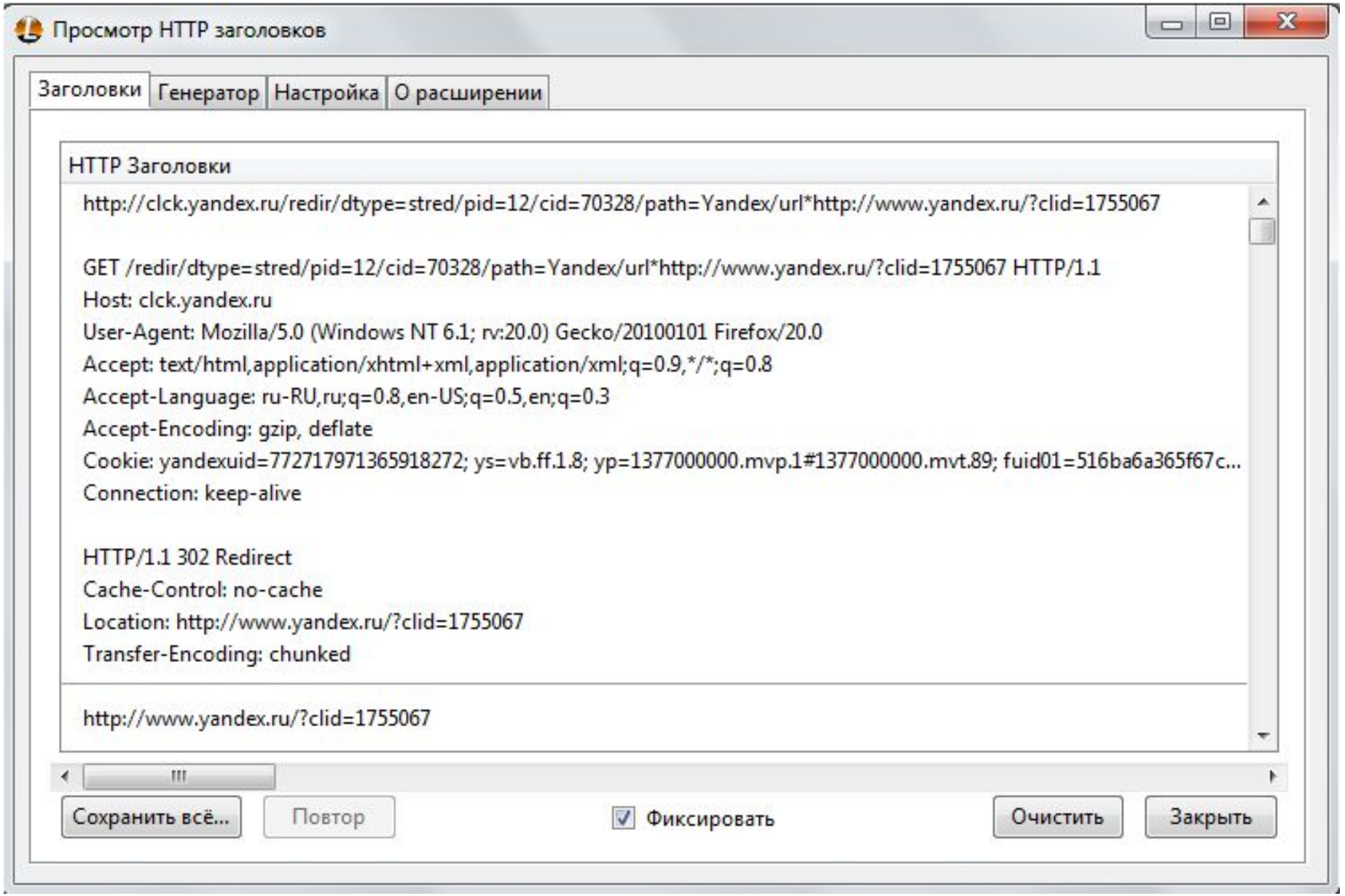
```
POST /login.aspx HTTP/1.1
Host: mysite.azure.com
User-Agent: Mozilla/5.0
Content-Length: 22

User=TEST&Pass=123456
```

