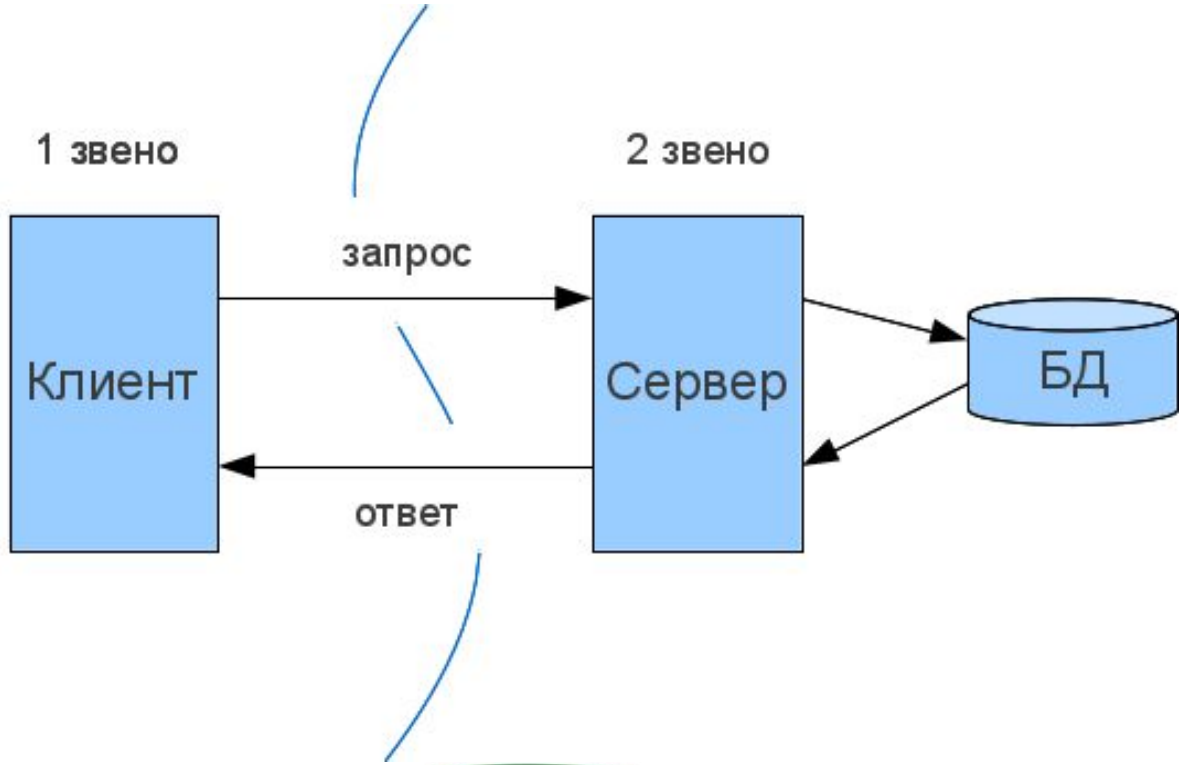
Теоретический курс

Содержание курса

1. Архитектура «клиент – сервер»
2. MVC. Толстый тонкий клиент
3. Что происходит при нажатии кнопки на сайте?
4. Основные сетевые протоколы
5. Запись трафика

