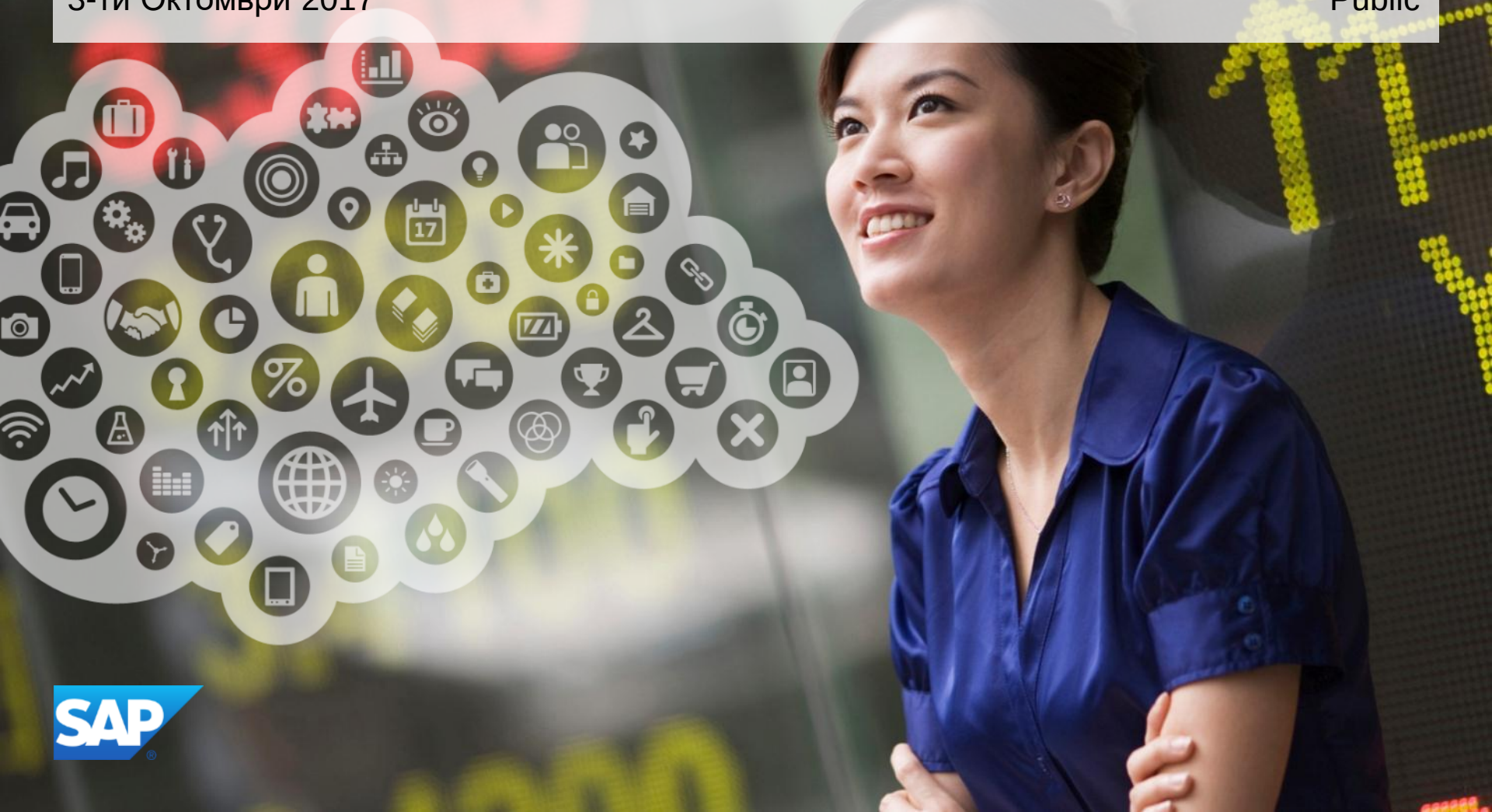
Въведение

WWW, HTTP, браузъри, инструменти за разработка.