HTTP-ответ – пример 1

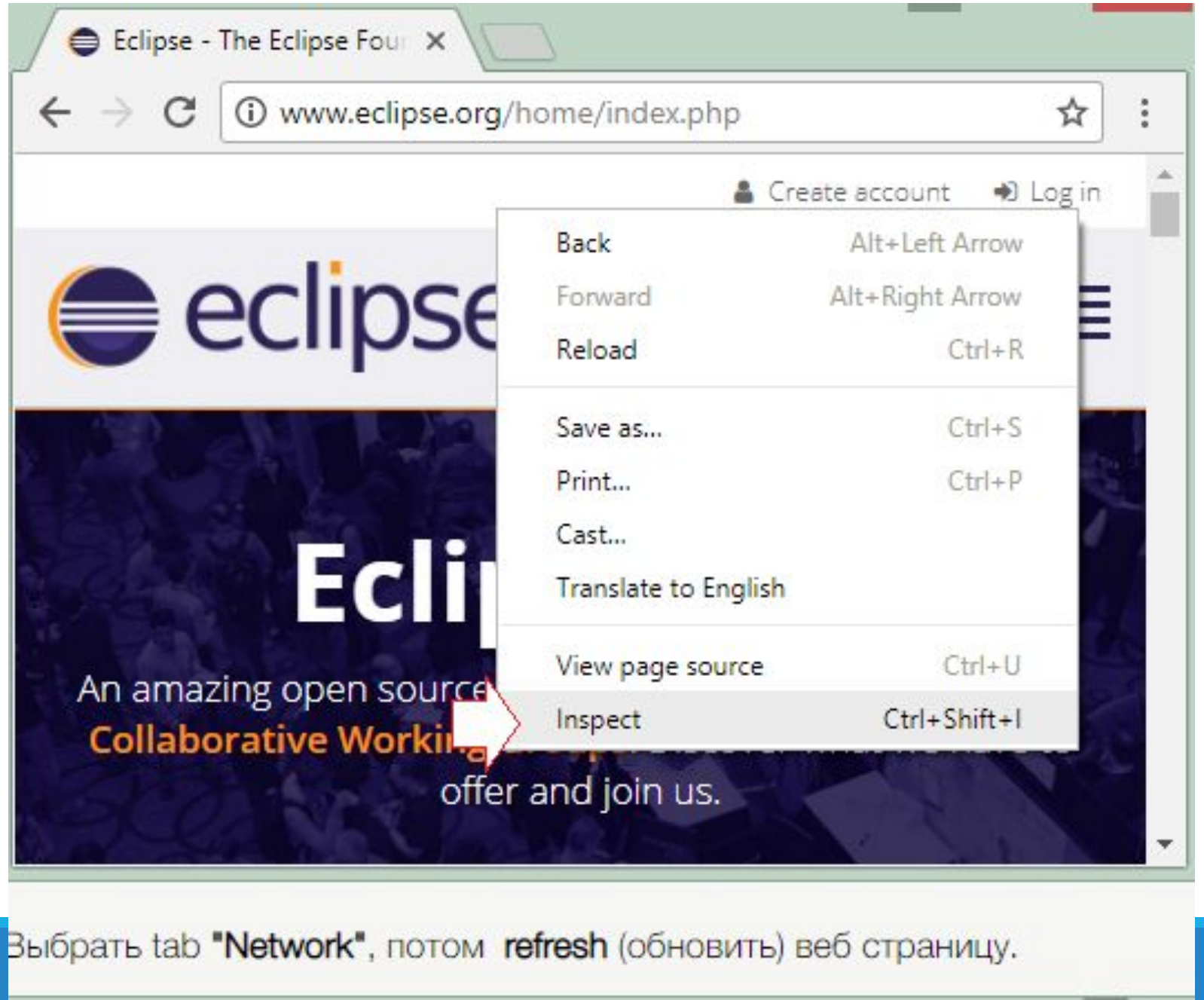
HTTP/1.1 301 Moved Permanently
Server: nginx
Date: Mon, 18 May 2015 11:59:09 GMT
Content-Type: text/html
Content-Length: 178
Connection: close
Location: http://www.tut.by/index.html

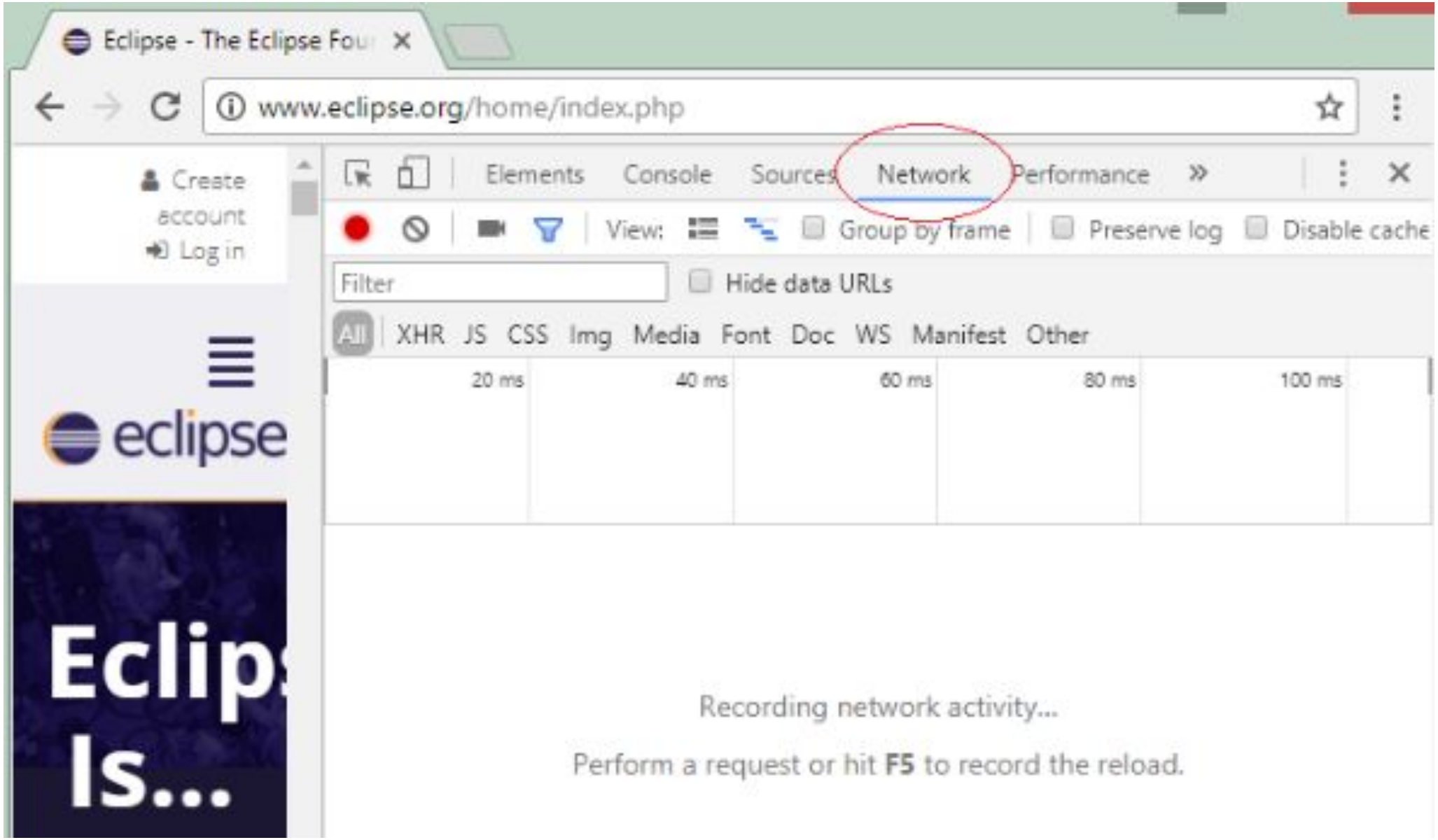
<html>
<head><title>301 Moved Permanently</title></head>
<body bgcolor="white">
<h1>301 Moved Permanently</h1>
<hr> nginx
</body>
</html>

