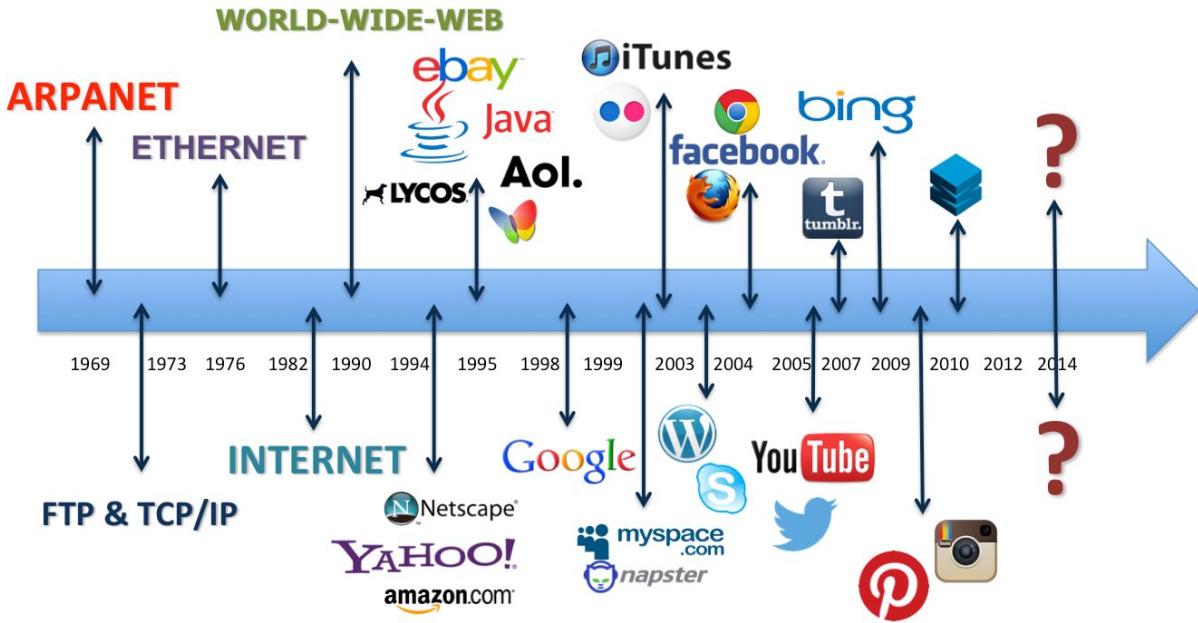




Interneto technologijos

Web sistemoms kurti

FMISB1611533



T1 - Žiniatinklis (WorldWideWeb/w3/web)



Nusikelkime atgal į 1980-ųjų pradžią: personalinių kompiuterių, kompiuterių tinklų ir pirmujų IT paslaugų tinkleose metą

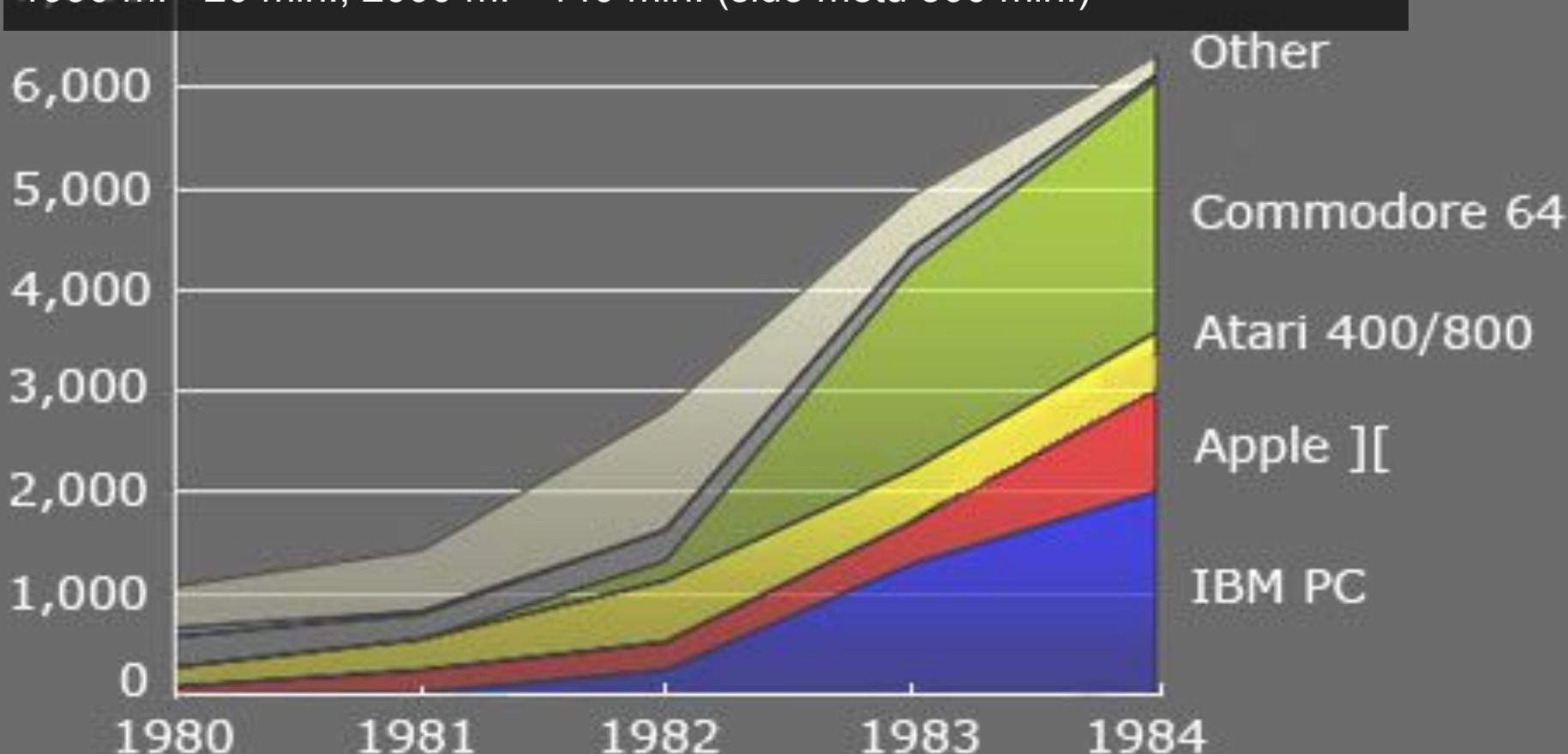


1973 - atsiranda pirmieji personaliniai kompiuteriai (PC): kaina, dydis ir greitis įgalina asmeninj naudojimą.

Personal Computer Sales (thousands of units)

1980-ųjų pradžioje jau parduodama daugiau nei milijonas PC.

1990 m. - 20 mln., 2000 m. - 140 mln. (šiuo metu 500 mln.)