Владислав Илиев / SAP Labs Bulgaria

3-ти Октомври 2017

Public



Съдържание

Интернет и WWW

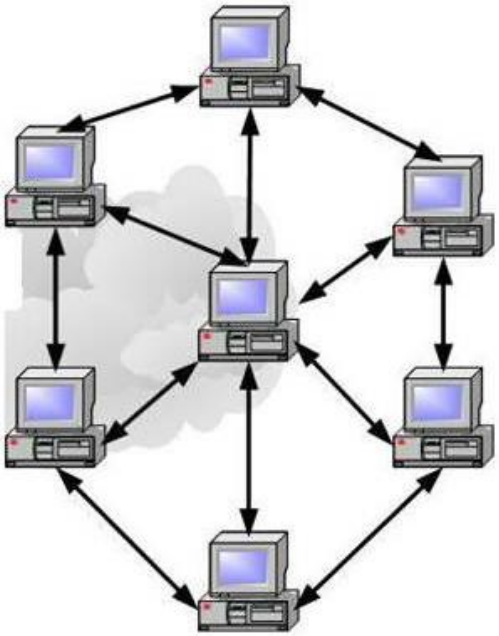
Моделът клиент-сървър

Web Browser

HTTP

Инструменти

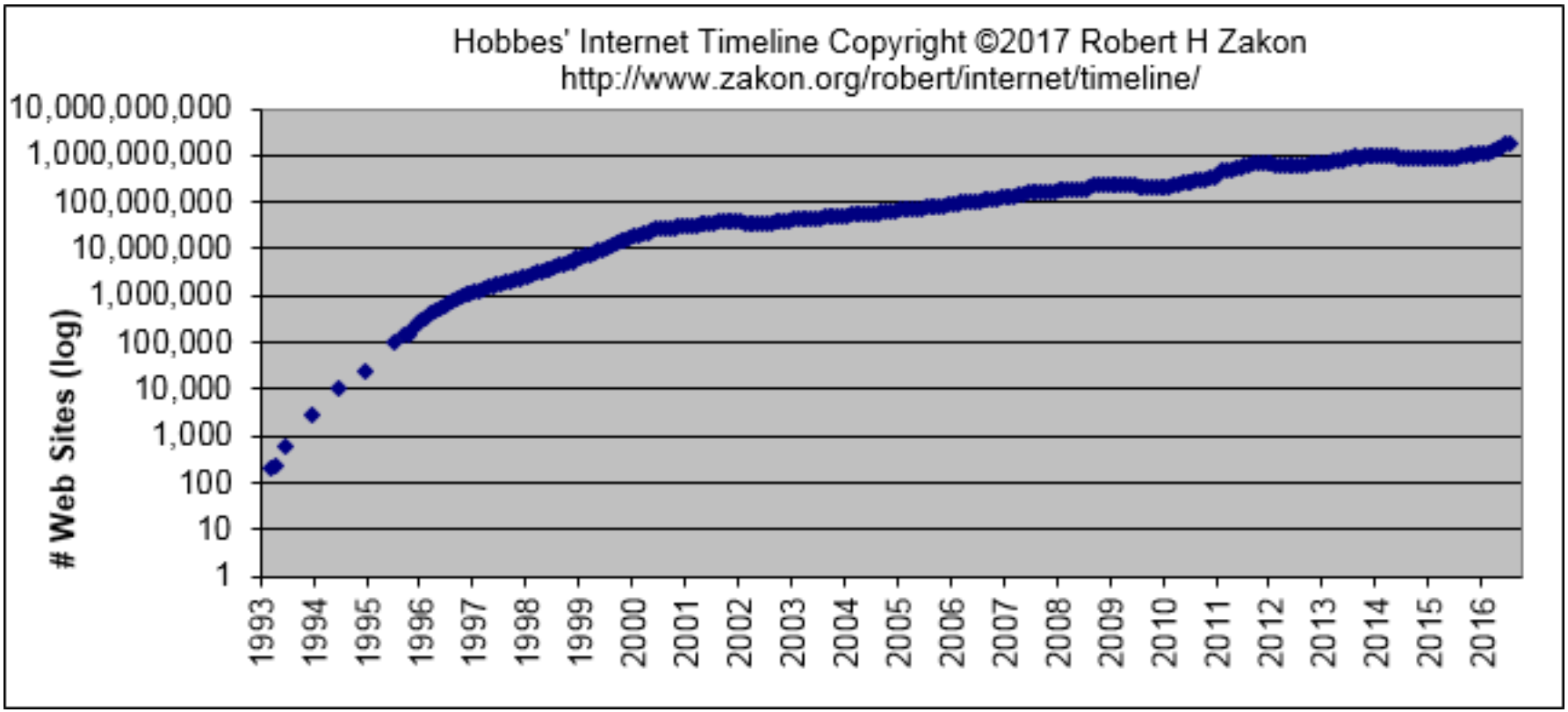
Интернет и WWW



The Internet vs World Wide Web (WWW)

Интернет и WWW

#Уебсайтове



Съдържание

Интернет и WWW

Моделът „клиент-сървър“

Web Browser

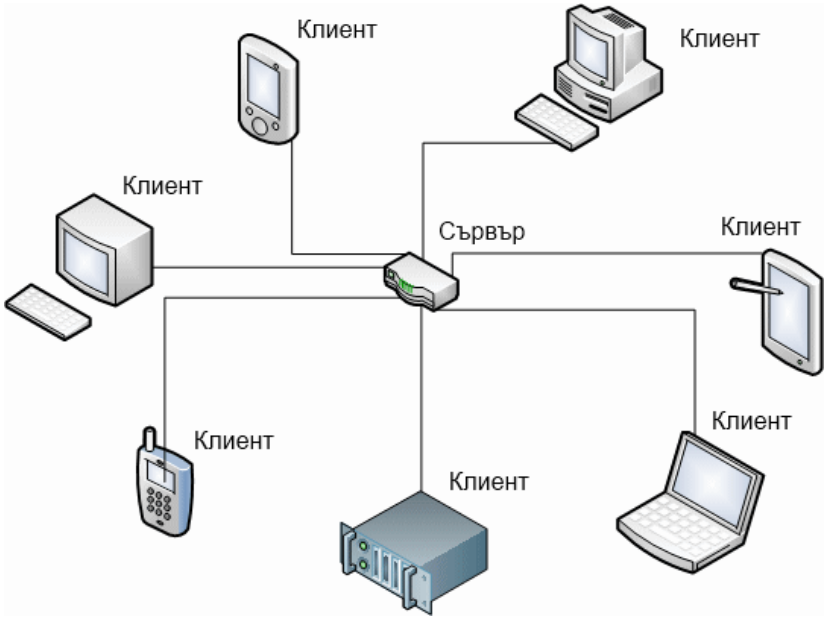
HTTP

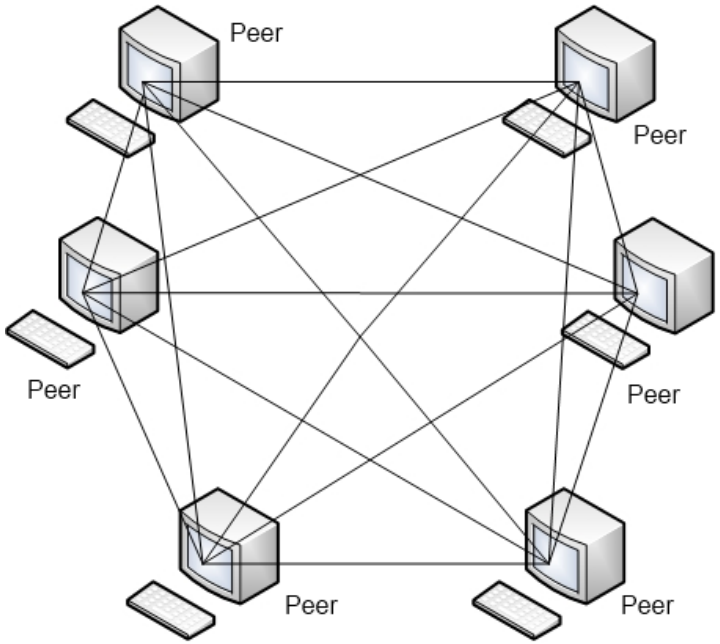
Инструменти

Моделът клиент-сървър

Архитектурни модели

- **Клиент-сървър** е разпределен изчислителен модел, при който част от задачите се разпределят между доставчиците на ресурси или услуги, наречени ***сървъри*** и консуматорите на услуги, наречени ***клиенти***.