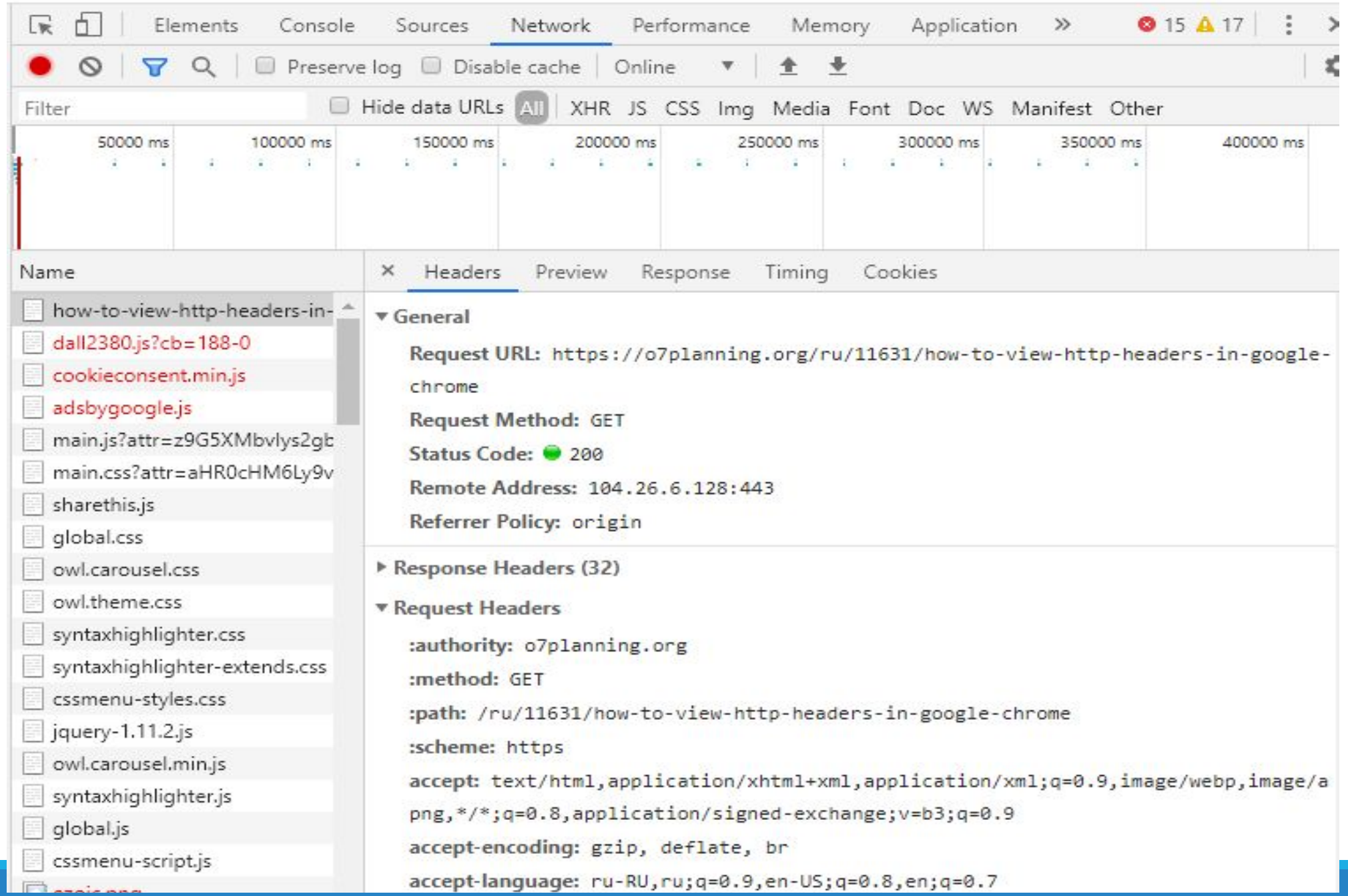

Визуальные закладки - Mozilla Firefox
Файл Правка Вид Журнал Закладки Инструменты Справка
file:///D:/torrent...s/audio/audio1.mp3
Перейти на веб-са
Загрузки Ctrl+J
Дополнения Ctrl+Shift+A
Настроить Sync...
Веб-разработка
Информация о странице
Стереть недавнюю историю... Ctrl+Shift+Del
Настройки
S3 Organizer
DownloadHelper
Просмотр HTTP заголовков