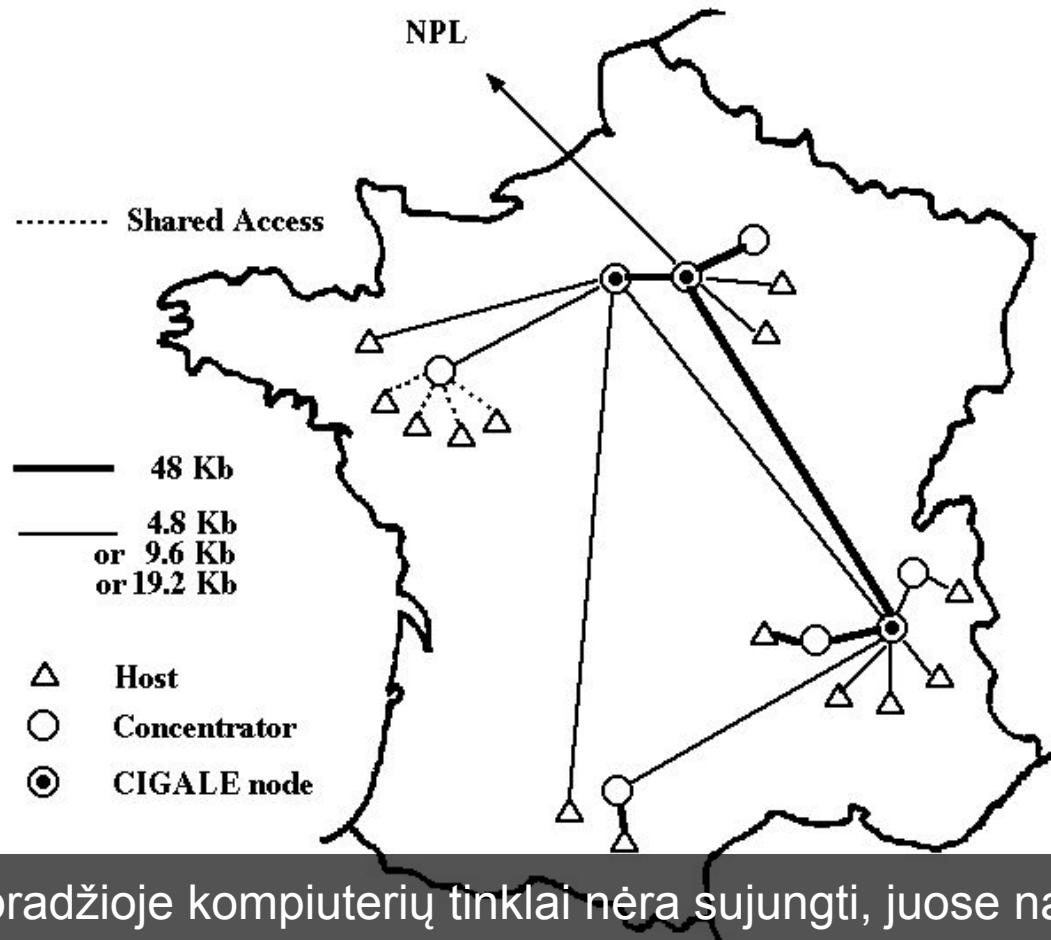




Tačiau PC savininkai neskuba jungtis prie tinklų. Jie mieliau diegiasi programas ir žaidimus iš tuo metu populiarų laikmenų



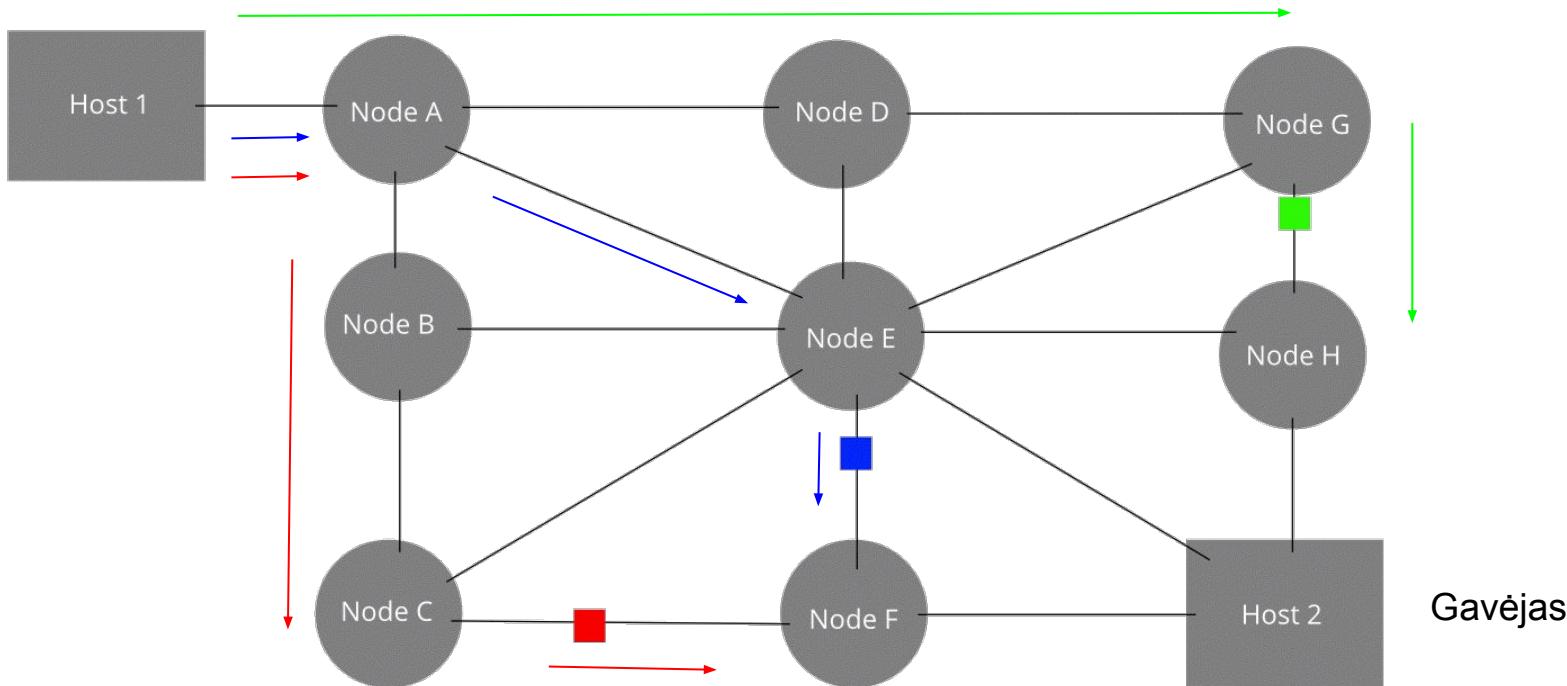
1983 pradėtos leisti CD-ROM laikmenos leidžia saugoti 550 MB informacijos
(tuometiniai PC arba neturėjo arba turėjo apie 5 MB dydžio kietuosius diskus)