- **Peer-to-peer** е разпределен архитектурен модел на приложение, при който задачите се разпределят по еднакъв начин между всички участници (peer, node). Всеки участник е едновременно и **клиент** и **сървър**.

Моделът клиент-сървър

Клиент-сървър – Клиенти

Според наличната функционалност в клиента:

- **Rich** клиенти.
- **Thin** клиенти.

Според семантиката (протокол):

- **Web** клиенти – Браузери (Chrome, Firefox, IE).
- **Mail** клиенти – POP/SMTP клиенти (MS Outlook, Lotus notes).
- **FTP** клиенти – Total Commander, Filezilla, WinSCP.
- …

Моделът клиент-сървър

Клиент-сървър - Сървъри

Файл сървър (Windows, Samba, UNIX NFS, OpenAFS).

DB сървър (MySQL, PostgreSQL, Oracle, MS SQL Server, Mongo DB, HANA).

Mail сървър (MS Exchange, GMail, Lotus Notes).

Name сървър (DNS).

FTP сървър (ftpd, IIS).

Print сървър.

Game сървър.

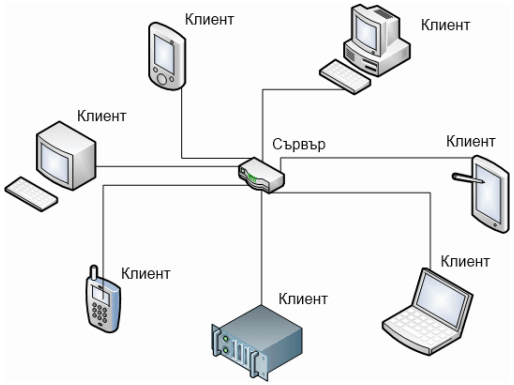
Web сървър (Apache, GWS, MS IIS, nginx).

Application сървър (SAP NetWeaver, Tomcat, GlassFish, JBoss, BEA, Oracle).

…

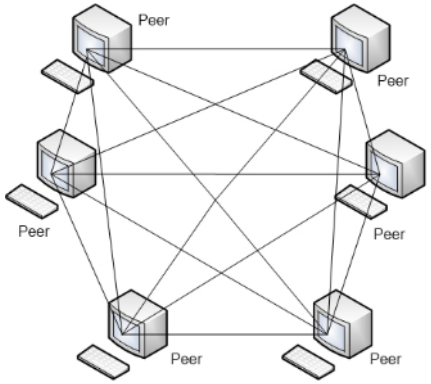
Моделът клиент-сървър

Предимства и недостатъци



Клиент-сървър

- Single Point Of Failure (SPOF).
- Увеличаването на броя на клиентите води до намаляване на производителността.
- 70-95% от времето, през което работи сървъра е idle.



Peer-to-peer

+ Няма SPOF.
+ Няма намаляване на производителността при увеличаване на клиентите.
- Проблеми със сигурността.
- Риск от умишлена промяна на съдържание.
- Липса на контрол върху съдържанието и възможност за загуба на съдържание.
- Труден процес на поддръжка.