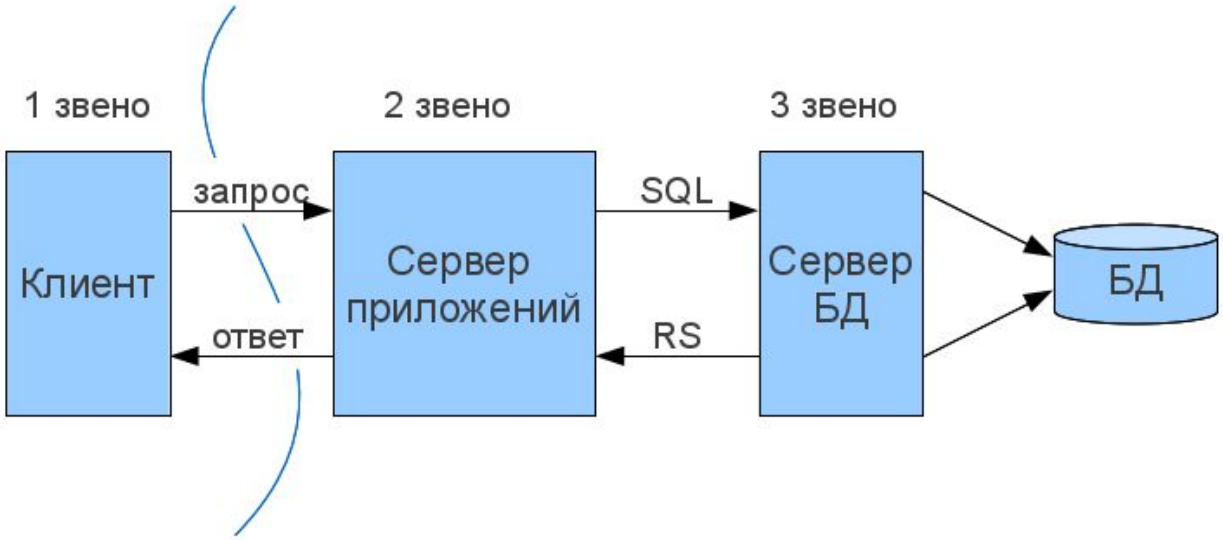
Архитектура «клиент-сервер»

Двухзвенная:



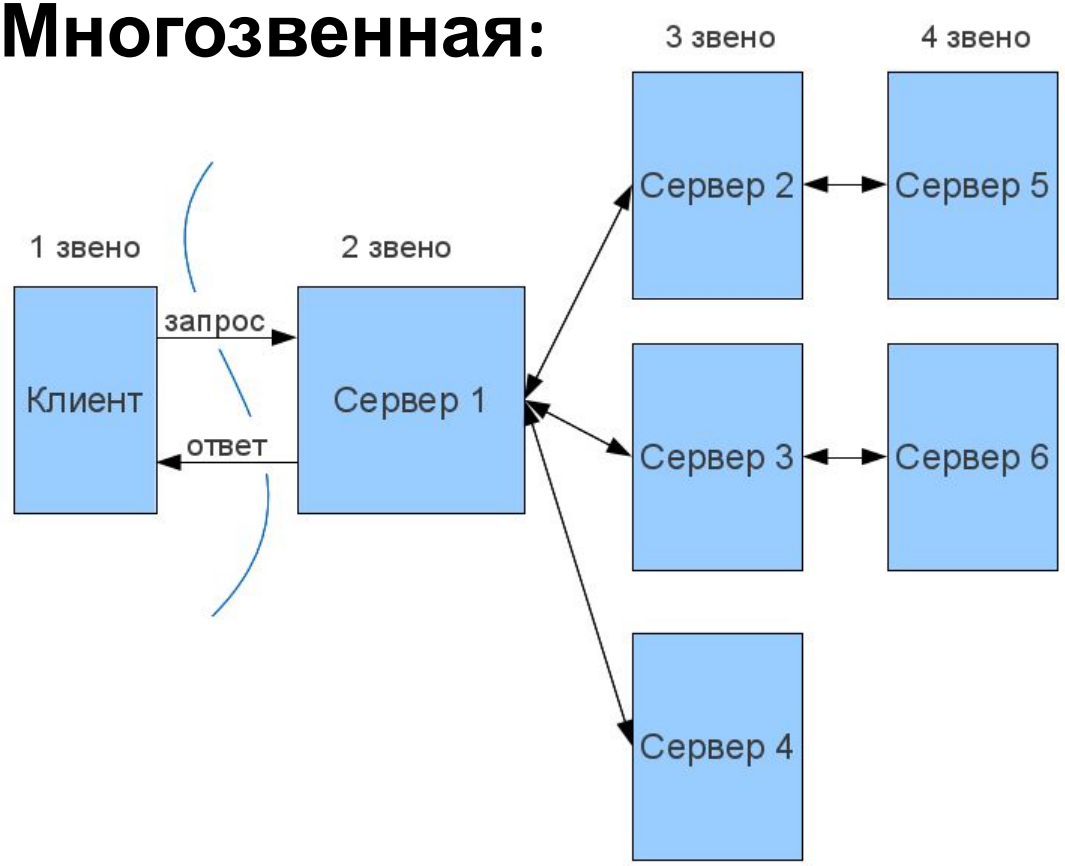
Архитектура «клиент-сервер»

Трехзвенная:



Архитектура «клиент-сервер»

Многозвенная:



Архитектура «клиент-сервер»

Сравнение:

Двухзвенная:

+ простота

- менее надежная

Трехзвенная:

+ гибкая и масштабируемая

+ высокая безопасность

+ высокая производительность

«Тонкий» и «Толстый» клиенты

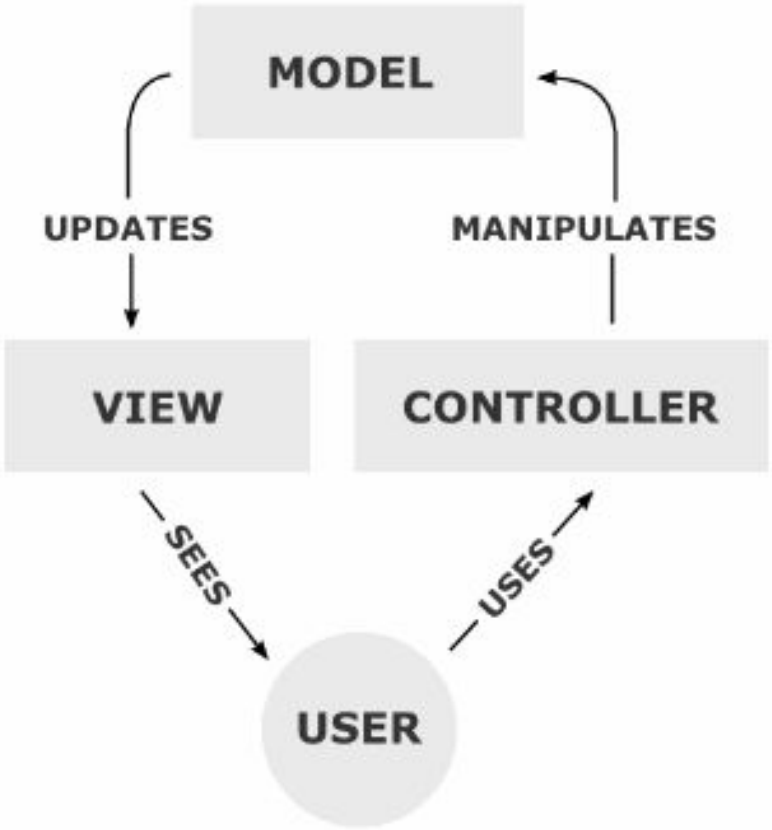
«Тонкий»:

- Только для отображения получаемой информации
- Все вычисления на сервере

«Толстый»:

- Расширенный функционал
- Большая часть данных обрабатывается на компьютере пользователя
- Локальная база данных
- Работа в оффлайн режиме

Model-View-Controller



Model

- Модель — это бизнес-логика приложения;
- Модель обладает знаниями о себе самой и не знает о контроллерах и представлениях;
- Для некоторых проектов модель — это просто слой данных (DAO, база данных, XML-файл);
- Для других проектов модель — это менеджер базы данных, набор объектов или просто логика приложения;