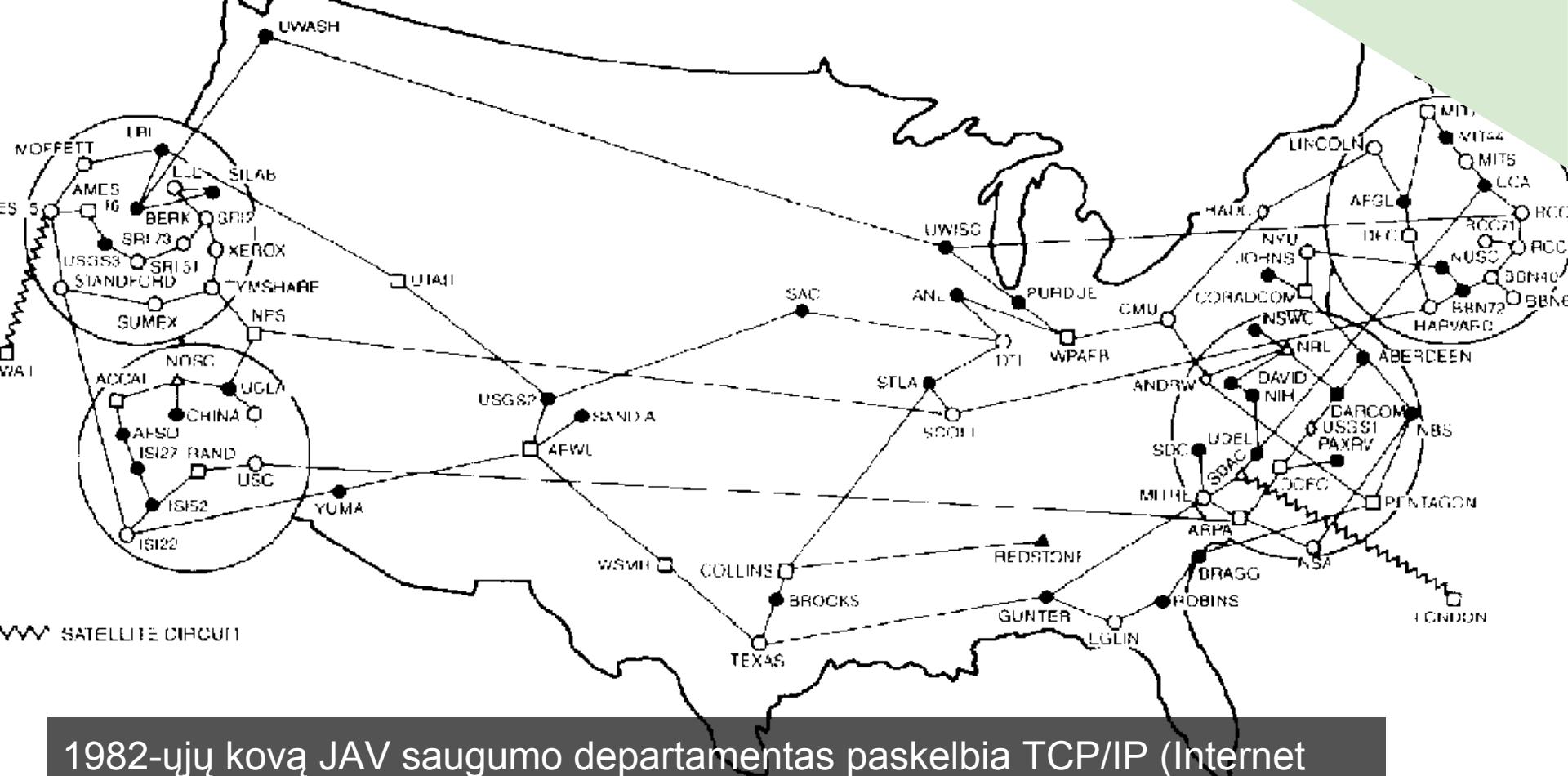
1980-ųjų pradžioje kompiuterių tinklai nėra sujungti, juose naudojami įvairiausi tinklo protokolai, techninė įranga, dauguma yra uždari

Tinklai naudoja paketu perėjimo tech. (packet switching) - žinutė suskaidoma ir siunčiama dalimis efektyviausiu maršrutu, o atkeliavusios dalys vėl sujungiamos

Siuntėjas



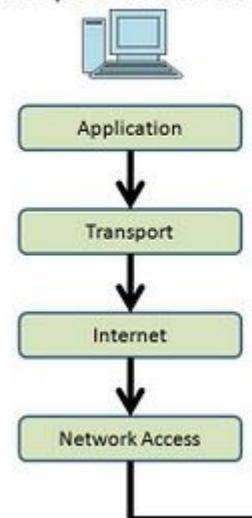
Gavėjas



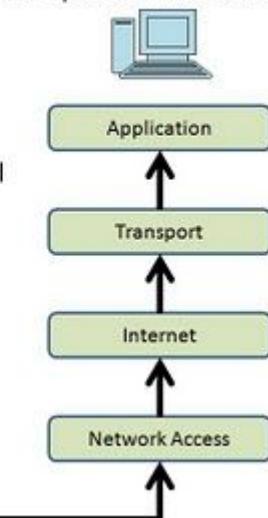
1982-ųjų kovą JAV saugumo departamentas paskelbia TCP/IP (Internet Protocol Suite) - tai interneto pradžia

TCP/IP modelis ir paketai (packets)

Computer A sends data.



Computer B receives data.