Съдържание

Интернет и WWW

Моделът „клиент-сървър“

Web Browser

HTTP

Инструменти

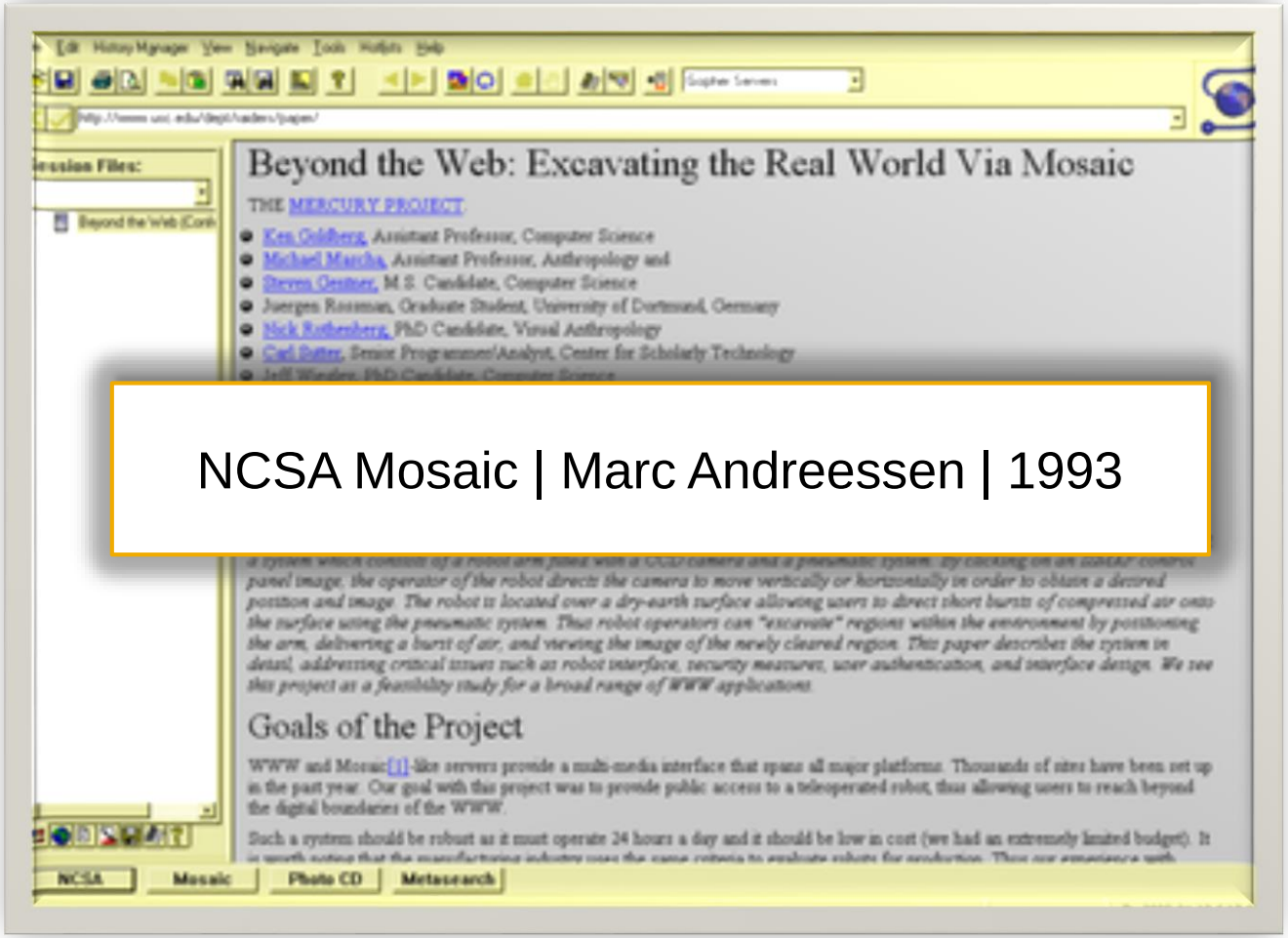
Web Browser

КАКВО Е

УЕБ БРАУЗЕР?