View

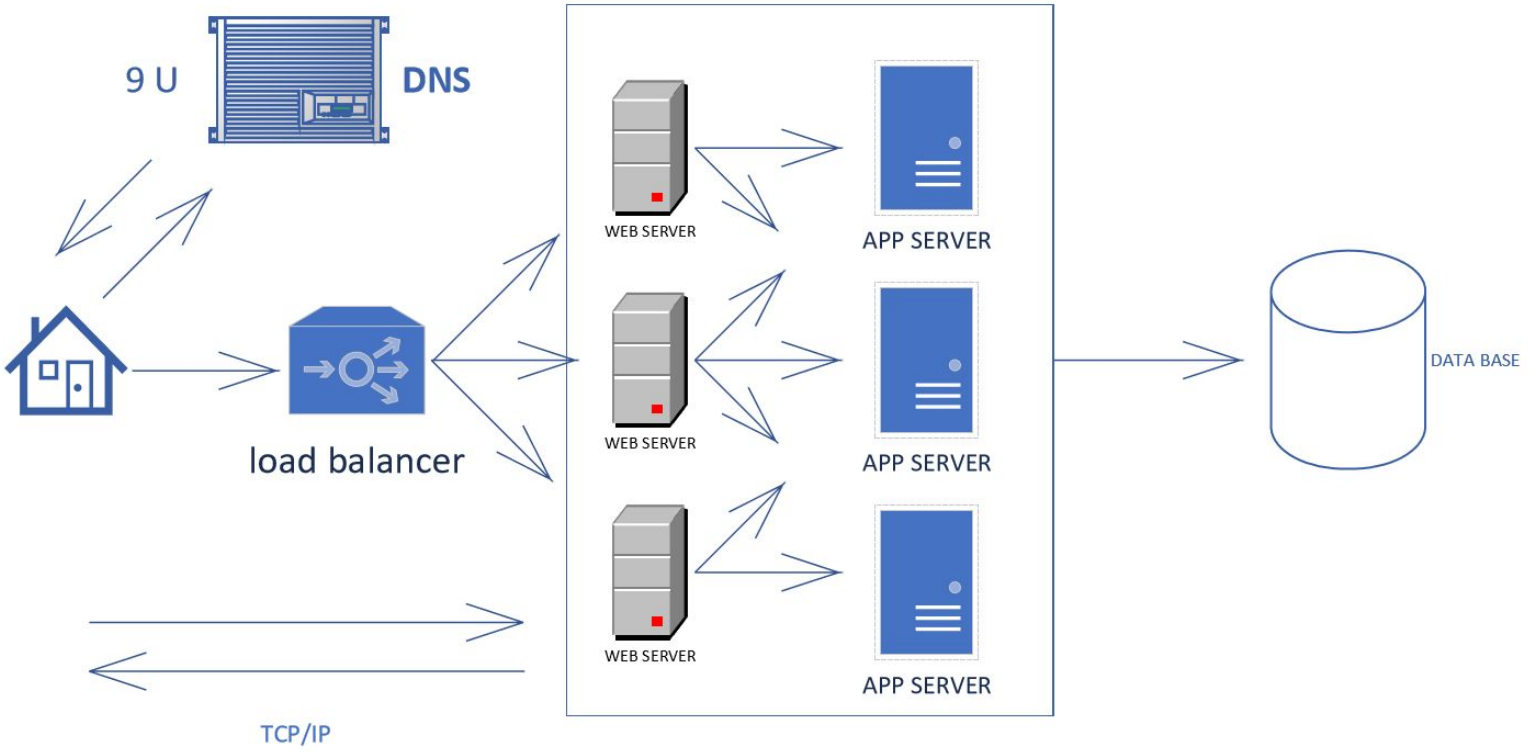
- В представлении реализуется отображение данных, которые получаются от модели любым способом;
- *В некоторых случаях, представление может иметь код, который реализует некоторую бизнес-логику.*

Примеры представления: HTML-страница, WPF форма, Windows Form.