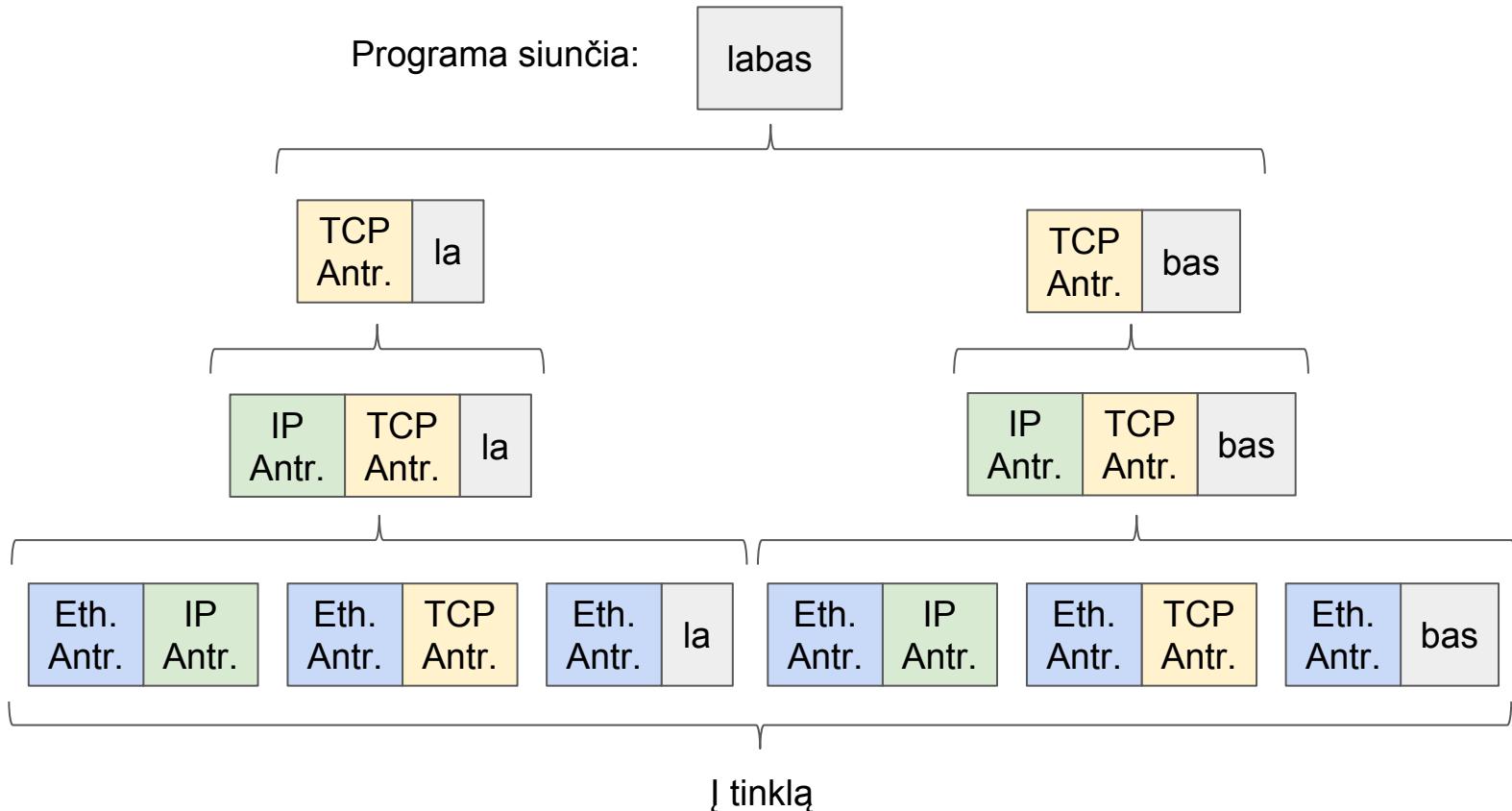
Siunčiami duomenys

TCP segments - patikimas pristatymas adresato programai

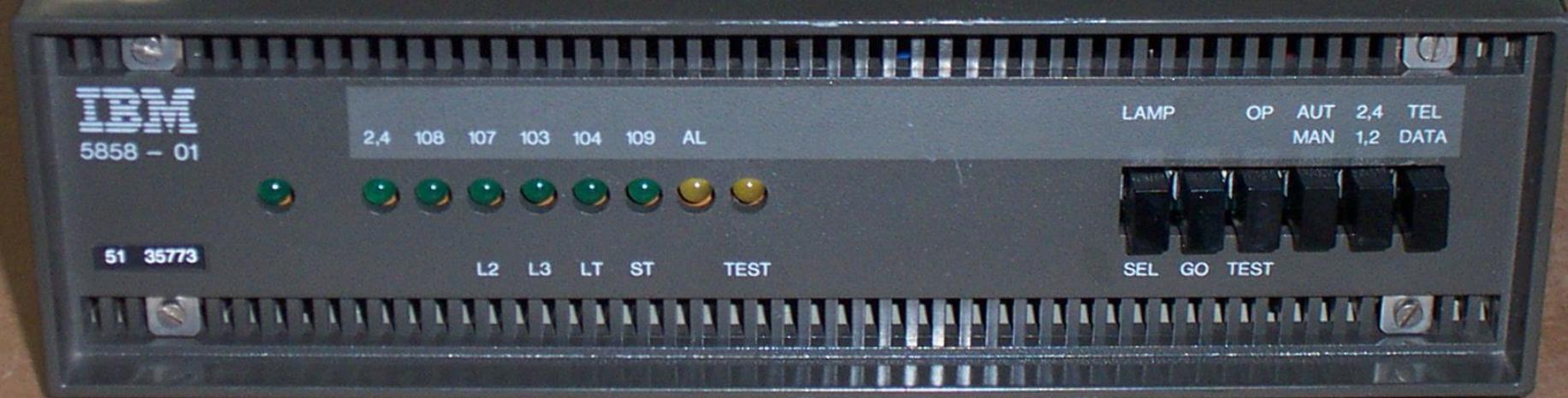
IP packets - pristatymas galutiniam adresatui

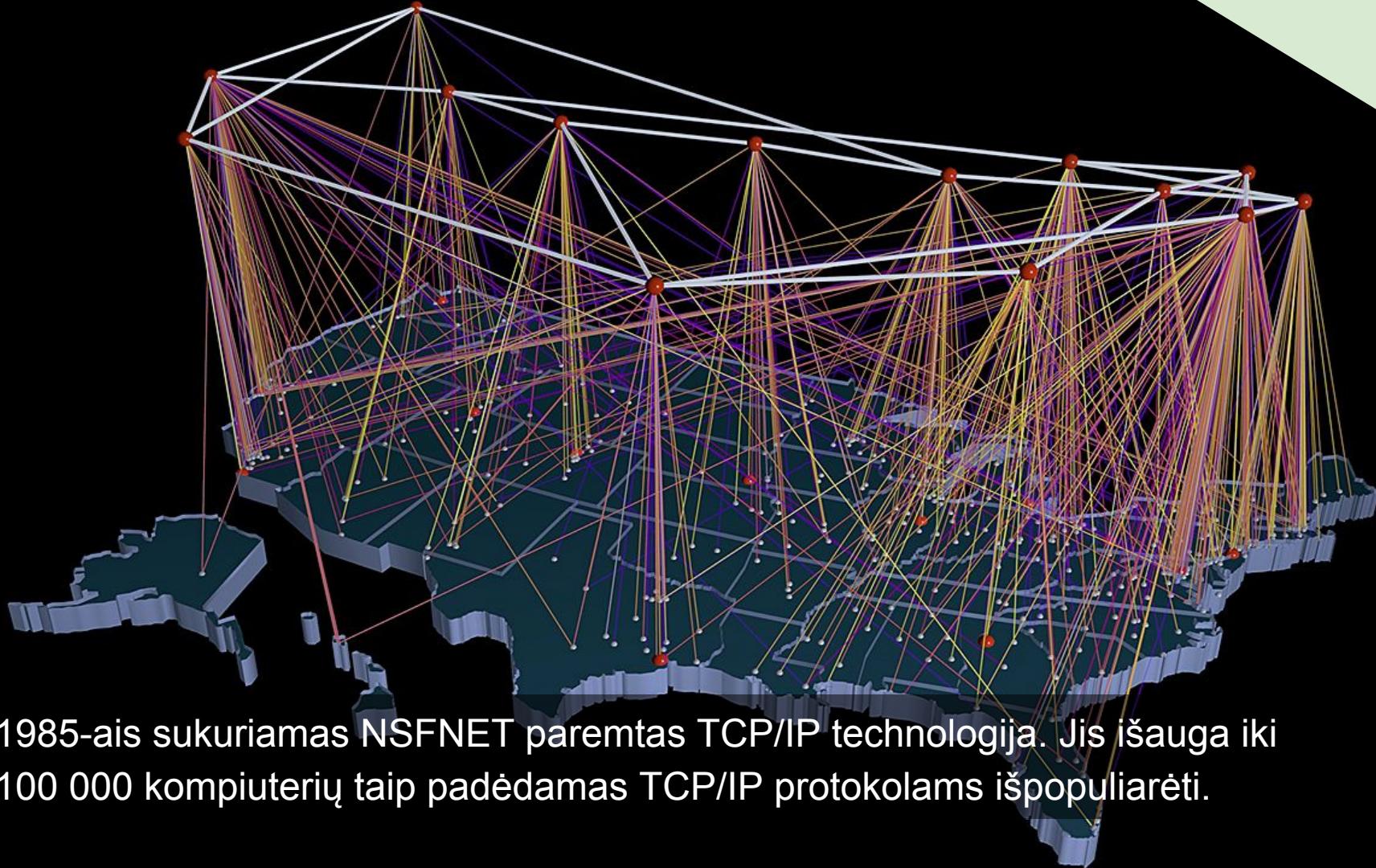
p.vz.: **Ethernet frames** - pristatymas tarpiniui "mazgui"

TCP/IP paketai (packets)