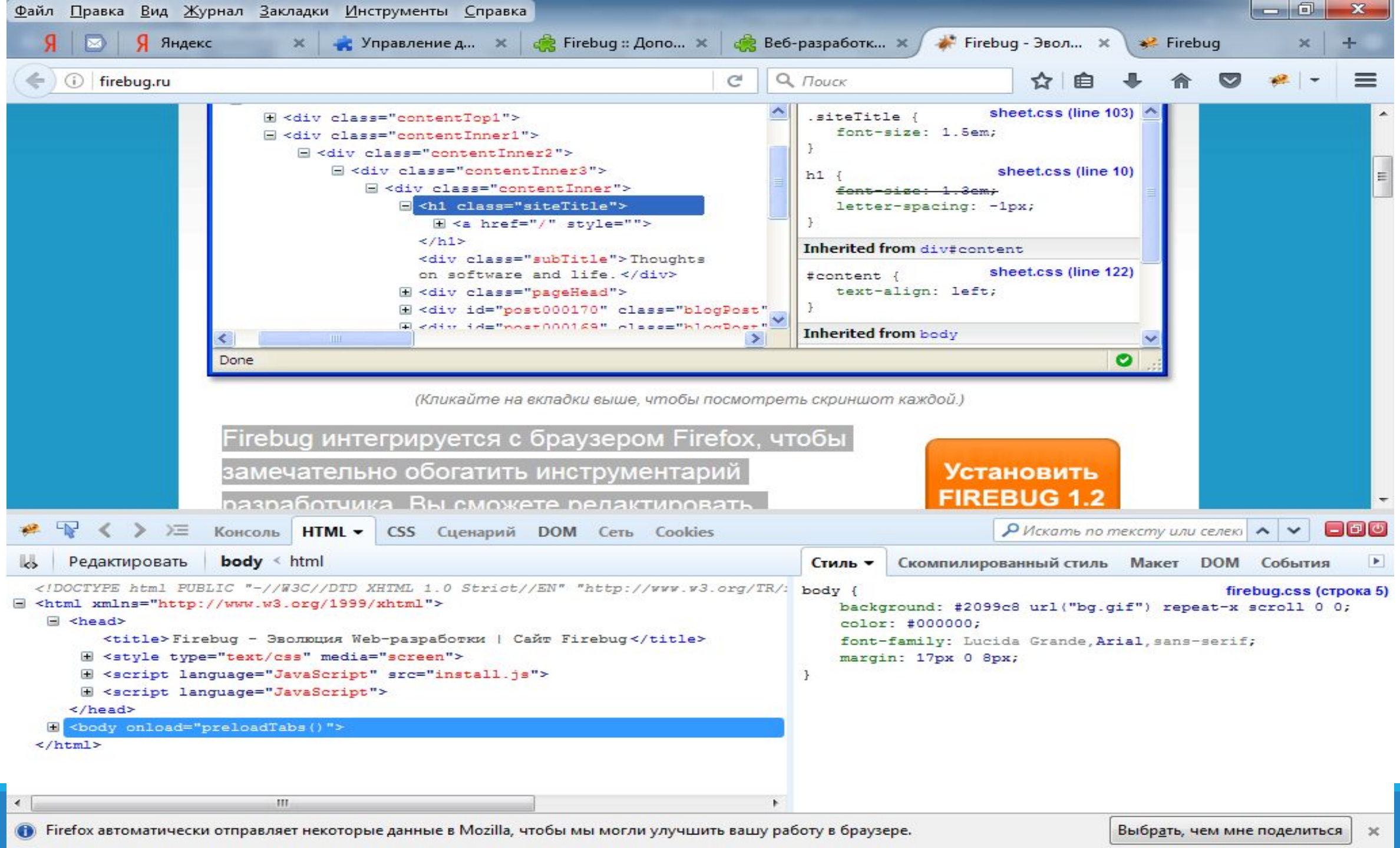

Просмотр HTTP заголовков
Заголовки
Генератор
Настройка
О расширении
HTTP Заголовки
http://clck.yandex.ru/redir/dtype=stred/pid=12/cid=70328/path=Yandex/url*http://www.yandex.ru/?clid=1755067
GET /redir/dtype=stred/pid=12/cid=70328/path=Yandex/url*http://www.yandex.ru/?clid=1755067 HTTP/1.1
Host: clck.yandex.ru
User-Agent: Mozilla/5.0 (Windows NT 6.1; rv:20.0) Gecko/20100101 Firefox/20.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: ru-RU,ru;q=0.8,en-US;q=0.5,en;q=0.3
Accept-Encoding: gzip, deflate
Cookie: yandexuid=772717971365918272; ys=vb.ff.1.8; yp=1377000000.mvp.1#1377000000.mvt.89; fuid01=516ba6a365f67c...
Connection: keep-alive