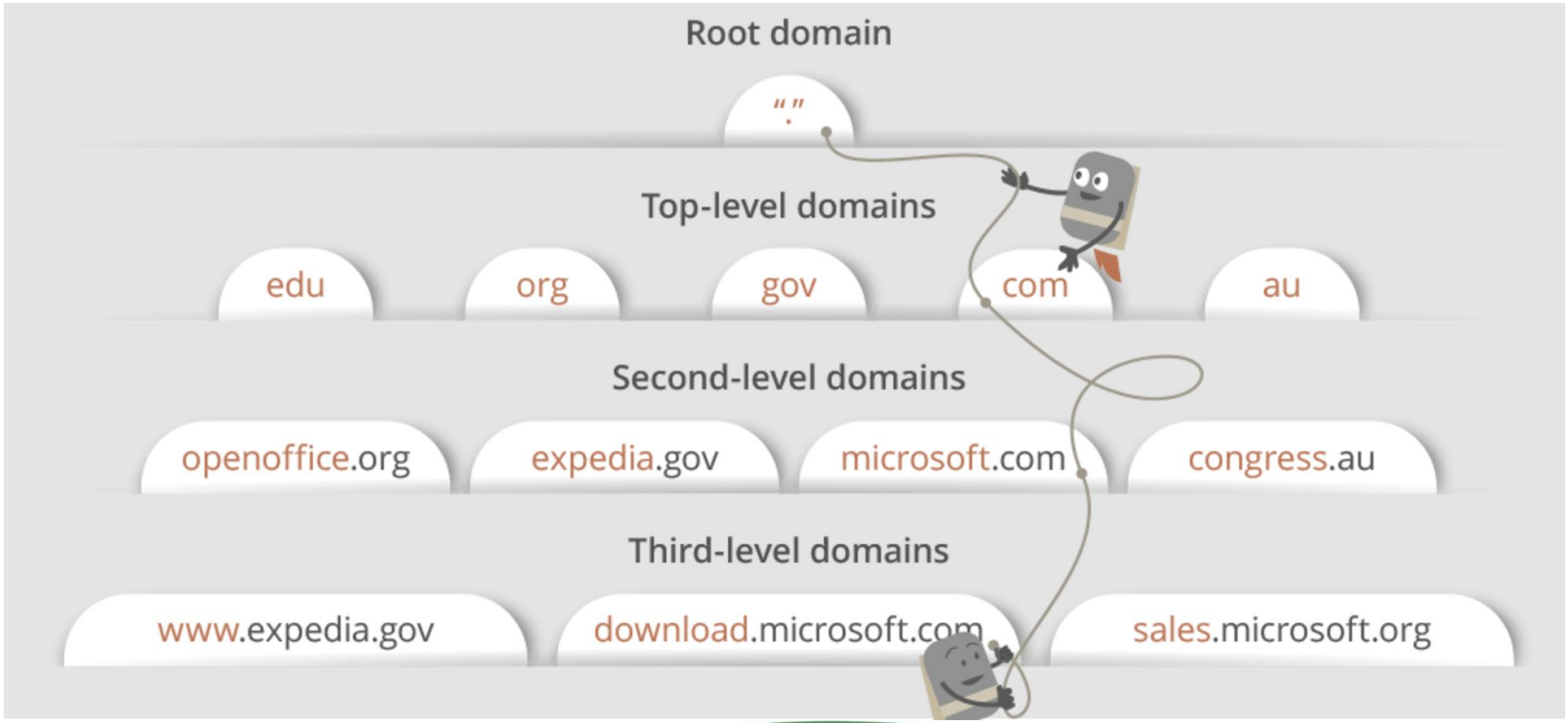
Controller

- Контроллер определяет, какие представление должно быть отображено в данный момент;
- События представления могут повлиять только на контроллер. контроллер может повлиять на модель и определить другое представление.
- Возможно несколько представлений только для одного контроллера;

Схема запроса



Архитектура домена



TCP/IP Трехстороннее рукопожатие

- 1. Клиентский компьютер отправляет пакет SYN на сервер через Интернет, спрашивая, открыт ли он для новых подключений.
- 2. Если сервер имеет открытые порты, которые могут принимать и инициировать новые подключения, он ответит ACKnowledgment пакета SYN, используя пакет SYN / ACK.
- 3. Клиент получит пакет SYN / ACK с сервера и подтвердит его, отправив пакет ACK.

Затем устанавливается TCP-соединение для передачи данных!

Алгоритмы балансировки

- Round Robin
- Weighted Round Robin
- Least Connections
- Destination Hash Scheduling и Source Hash Scheduling
- Sticky Sessions

Основные сетевые протоколы

- TCP/IP
- Http/Https
- FTP
- POP3
- SMPT
- TELNET
- SSH
- JDBC
- JMS

Http протокол

Каждое HTTP-сообщение состоит из трёх частей, которые передаются в указанном порядке:

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

Методы http запроса

- OPTIONS
- GET
- HEAD
- POST
- PUT
- PATCH
- DELETE
- TRACE
- CONNECT

Код	Класс	Назначение
`1xx`	Информационный (англ. **informational**)	Информирование о процессе передачи. В HTTP/1.0 — сообщения с такими кодами должны игнорироваться. В HTTP/1.1 — клиент должен быть готов принять этот класс сообщений как обычный ответ, но ничего отправлять серверу не нужно. Сами сообщения от сервера содержат только стартовую строку ответа и, если требуется, несколько специфичных для ответа полей заголовка. Прокси-серверы подобные сообщения должны отправлять дальше от сервера к клиенту.
`2xx`	Успех (англ. **Success**)	Информирование о случаях успешного принятия и обработки запроса клиента. В зависимости от статуса, сервер может ещё передать заголовки и тело сообщения.
`3xx`	Перенаправление (англ. **Redirection**)	Сообщает клиенту, что для успешного выполнения операции необходимо сделать другой запрос (как правило по другому URI). Из данного класса пять кодов `301`, `302`, `303`, `305` и `307` относятся непосредственно к перенаправлениям (редирект). Адрес, по которому клиенту следует произвести запрос, сервер указывает в заголовке `Location`. При этом допускается использование фрагментов в целевом URI.
`4xx`	Ошибка клиента (англ. **Client Error**)	Указание ошибок со стороны клиента. При использовании всех методов, кроме `HEAD`, сервер должен вернуть в теле сообщения гипертекстовое пояснение для пользователя.
`5xx`	Ошибка сервера (англ. **Server Error**)	Информирование о случаях неудачного выполнения операции по вине сервера. Для всех ситуаций, кроме использования метода `HEAD`, сервер должен включать в тело сообщения объяснение, которое клиент отобразит пользователю.