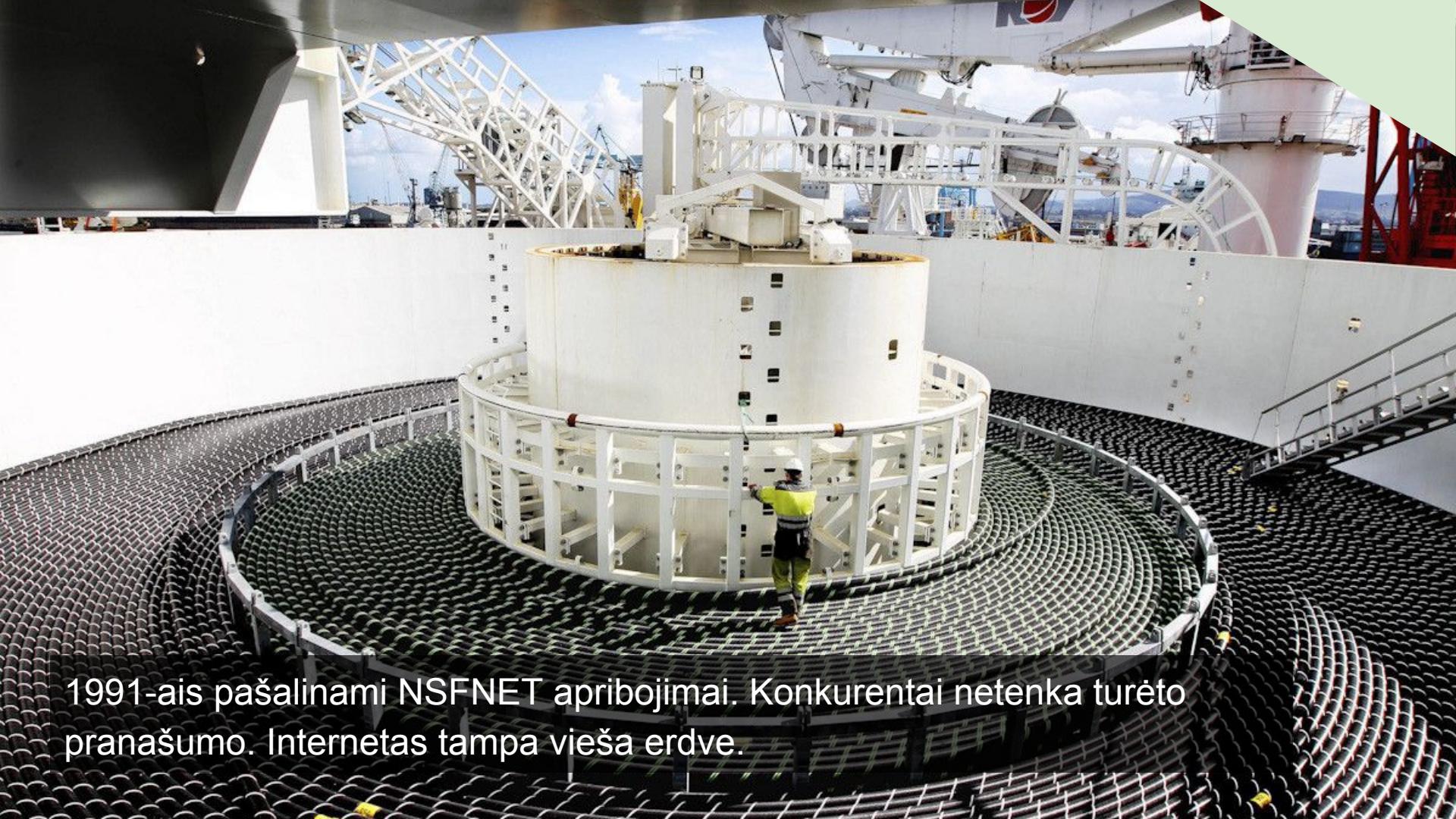


1984-ais publikuojamas OSI modelis. Tačiau susiduria su konkurencija:
DECNET (būsimas HP), SNA (IBM) ir APRANET (TCP/IP)

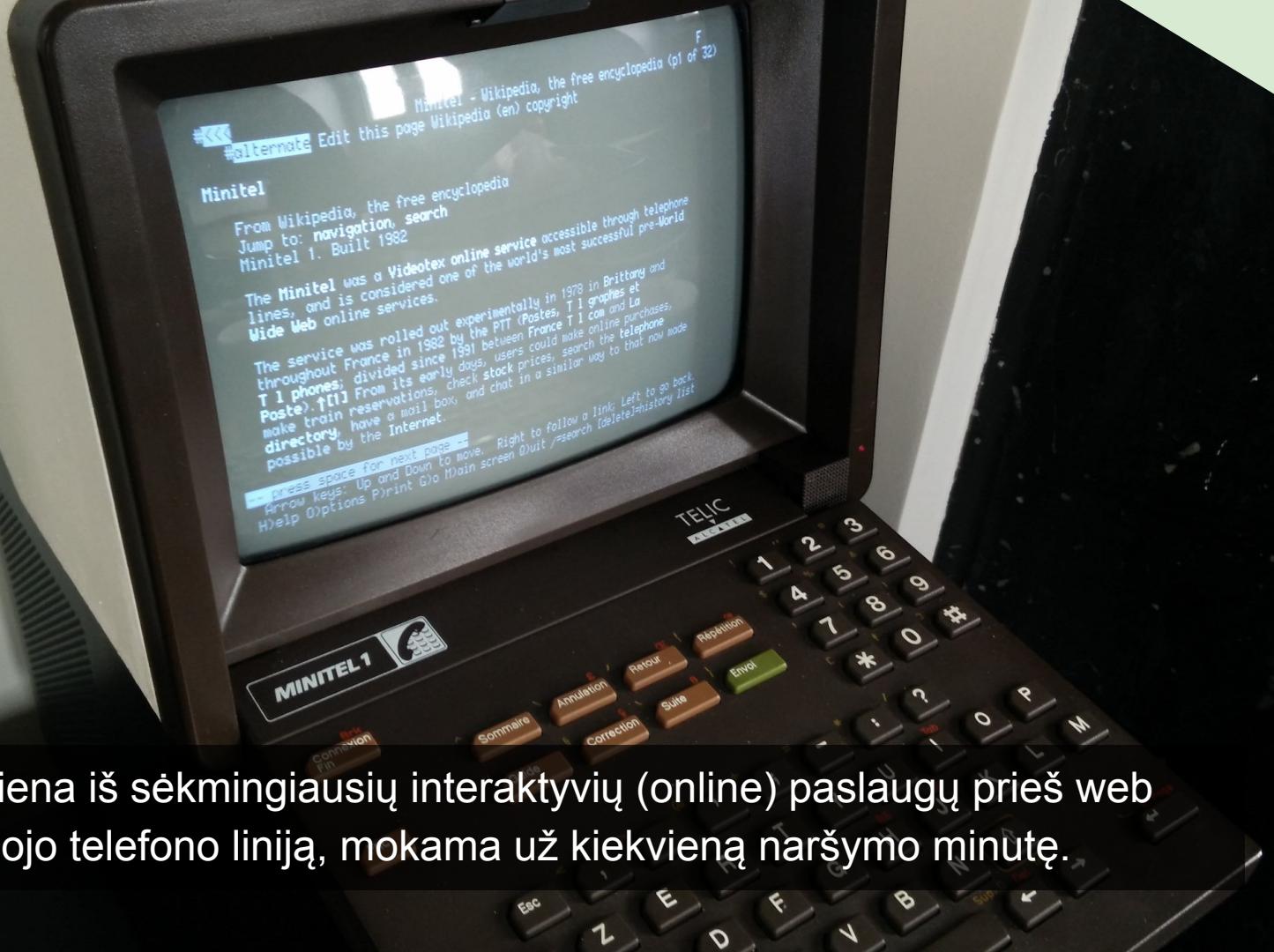




1985-ais sukuriamas NSFNET paremtas TCP/IP technologija. Jis išauga iki 100 000 kompiuterių taip padėdamas TCP/IP protokolams išpopuliarieti.



1991-ais pašalinami NSFNET apribojimai. Konkurentai netenka turėto pranašumo. Internetas tampa vieša erdve.



Minitel - viena iš sėkmingiausių interaktyvių (online) paslaugų prieš web erą. Naudojo telefono liniją, mokama už kiekvieną naršymo minutę.