HTTP/1.1 302 Redirect
Cache-Control: no-cache
Location: http://www.yandex.ru/?clid=1755067
Transfer-Encoding: chunked
http://www.yandex.ru/?clid=1755067
Сохранить всё...
Повтор
Фиксировать
Очистить
Закрыть

Просмотр информации Http Headers в Chrome



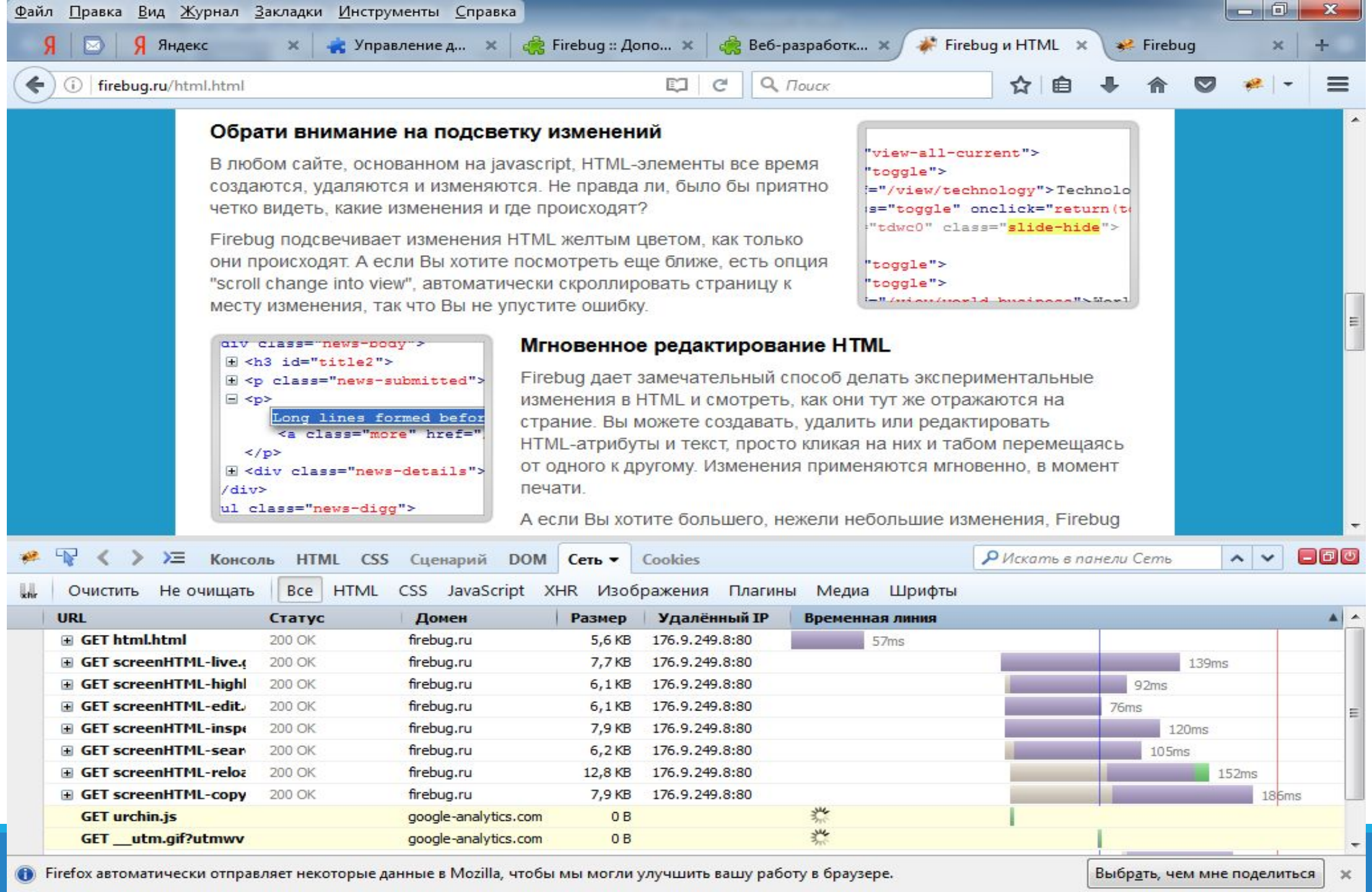

Eclipse - The Eclipse Fou
www.eclipse.org/home/index.php
Create account
Log in
eclipse
Eclip
Is...
Elements
Console
Sources
Network
Performance
View:
Group by frame
Preserve log
Disable cache
Filter
Hide data URLs
All
XHR
JS
CSS
Img
Media
Font
Doc
WS
Manifest
Other
20 ms
40 ms
60 ms
80 ms
100 ms
Recording network activity...
Perform a request or hit F5 to record the reload.