HTTP Headers

В зависимости от того, где эти заголовки могут находиться, они разделяются на:

- **General Headers** (*Основные заголовки*) — должны быть и в запросах и в ответах клиента и сервера.
- **Request Headers** (*Заголовки запроса*) — используются только в запросах клиента.
- **Response Headers** (*Заголовки ответа*) — используются только в ответах сервера.
- **Entity Headers** (*Заголовки сущности*) — сопровождают каждую сущность сообщения.

Тело сообщения

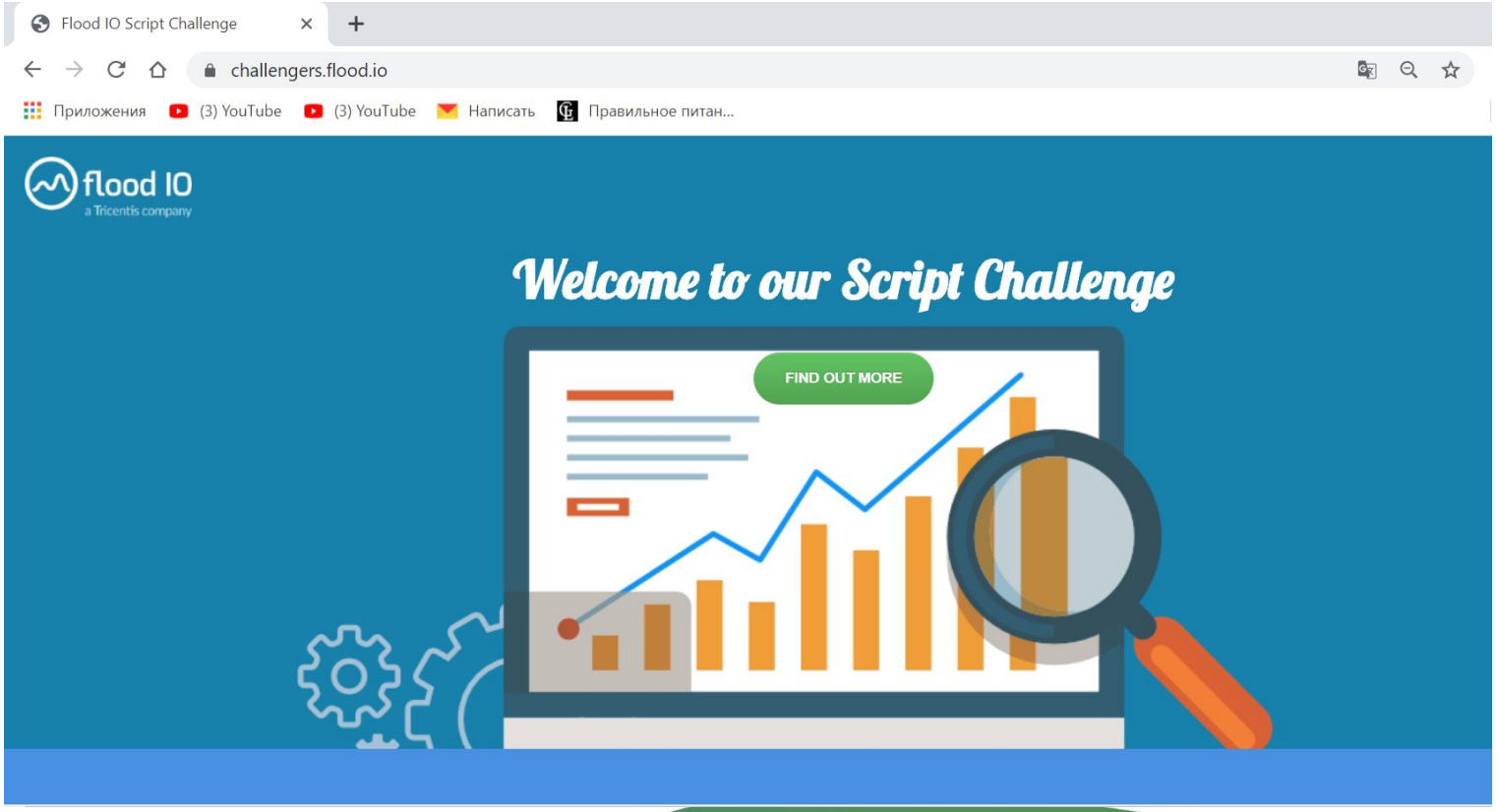
тело сообщения — это сами данные, которые передаются в запросе. Тело сообщения – это необязательный параметр и может отсутствовать.