/Web Browser/

Web Browser



NCSA Mosaic | Marc Andreessen | 1993

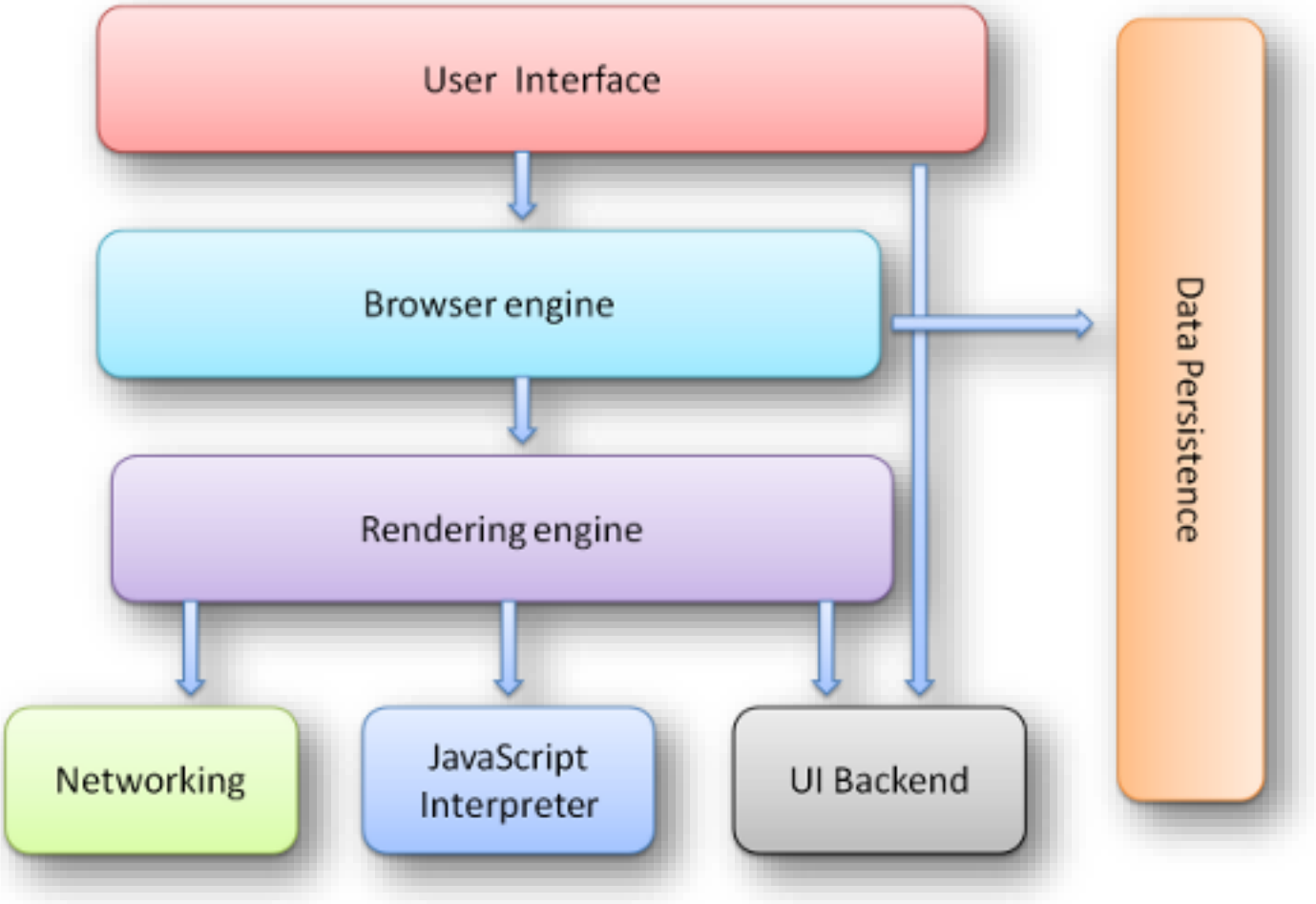
Web Browser
Развитие

Web Browser
Последни версии

към 27.09.2017

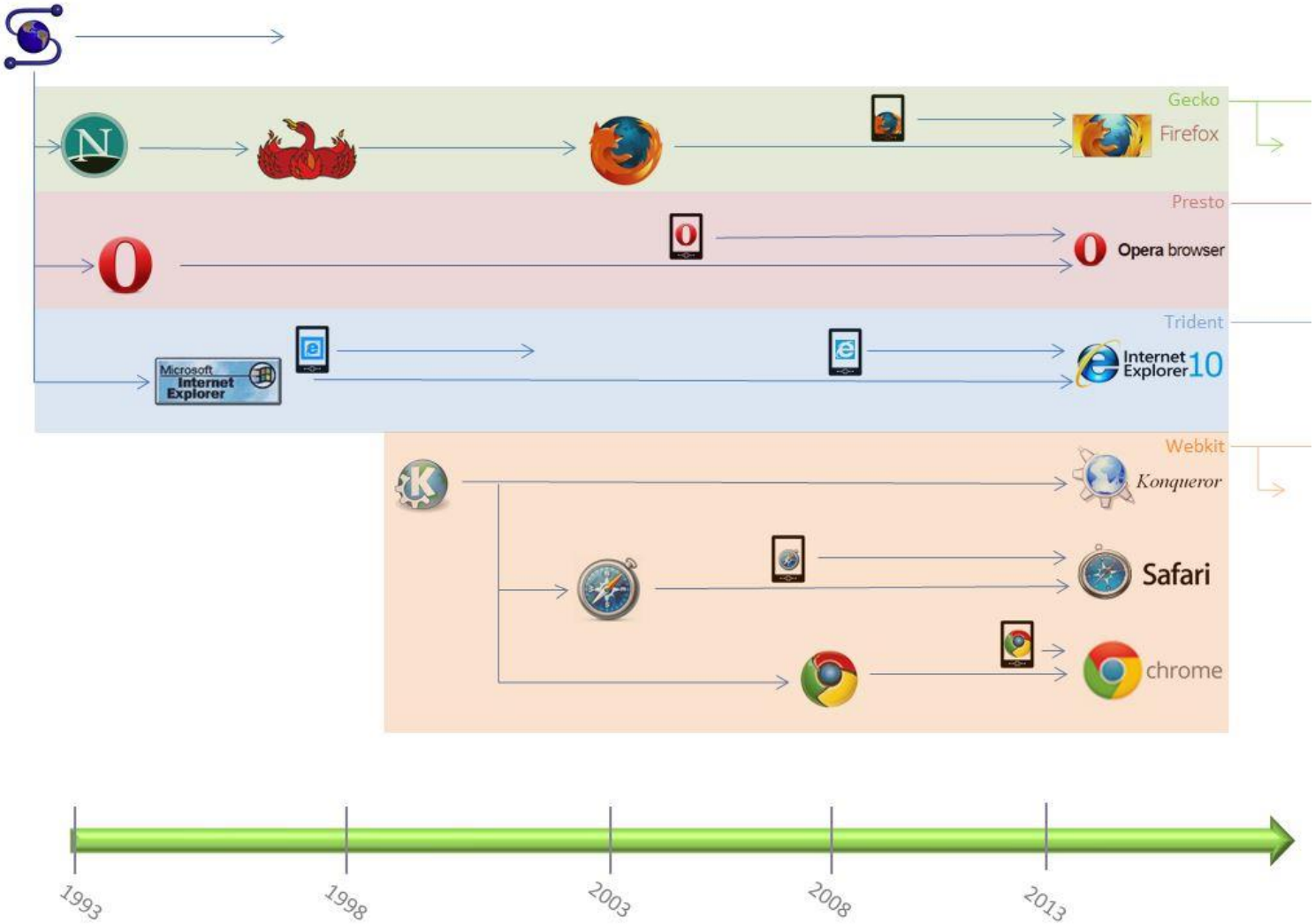
Източник: https://browsehappy.com

Web Browser
Архитектура



Web Browser

Browser and Rendering Engines



Съдържание

Интернет и WWW

Моделът „клиент-сървър“

Web Browser

HTTP

Инструменти

Протоколи

ЩО Е ТО

МРЕЖОВИ ПРОТОКОЛ?

Протоколи според OSI (Open Systems Interconnection) модел

№	Слой	Описание	Протоколи
7	Application	Позволява на потребителските приложения да заявяват услуги или информация, а на сървър приложенията — да се регистрират и предоставят услуги в мрежата.	DNS, FTP, HTTP, NFS, NTP, DHCP, SMTP, Telnet
6	Presentation	Конвертиране, компресиране и криптиране на данни.	TLS/SSL
5	Session	Създаването, поддържането и терминирането на сесии. Сигурност. Логически портове.	Sockets
4	Transport	Грижи се за целостта на съобщенията, за пристигането им в точна последователност, потвърждаване за пристигане, проверка за загуби и дублиращи се съобщения.	TCP, UDP
3	Network	Управлява на пакетите в мрежата. Рутиране. Фрагментация на данните. Логически адреси.	IPv4, IPv6, IPX, ICMP
2	Data Link	Предаване на фреймове от един възел на друг. Управление на последователността на фреймовете. Потвърждения. Проверка за грешки. MAC.	ATM, X.25, DSL, IEEE 802.11
1	Physical	Отговаря за предаването и приемането на неструктурирани потоци от данни по физическият носител. Кодиране/декодиране на данните. Свързване на физическият носител.	IEEE 802.11, IEEE 1394, Bluetooth

HTTP
Характеристики на HTTP

- **Приложен протокол** – като транспортен протокол, почти винаги се ползва TCP/IP, в редки случай и UDP. По подразбиране слуша на порт 80.
- **Модел „Заявка-Отговор“ (“Request-Response”) -** служи за комуникационен канал в „Клиент-Сървър“ архитектура, като следва строги правила за ред и формат на съобщенията между участниците.
- **Не пази състояние (Stateless)** – всяка клиентска заявка е независима сама по себе си. Сървърът не обвързва логически серия заявки от определен клиент. Това води до липса на вграден в протокола механизъм за поддържане на сесии.
- **HTTP Транзакция** - /опростен модел – без преизползване на конекцията/
  1. Клиентът отваря комуникационен канал (TCP сокет)
  2. Изпращане на заявка от клиента към сървъра
  3. Сървърът връща отговор на клиента
  4. Затваряне на сокет-а от сървъра.