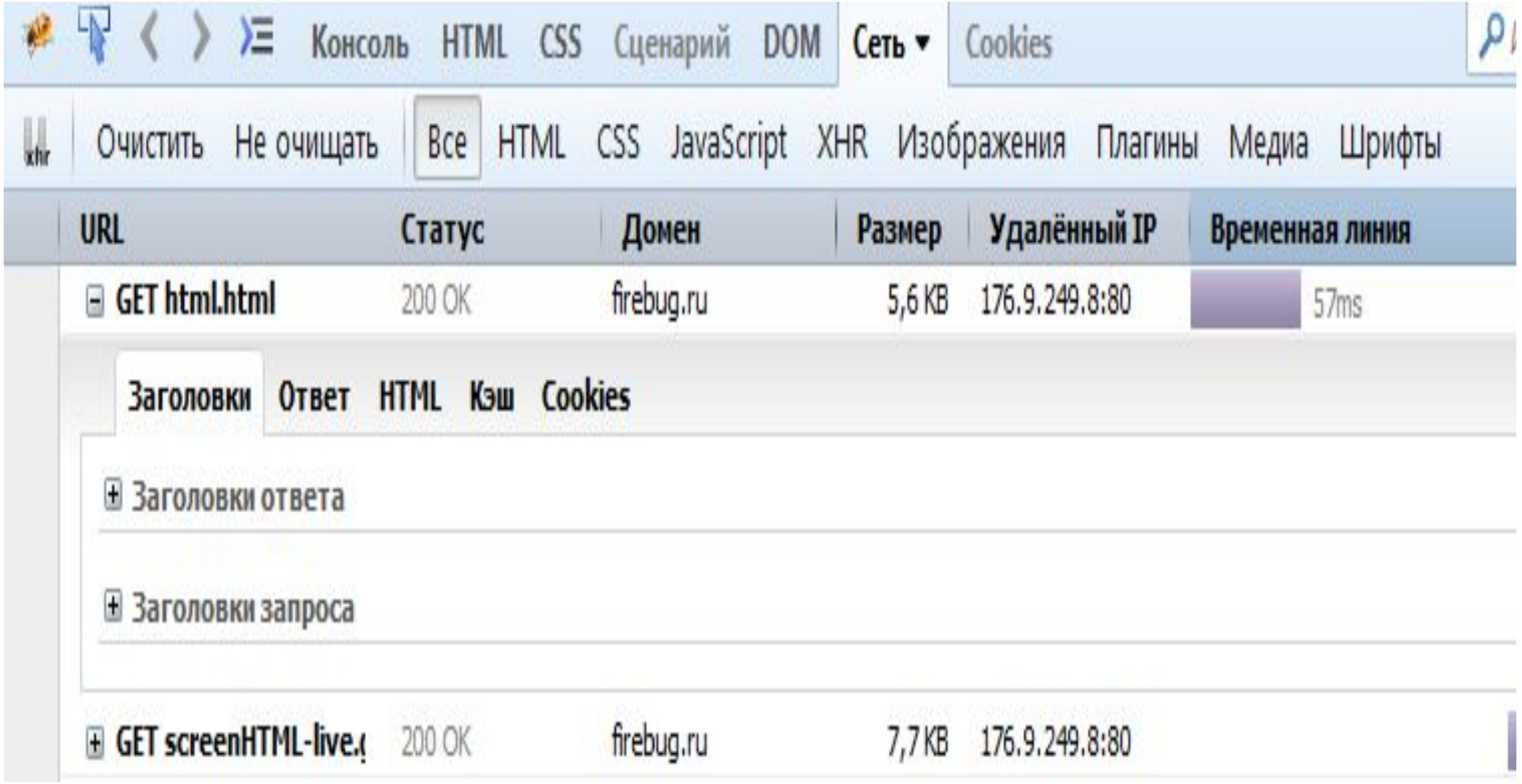

Elements
Console
Sources
Network
Performance
Memory
Application
15
17
Preserve log
Disable cache
Online
Filter
Hide data URLs
All
XHR
JS
CSS
Img
Media
Font
Doc
WS
Manifest
Other
50000 ms
100000 ms
150000 ms
200000 ms
250000 ms
300000 ms
350000 ms
400000 ms
Name
how-to-view-http-headers-in-
dall2380.js?cb=188-0
cookieconsent.min.js
adsbygoogle.js
main.js?attr=z9G5XMbvlys2gb
main.css?attr=aHR0cHM6Ly9v
sharethis.js
global.css
owl.carousel.css
owl.theme.css
syntaxhighlighter.css
syntaxhighlighter-extends.css
cssmenu-styles.css
jquery-1.11.2.js
owl.carousel.min.js
syntaxhighlighter.js
global.js
cssmenu-script.js
Headers
Preview
Response
Timing
Cookies
General
Request URL: https://o7planning.org/ru/11631/how-to-view-http-headers-in-google-chrome
Request Method: GET
Status Code: 200
Remote Address: 104.26.6.128:443
Referrer Policy: origin
Response Headers (32)
Request Headers
:authority: o7planning.org
:method: GET
:path: /ru/11631/how-to-view-http-headers-in-google-chrome
:scheme: https
accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9
accept-encoding: gzip, deflate, br
accept-language: ru-RU,ru;q=0.9,en-US;q=0.8,en;q=0.7