->

| | | |
|----|-----|-----------------|
| 1 | 7 | comments |
| 2 | 4 | fa.apollo |
| 3 | 49 | fa.arms-d |
| 4 | 3 | fa.arpa-bboard |
| 5 | 2 | fa.dungeon |
| 6 | 7 | fa.energy |
| 7 | 16 | fa.human-nets |
| 8 | 31 | fa.info-cpm |
| 9 | 4 | fa.info-micro |
| 10 | 13 | fa.info-terms |
| 11 | 23 | fa.sf-lovers |
| 12 | 2 | fa.test |
| 13 | 33 | fa.unix-wizards |
| 14 | 27 | general |
| 15 | 89 | gripes |
| 16 | 383 | hacknews |
| 17 | 3 | humour |

Usenet - leidžia skaityti ir rašyti žinutes įvairiomis temomis (newsgroups) - laikomas šiuolaikinių interneto forumų protėviu



Seras Tim Berners-Lee yra Britų kompiuterių mokslininkas. Pabaigę studijas Oxfordo universitete, Berners-Lee dirba programuotoju CERN.

Vague but exciting ...

CERN DD/OC
Information Management: A Proposal

Tim Berners-Lee, CERN/DD
March 1989

Information Management: A Proposal

Abstract

This proposal concerns the management of general information about accelerators and experiments at CERN. It discusses the problems of loss of information about complex evolving systems and derives a solution based on a distributed hypertext system.

Keywords: Hypertext, Computer conferencing, Document retrieval, Information management, Project control

The diagram illustrates the proposed hypertext system architecture. At the center is a circle labeled "A Proposal X". Arrows point from various external sources to this central node, representing different types of information and their relationships:

- A dashed arrow labeled "for example" points from "Hyper Card" to "A Proposal X".
- A solid arrow labeled "includes" points from "Linked information" to "A Proposal X".
- A solid arrow labeled "describes" points from "ENQUIRE" to "A Proposal X".
- A solid arrow labeled "describes" points from "Computer conferencing" to "A Proposal X".
- A solid arrow labeled "describes" points from "VAX/ NOTES" to "A Proposal X".
- A solid arrow labeled "describes" points from "IBM GroupTalk" to "A Proposal X".
- A solid arrow labeled "describes" points from "Hierarchical systems" to "A Proposal X".
- A solid arrow labeled "describes" points from "CERNDOC" to "A Proposal X".
- A solid arrow labeled "includes" points from "This document" to "A Proposal X".
- A solid arrow labeled "refers to" points from "Hyper text" to "A Proposal X".
- A solid arrow labeled "includes" points from "Hyper text" to "This document".

At the bottom of the diagram, there is a legend with the following entries:

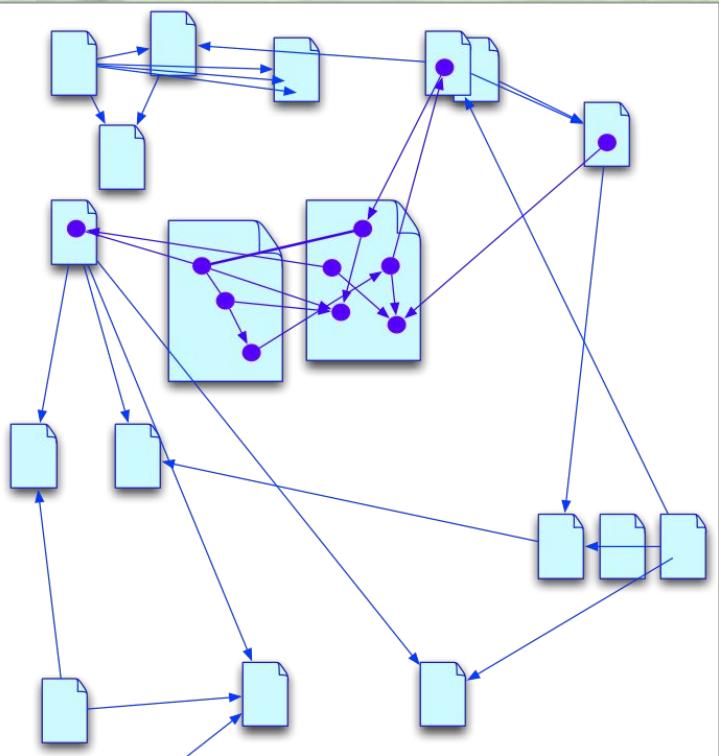
- MIS
- OC group
- RA section

1989-ų kovą, Timas išdėsto savo informacijos valdymo viziją dokumente
“Information Management: A Proposal”





“(...) norėdamas prieiti prie informacijos turėjai prisijungti prie kompiuterio. Taip pat kartais turėjai išmokti naudotis nauja programa (...) Dažnai tiesiog buvo paprasčiau pakalbėti su autoriumi per jo kavos pertraukėlę...”

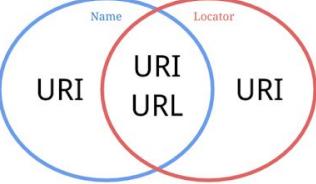


(...) užrašų "voratinklis" su nuorodomis (references) tarp jų yra daug naudingiau nei fiksuota hierarchinė sistema



```
<!DOCTYPE html>
<html>
<!-- created 2010-01-01 -->
<head>
<title>sample</title>
</head>
<body>
<p>Voluptatem accusantium  
totam rem aperiam.</p>
</body>
</html>
```

HTML



http://

Iki 1990-ų spalio, Timas sukuria tris fundamentalias technologijas, kurios išlieka šiuolaikinio žiniatinklio (web) pagrindu



WorldWideWeb

Info

HyperMedia Browser/Editor

An excercise in global information availability

by Tim Berners-Lee

Copyright 1990,91, CERN. Distribution restricted: ask for terms. TEST VERSION ONLY

HyperText: Text which is not constrained to be linear.

HyperMedia: Information which is not constrained linear... or to be text.

This version of the WWW application can pick up hypertext information from files in a number of formats, from local files, from remote files using NFS or anonymous FTP, from hypertext servers by name or keyword search, and from internet news. Hypertext files may be edited, and links made from hypertext files to other files or any other information.

For more help, use "Help" from the menu. If that doesn't work, then your application has been incompletely installed.

If you have any comments or have bugs, please mail timbl@info.cern.ch quoting the version number (above).

Style editor

Style name: List

<< style of selection >>

Apply style to selection

Apply style to all visible text

Save as...

Find unstyled text

Format

HotWired: What's New

Welcome to Yorb . the electric neighborhood hits Wired, 24 May. via In On The R

Flux Ned hears rumors of a deal between GN

Fetish Solar

Net Surf Pro

Jenny Holzer's truisms take a turn for the W

Links

Mark all A

Mark selection M

Link to marked L

Link to file...

Link to New N

Follow link

Unlink Z

Help

1990 metais sukuriama pirmoji web naršyklė "WorldWideWeb" ir serveris "httpd". Gimė "žiniatinklis".

JFG Home

JFG's home page at InfoDesign

This home page demonstrates some simple concepts of the World-Wide Web infrastructure for global information sharing.

Today, I showed Tim's original Web browser to Marc Weber. I just took some notes about him and linked them to his name on the fly.

I use this World-Wide Web editor exactly like any word processor, with the power to create links from sensitive pieces of text to other places, for instance to personal notes or to any information source in the world.

WWW research stuff

- Hypertext Resources (IN-Box)
- Geographical list of WWW servers (at C same list. Example : CERN phone book)
- WWW project documentation (at CERN)

Weber -- /Hypertext

A computer consultant, small publisher and freelance journalist who is currently investigating the early history of the Web, to be published in Wired. He loves HotWired.