HTTP
Видове HTTP съобщения

- **Заявка** – инициатор е клиентът – подава информация на сървъра, достъп до кой ресурс иска да получи и каква операция иска да извърши с него (и евентуални входни параметри). **Клиент** (условно наречен User-Agent в HTTP) може да бъде всяко софтуерно приложение, спазващо правилата на протокола на комуникация.
- **Отговор** – изпраща се от уеб сървъра, като резултат от изпълнението на клиентска заявка. Под **уеб сървър** разбираме софтуерно приложение, служещо като доставчик на дадени услуги върху определени негови ресурси.

HTTP
HTTP Заявка

- **Начален ред**

  HTTP Метод (Глагол) – указва типът операция, която клиентът иска да извърши със заявеният ресурс.

  URL – уникален локатор на заявеният ресурс

  Версия на HTTP – версията на протокола, която ще се позлва за комуникация

  GET en.wikipedia.org/w/index.php HTTP/1.1

- **Хедъри** - опционални (HTTP 1.1 задължава специфицирането на хедър HOST в заявката). Възможно е да дефинира множество хедъри, като всеки от тях заема точно един ред и следва форматът: “Име на хедър: Стойност на хедър”

  Connection: Keep-Alive

  Host: en.wikipedia.org

- **Данни (Тяло)** – опционални, може да съдържат множество редове, включително и празни

HTTP
HTTP Заявка / Пример

```
GET /w/index.php?search=Students&title=Special%3ASearch HTTP/1.1
```

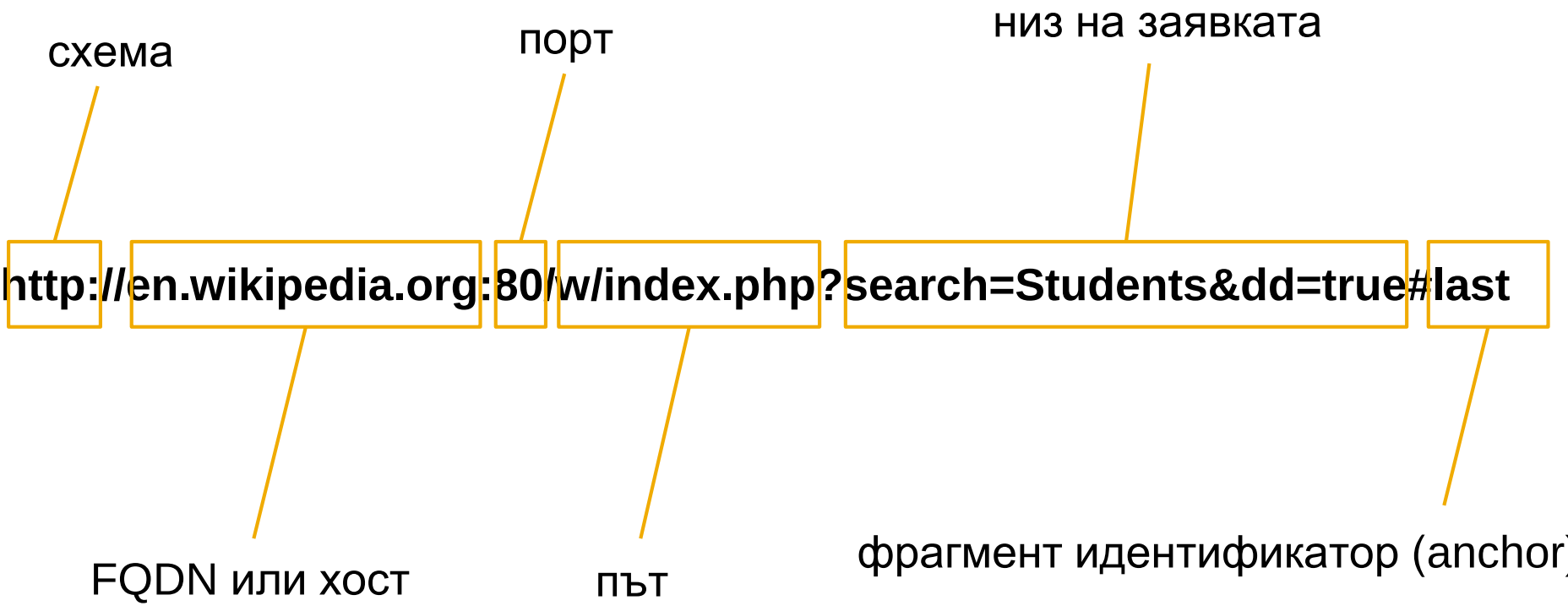
```
Accept: application/x-ms-application, image/jpeg,
application/xaml+xml, image/gif, image/pjpeg, application/x-ms-
xbap, */*
Referer: http://en.wikipedia.org/wiki/Main_Page
Accept-Language: en-US
User-Agent: Mozilla/4.0
Accept-Encoding: gzip, deflate
Host: en.wikipedia.org
Connection: Keep-Alive
```

```
<Празен ред>
```

```
<GET заявките нямат тяло!>
```

HTTP
HTTP Ресурси

Унифициран локатор на ресурси (URL) - стандартизиран адрес на даден мрежов ресурс (документ или страница). Всяка уеб страница е идентифицирана уникално чрез URL