Cookie

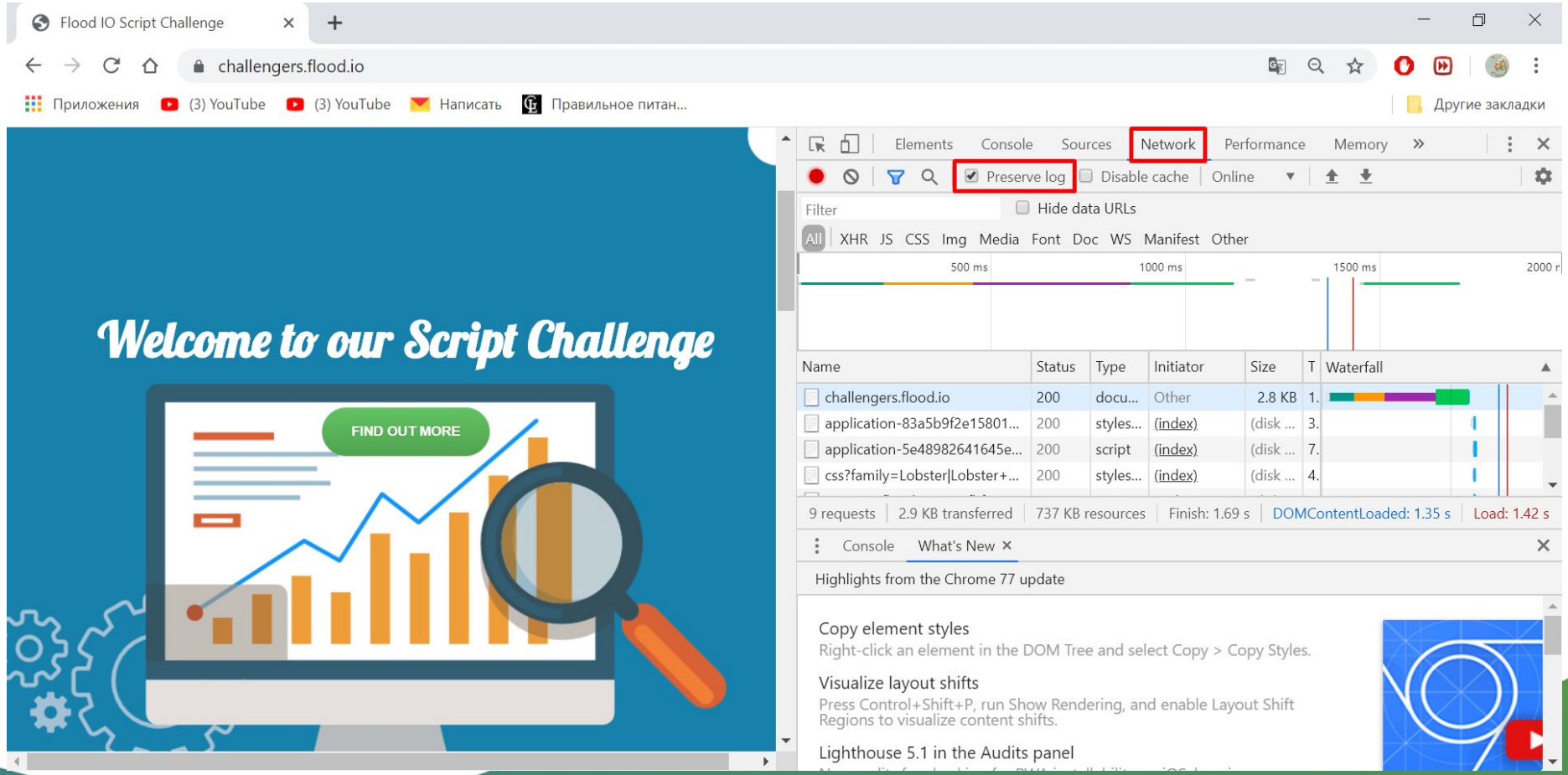
Ку́ки — небольшой фрагмент данных, отправленный веб-сервером и хранимый на компьютере пользователя. Веб-клиент всякий раз при попытке открыть страницу соответствующего сайта пересылает этот фрагмент данных веб-серверу в составе HTTP-запроса. Применяется для сохранения данных на стороне пользователя, на практике обычно используется для:

- аутентификации пользователя;
- хранения персональных предпочтений и настроек пользователя;
- отслеживания состояния сеанса доступа пользователя;
- ведения статистики о пользователях

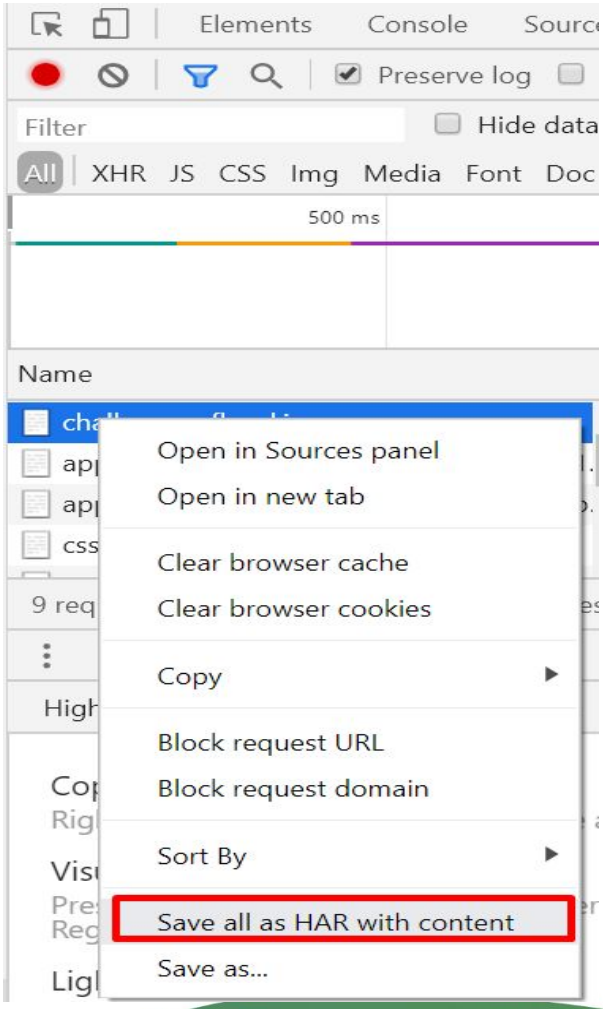
Запись трафика DevTools



Press F12


Flood IO Script Challenge
challengers.flood.io
Приложения
(3) YouTube
(3) YouTube
Написать
Правильное питан...
Другие закладки
Welcome to our Script Challenge
FIND OUT MORE
Elements
Console
Sources
Network
Performance
Memory
Preserve log
Disable cache
Online
Filter
Hide data URLs
All
XHR
JS
CSS
Img
Media
Font
Doc
WS
Manifest
Other
500 ms
1000 ms
1500 ms
2000 r
Name
Status
Type
Initiator
Size
T
Waterfall
challengers.flood.io 200 docu... Other 2.8 KB 1.
application-83a5b9f2e15801... 200 styles... (index) (disk ... 3.
application-5e48982641645e... 200 script (index) (disk ... 7.
css?family=Lobster|Lobster+... 200 styles... (index) (disk ... 4.
9 requests
2.9 KB transferred
737 KB resources
Finish: 1.69 s
DOMContentLoaded: 1.35 s
Load: 1.42 s
Console
What's New
Highlights from the Chrome 77 update
Copy element styles
Right-click an element in the DOM Tree and select Copy > Copy Styles.
Visualize layout shifts
Press Control+Shift+P, run Show Rendering, and enable Layout Shift Regions to visualize content shifts.
Lighthouse 5.1 in the Audits panel

Save all


Elements
Console
Preserve log
Filter
Hide data
All
XHR
JS
CSS
Img
Media
Font
500 ms
Name
Open in Sources panel
Open in new tab
Clear browser cache
Clear browser cookies
Copy
Block request URL
Block request domain
Sort By
Save all as HAR with content
Save as...