World Wide Web

The WorldWideWeb (W3) is a wide-area [hypermedia](#) information retrieval initiative aiming to give universal access to a large universe of documents.

Everything there is online about W3 is linked directly or indirectly to this document, including an [executive summary](#) of the project, [Mailing lists](#) , [Policy](#) , November's [W3 news](#) , [Frequently Asked Questions](#) .

What's out there?

Pointers to the world's online information, [subjects](#) , [W3 servers](#), etc.

Help

on the browser you are using

Software Products

A list of W3 project components and their current state. (e.g. [Line Mode](#) ,[X11](#) [Viola](#) , [NeXTStep](#) , [Servers](#) , [Tools](#) , [Mail robot](#) , [Library](#))

Technical

Details of protocols, formats, program internals etc

Bibliography

Paper documentation on W3 and references.

People

A list of some people involved in the project.

History

1991 rugpjūčio 6-ą dieną yra paskelbiamas pirmasis web saitas pasaulyje

adresu <http://info.cern.ch> (veikia iki šių dienų)

Getting code

Getting the code by [anonymous FTP](#) , etc.

[Return to the main menu for gopher.floodgap.com:70](#)

[Floodgap Systems gopher root](#)

```
.o88o.          o8o          o8o
888 `''         'YP          'YP
o888oo  oooo  oooo  ooo. .oo.  '  ooo. .oo.  '
 888  '888  '888  '888P"Y88b  '888P"Y88b
 888  888  888  888  888  888  888
 888  888  888  888  888  888  888
o888o  'V88V"V8P' o888o o888o  o888o o888o

.oooooooooo  .oooo.  ooo. .oo.  .ooo.  .oooo.  .oooo.o
888` 88b  'P )88b  '888P"Y88bP"Y88b  d88` 88b d88(  "8
888  888  .oP"888  888  888  888  888oooo888  'Y88b.
`88bod8P'  d8( 888  888  888  888  888  .o o. )88b
`8ooooooo.  'Y888""8o o888o o888o o888o 'Y8bod8P' 8""888P'
d"      YD
"Y88888P'  @ Floodgap.com
```

Don't you think you need a little diversion? Here are some not-so-useful gopher resources for when you need to relax.

[Figlet Gateway](#)

Design your own ASCII text art! Utilizes the open-source tool figlet.

[Twitpher Twitter->Gopher Interface](#)

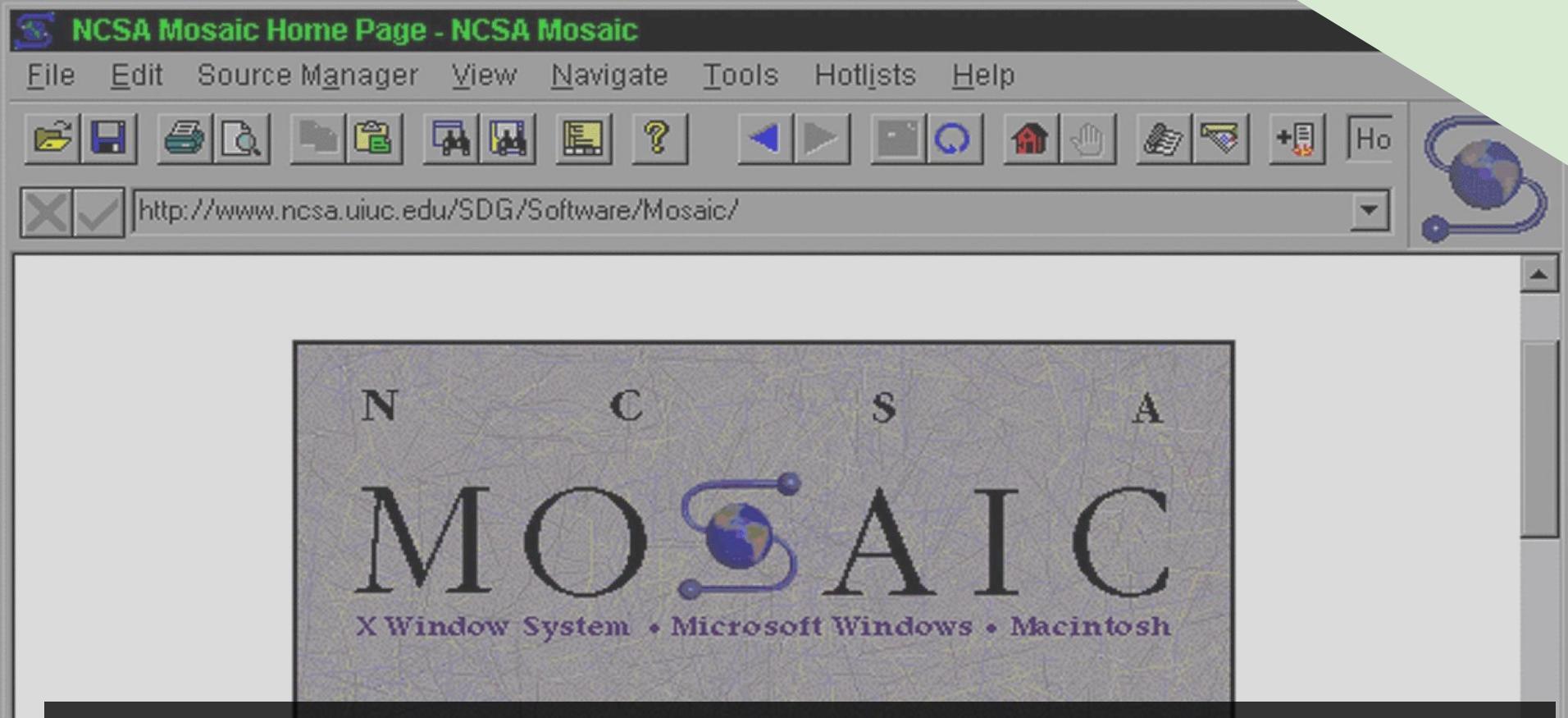
Forge: <http://twitpher.com/> Go!

[Current phase of the moon](#)

[Thoughts on the future of the Internet](#)

1990-ųjų pradžioje "Gopher", protokolas organizuojantis duomenis medžio struktūra (failai ir folderiai), yra pagrindinis web konkurentas

[Return to the main menu for gopher.floodgap.com:70](#)



1993 Minesotos universitetas, kuriam priklauso Gopher technologija, ją apmokestina, o CERN, kuriam priklauso Web technologija, atvirkščiai ją padaro nemokama

NCSA Mosaic was developed at the [National Center for Supercomputing Applications](#) at the University of Illinois in Urbana-Champaign. NCSA Mosaic software is copyrighted by The

 Search Options

[Yellow Pages](#) - [People Search](#) - [City Maps](#) -- [News Headlines](#) - [Stock Quotes](#) - [Sports Scores](#)

- [Arts](#) - - [Humanities](#), [Photography](#), [Architecture](#), ...
- [Business and Economy \[Xtra!\]](#) - - [Directory](#), [Investments](#), [Classifieds](#), ...
- [Computers and Internet \[Xtra!\]](#) - - [Internet](#), [WWW](#), [Software](#), [Multimedia](#), ...
- [Education](#) - - [Universities](#), [K-12](#), [Courses](#), ...
- [Entertainment \[Xtra!\]](#) - - [TV](#), [Movies](#), [Music](#), [Magazines](#), ...
- [Government](#) - - [Politics \[Xtra!\]](#) , [Agencies](#), [Law](#), [Military](#), ...
- [Health \[Xtra!\]](#) - - [Medicine](#), [Drugs](#), [Diseases](#), [Fitness](#), ...
- [News \[Xtra!\]](#) - - [World \[Xtra!\]](#) , [U.S.](#), [Current Events](#), ...
Iš viso pasaulyje yra daugiau nei 1 milijardas web saityų: 1992 metais buvo
- [Recreation and Sports \[Xtra!\]](#) - - [Sports](#), [Games](#), [Travel](#), [Autos](#), [Outdoors](#), ...
10, 1993: 130, 1994: 2738 (buvo sukurtas Yahoo!)
- [Reference](#) - - [Libraries](#), [Dictionaries](#), [Phone Numbers](#), ...
- [Regional](#) - - [Countries](#), [Regions](#), [U.S. States](#), ...