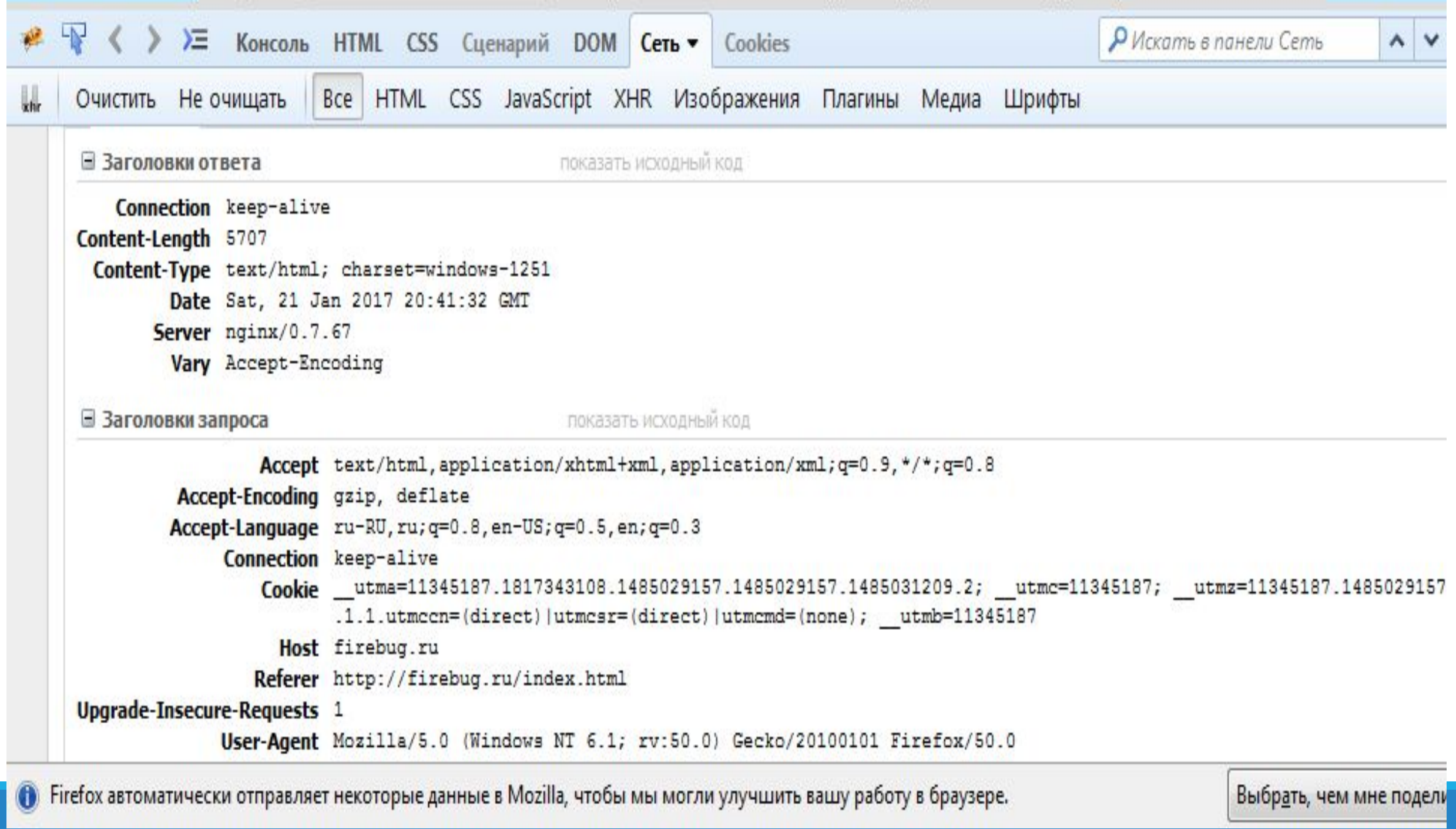

Файл Правка Вид Журнал Закладки Инструменты Справка
Яндекс
Управление д...
Firebug :: Допо...
Веб-разработк...
Firebug - Эвол...
Firebug
firebug.ru
Поиск

<h1 class="siteTitle">
<a href="/" style="">
</h1>
Thoughts on software and life.