HTTP
Видове HTTP методи

- **GET** – за зареждане на ресурс от сървъра
- **POST** - изпраща данни (например от HTML форма) за обработка от сървъра. Данните се съдържат в тялото на заявката
- **HEAD** - идентичен с GET, с разликата, че отговорът няма да върне тяло, а само хедъри
- **PUT** – ъплоудва специфичен ресурс
- **DELETE** – трие специфичен ресурс
- **TRACE** – указва на сървъра да върне низа на заявката в тялото на отговора
- **OPTIONS** – казва на сървъра да му върне всички позволени методи за даден ресурс
- **CONNECT** – за работа с проксита
- **PATCH** – за подмяна на части от ресурса

HTTP
HTTP Отговор

- **Начален ред –** съдържа 3 елемента, разделени с празно пространство помежду си:

  Версия на HTTP

  Статус код – обяснява резултата на изпълнието на заявката

  Причина – кратко обяснение на статус-кода

  HTTP/1.1 200 OK

- **Хедъри**

  Date: Sat, 06 Oct 2015 15:08:15 GMT

  Server: Apache

- **Данни (Тяло)** – отговорите обикновено връщат данни, като тук най-често се съдържа **HTML документът**, получен на базата на клиентската заявка.

HTTP
HTTP Отговор / Пример

```
HTTP/1.1 200 OK
```

```
Date: Sat, 17 Nov 2012 15:08:15 GMT
Server: Apache
X-Content-Type-Options: nosniff
Cache-Control: private, s-maxage=0, max-age=0, must-revalidate
Content-Language: en
Vary: Accept-Encoding,Cookie
Expires: Thu, 01 Jan 1970 00:00:00 GMT
Content-Encoding: gzip
Content-Length: 8582
```

```
<Празен ред>
```

```
<HTML>Пропускаме документа за простота!</HTML>
```

HTTP
HTTP Статус Кодове / 1

Трицифрени кодове, идентифициращи какъв е резултът от обработката на клиентската заявка

Групирани са в 5 категории, на базата на цифрата на стотиците

1. Група 100 – те са чисто информационни, не дават индикация дали заявката е била успешна или не. Служат за „временни“ кодове, т.е. заявката е пристигнала, но сървърът не е готов с резултата все още:

100 Continue
101 Switching protocols

2. Група 200 – сървърът е обработил успешно клиентската заявка

200 OK
206 Partial content

HTTP
HTTP Статус Кодове / 2

3. Група 300 – ресурсът е наличен, но е разположен на друго място

301 Moved permanently	**304 Not Modified**
307 Temporary redirect	

4. Група 400 - клиентска грешка

400 Bad Request	**404 Not Found**
401 Not Authorized	**408 Request Timeout**

5. Група 500 - сървърна грешка

500 Internal Server Error	**501 Not Implemented**
503 Service Unavailable	

HTTP
HTTP Хедъри / 1

- **Основни (General headers)** – могат да се ползват едновременно и в заявки, и в отговори. Съдържат информация (мета-данни) за самото съобщение или за метода на комуникация

```
Connection: keep-alive
Date: Sat, 17 Nov 2015 16:08:15 GMT
```

- **Заявка (Request headers)** – специфични са само за заявките и могат да съдържат данни за самата заявка или за клиента

```
Accept: text/html
Accept-Charset: utf-8
Accept-Language: en-US
User-Agent: Mozilla/4.0
```

HTTP
HTTP Хедъри / 2

- **Отговор (Response headers) -** съдържат информация (мета-данни) за сървъра и формата на съобщението

```
Server: Apache
Allow: GET, HEAD
```

- **Същински (Entity headers)** – информация за самото съдържание на данни (тяло) и/или за ресурса, заявен от клиента:

```
Content-Language: en
Content-Encoding: gzip
Content-Length: 8582
Last-Modified: Tue, 15 Nov 2012 12:45:26 GMT
```

HTTP
HTTP Хедъри / User Agent

User Agent е софтуер, който извършва действие от името на потребителя:

- E-mail клиенти.
- Web Browser-и.
- Месинджъри: Skype, WhatsApp.
- …

Примерен низ за Google Chrome Web Browser:

Mozilla/5.0 (Windows NT 6.1; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/38.0.2125.101 Safari/537.36

HTTP
Кеширане

- **Браузър кеш** - механизъм за временно съхранение на ресурси, с цел по-бързият им достъп.

- **Срок на годност (“expiration period”)** – целта е да се елиминират HTTP заявки, които биха получили еднакъв документ като отговор. За целта браузър кешът трябва да знае за колко дълго може да се “довери” на кешираният документ.

Cache-Control: max-age=3600

Expires: Wed, 21 Sep2017 16:00:00 GMT

- **Актуалност на данните (“validation”) –** сървърът предоставя възможност на клиента да провери дали кешираните му ресурси са били променяни.

Last-Modified: 01 Oct 2017 16:00:00

If-Modified-Since: 01 Oct 2017 16:00:00

HTTP
HTTP Сесии

Сесията е концепция, която позволява да се поддържа връзка (състояние) между 2 или повече http requests, изпратени към даден сървър в Internet.

HTTP
HTTP Сесии / Механизми

- **Бисквитки (Cookies)**

- **Hidden fields forms**

  HTML страницата трябва да съдържа скрита (hidden) форма:

```
<INPUT TYPE="HIDDEN" NAME="jsessionid" VALUE="...">
```

- **URL Rewriting (презаписване)**

Може да добавите в края на всяко URL данни, които да унифицират сесията, за да може сървъра да прочете тези данни и да намери вашата сесия.

```
http://<my_java_site>?jsessionid=I_am_unique_session_identifier
```

HTTP
Cookies

- **Cookies (Бисквитки)**

Cookie-тата са малки текстови файлове генерирани от сървъра и изпратени на клиента в header-ите.

Как работят бисквитките ☺

1. Клиентът изпраща request към сървъра.

2. Сървърът отговоря и в header-ите на response-а праща към клиента cookie-тата, които ще се ползват за проследяване на сесията

Примерен отговор (response) на apache tomcat web container-а съдържа header:

Set-Cookie: JSESSIONID=ACFF1B473DAB71CD27AA16049D61265E; Path=/SessionTest

3. Всеки следващ request, изпратен от клиента, трябва да съдържа Cookie header-а, за да може сървърът да намери сесията на клиента

Cookie: JSESSIONID=ACFF1B473DAB71CD27AA16049D61265E

HTTP
Cookies / Атрибути

Cookie-тата са дефинирани в RFC 2109.