<div id="post000170" class="blogPost"
.siteTitle { sheet.css (line 103)
font-size: 1.5em;
}
h1 { sheet.css (line 10)
font-size: 1.3em;
letter-spacing: -1px;
}
Inherited from div#content
#content { sheet.css (line 122)
text-align: left;
}
Inherited from body
Done
(Кликайте на вкладки выше, чтобы посмотреть скриншот каждой.)
Firebug интегрируется с браузером Firefox, чтобы замечательно обогатить инструментарий разработчика. Вы сможете редактировать
Установить FIREBUG 1.2
Консоль
HTML
CSS
Сценарий
DOM
Сеть
Cookies
Искать по тексту или селек
Редактировать
body < html
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" "http://www.w3.org/TR/
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<title>Firebug - Эволюция Web-разработки | Сайт Firebug</title>
<style type="text/css" media="screen">
<script language="JavaScript" src="install.js">
<script language="JavaScript">
</head>
<body onload="preloadTabs()">
</html>
Стиль
Скомпилированный стиль
Макет
DOM
События
body { firebug.css (строка 5)
background: #2099c8 url("bg.gif") repeat-x scroll 0 0;
color: #000000;
font-family: Lucida Grande,Arial,sans-serif;
margin: 17px 0 8px;
}
Firefox автоматически отправляет некоторые данные в Mozilla, чтобы мы могли улучшить вашу работу в браузере.
Выбрать, чем мне поделиться


Файл Правка Вид Журнал Закладки Инструменты Справка
Яндекс
Управление д...
Firebug :: Допо...
Веб-разработк...
Firebug и HTML
Firebug
firebug.ru/html.html
Поиск
Обрати внимание на подсветку изменений
В любом сайте, основанном на javascript, HTML-элементы все время создаются, удаляются и изменяются. Не правда ли, было бы приятно четко видеть, какие изменения и где происходят?
Firebug подсвечивает изменения HTML желтым цветом, как только они происходят. А если Вы хотите посмотреть еще ближе, есть опция "scroll change into view", автоматически скроллировать страницу к месту изменения, так что Вы не упустите ошибку.
Мгновенное редактирование HTML
Firebug дает замечательный способ делать экспериментальные изменения в HTML и смотреть, как они тут же отражаются на страние. Вы можете создавать, удалить или редактировать HTML-атрибуты и текст, просто кликая на них и табом перемещаясь от одного к другому. Изменения применяются мгновенно, в момент печати.
А если Вы хотите большего, нежели небольшие изменения, Firebug
Консоль HTML CSS Сценарий DOM Сеть Cookies
Искать в панели Сеть
Очистить Не очищать Все HTML CSS JavaScript XHR Изображения Плагины Медиа Шрифты
URL Статус Домен Размер Удалённый IP Временная линия
GET html.html 200 OK firebug.ru 5,6 KB 176.9.249.8:80 57ms
GET screenHTML-live.(200 OK firebug.ru 7,7 KB 176.9.249.8:80 139ms
GET screenHTML-highl 200 OK firebug.ru 6,1 KB 176.9.249.8:80 92ms
GET screenHTML-edit. 200 OK firebug.ru 6,1 KB 176.9.249.8:80 76ms
GET screenHTML-inspe 200 OK firebug.ru 7,9 KB 176.9.249.8:80 120ms
GET screenHTML-sear 200 OK firebug.ru 6,2 KB 176.9.249.8:80 105ms
GET screenHTML-reloa 200 OK firebug.ru 12,8 KB 176.9.249.8:80 152ms
GET screenHTML-copy 200 OK firebug.ru 7,9 KB 176.9.249.8:80 186ms
GET urchin.js google-analytics.com 0 B
GET __utm.gif?utmwv google-analytics.com 0 B
Firefox автоматически отправляет некоторые данные в Mozilla, чтобы мы могли улучшить вашу работу в браузере.
Выбрать, чем мне поделиться


Консоль HTML CSS Сценарий DOM Сеть Cookies
Очистить Не очищать Все HTML CSS JavaScript XHR Изображения Плагины Медиа Шрифты
URL Статус Домен Размер Удалённый IP Временная линия
GET html.html 200 OK firebug.ru 5,6 KB 176.9.249.8:80 57ms
Заголовки Ответ HTML Кэш Cookies
Заголовки ответа
Заголовки запроса
GET screenHTML-live.(200 OK firebug.ru 7,7 KB 176.9.249.8:80


Консоль HTML CSS Сценарий DOM Сеть Cookies
Искать в панели Сеть
Очистить Не очищать Все HTML CSS JavaScript XHR Изображения Плагины Медиа Шрифты
Заголовки ответа
показать исходный код
Connection keep-alive
Content-Length 5707
Content-Type text/html; charset=windows-1251
Date Sat, 21 Jan 2017 20:41:32 GMT
Server nginx/0.7.67
Vary Accept-Encoding
Заголовки запроса
показать исходный код
Accept text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Encoding gzip, deflate
Accept-Language ru-RU,ru;q=0.8,en-US;q=0.5,en;q=0.3
Connection keep-alive
Cookie __utma=11345187.1817343108.1485029157.1485029157.1485031209.2; __utmc=11345187; __utmz=11345187.1485029157.1.1.utmccn=(direct)|utmcsr=(direct)|utmcmd=(none); __utmb=11345187
Host firebug.ru
Referer http://firebug.ru/index.html
Upgrade-Insecure-Requests 1
User-Agent Mozilla/5.0 (Windows NT 6.1; rv:50.0) Gecko/20100101 Firefox/50.0
Firefox автоматически отправляет некоторые данные в Mozilla, чтобы мы могли улучшить вашу работу в браузере.
Выбрать, чем мне подели