Атрибути:

- **Comment** - обикновено се използва от програмистите, за да обсноват нуждата от ползването на cookie-то.
- **Domain** – определя домейна, за който cookie-то е валидно и ще бъде изпращано.
- **Max-age** – задава lifetime-а на cookie-то в секунди. След като изтече валидността на cookie-то, клиентът не трябва да го праща повече.
- **Path** – специфицира subset от URL-та, където cookie-то може да бъда изпращано.
- **Secure** – този атрибут указва, че cookie-то може да бъде трансферирано само по https протокола.
- **HttpOnly** – когато този атрибут е добавен, cookie-то не може да бъде четено или променяно от JavaScript
- **Version** – цяло число, което определя на коя версия на RFC-то отговоря cookie-то.

HTTP
HTTP2

- Защо е нужен?

- HTTP/2.0 или HTTP/2?

- Какви са разликите с HTTP/1.x?
  - двоичен;
  - напълно multiplexed;
  - паралелизъм само с една TCP връзка за всеки origin;
  - компресия на хедъри;
  - Разрешава “push” от сървъра обратно към клиента без предхождаща заявка.

- Кои браузери поддържат HTTP2 към момента (2017??
  - Firefox, Chrome. Opera, Safari, Internet Explored 11, Microsoft Edge.

Съдържание

Интернет и WWW

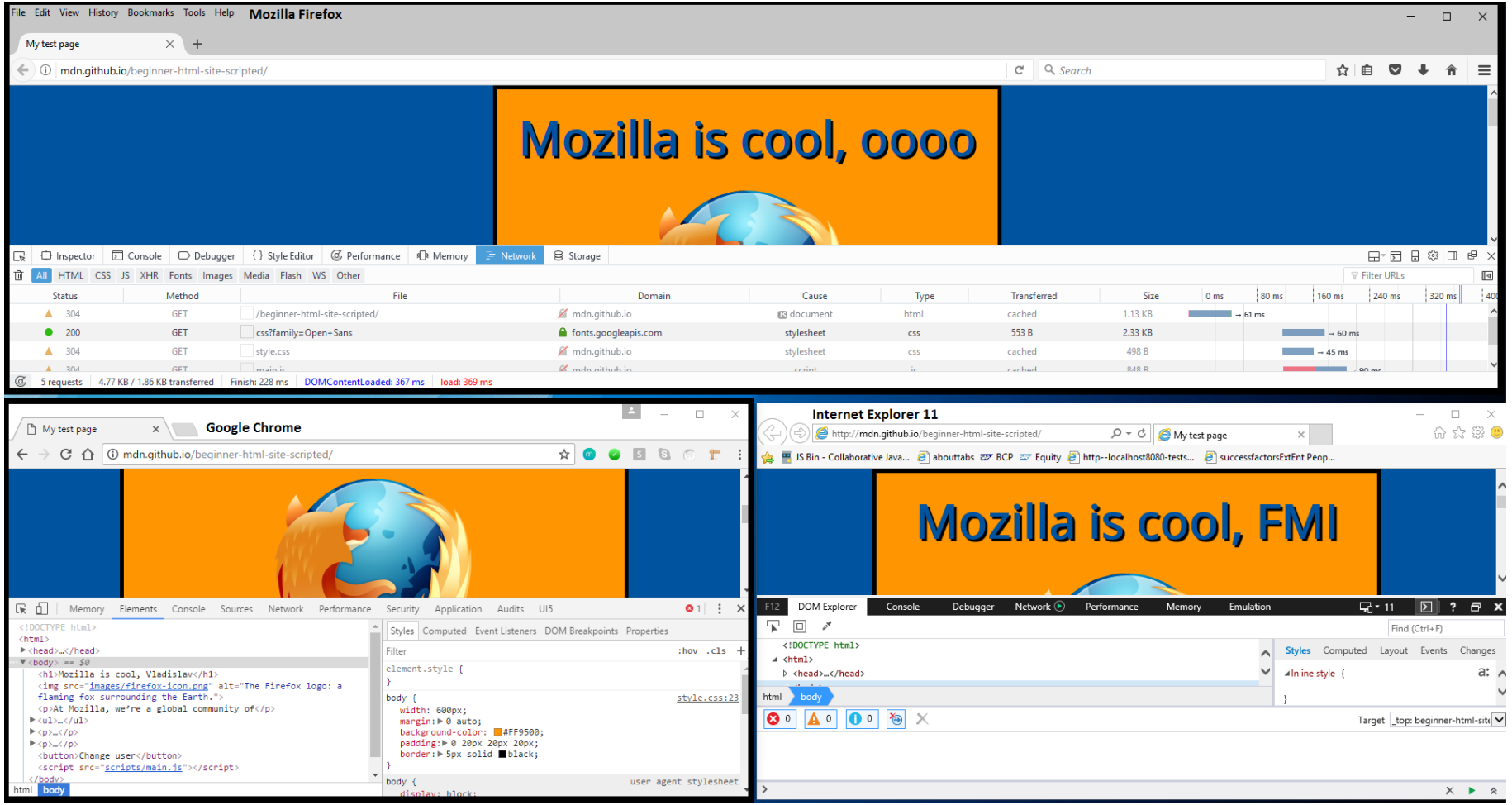
Моделът „клиент-сървър“

Web Browser

HTTP

Инструменти

Инструменти
В браузер-а



Инструменти
За разработка

Text Editors/IDE:

- Notepad++
- Sublime Text Editor
- Atom
- WebStorm (30 days trial | students license)
- Visual Code (free)

HTTP Servers:

- Node.js
- Apache

…

Благодаря за вниманието!

За контакти:
Владислав Илиев
e-mail: vladislav.Iliev@sap.com