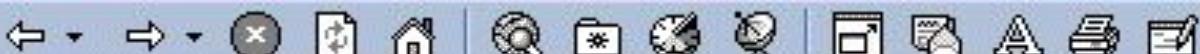
Les Horribles Cernettes



1992-ais metais į web įkeltas pirmasis paveikslukas vaizduojantis parodijomis užsiimančią merginų pop grupę



1994-ais Seras Tim Berners-Lee ikuria World Wide Web Consorciumą -
pagrindinę tarptautinę web standartų organizaciją



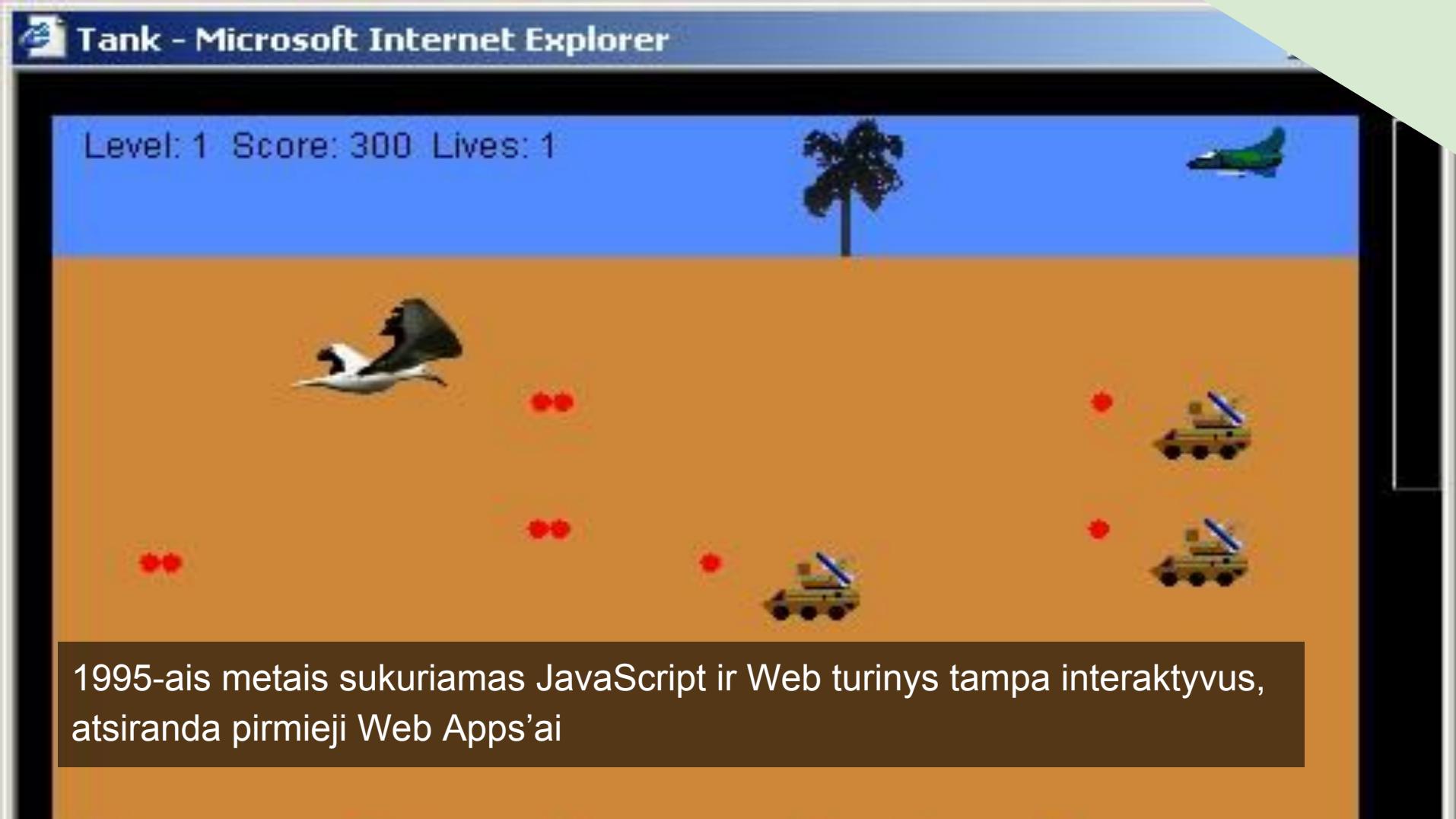
File

Address  http://members.home.net/dgreene/html/style.html

The Easy Way To Stylize Your Web Pages

1994-ais metais sukuriamas CSS - Web saitų stilius atskiriamas nuo struktūros

Finally, style separated from structure!



1995-ais metais sukuriamas JavaScript ir Web turinys tampa interaktyvus, atsiranda pirmieji Web Apps'ai

This clip had never been publicly shown before --- until, of course, it was publicly shown.



the cool thing about these guys
is that they have really

2005-ais metais į YouTube įkeltas pirmasis video. Video grotuvas sukurtas naudojant Macromedia (vėliau Adobe) Flash bei AJAX technologijomis.

Browser usage by region, March 2013

North America

1. 39%
2. 28%
3. 16%

Europe

1. 35%
2. 28%
3. 24%

Asia

1. 45%
2. 28%
3. 19%

South America

1. 59%
2. 20%
3. 18%

Africa

1. 39%
2. 37%
3. 18%

Oceania

1. 33%
2. 29%
3. 19%

Konkurencija tarp Web naršyklių, Web technologijas kuriančių bei paslaugas internetu teikiančių kompanijų skatina progresą



Rezultatas: iš apie 7 milijardų gyventojų kasdien naršo apie 3 milijardai
(apie 40% populiacijos)



Google

amazon

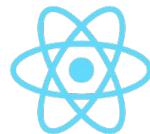
airbnb

LinkedIn

NETFLIX



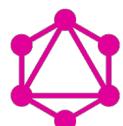
Google



React



Bootstrap



GraphQL

{less}



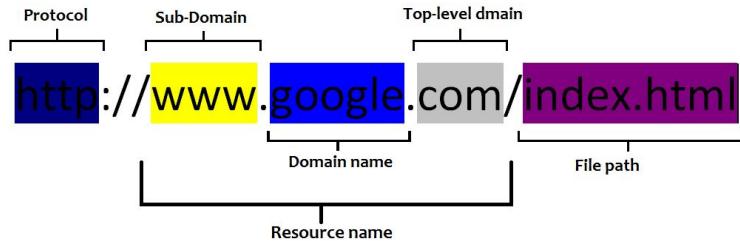
ANGULARJS



HTML



css



http://

JavaScript

HTML5

A vocabulary and associated APIs for HTML and XHTML

W3C Recommendation 28 October 2014

This Version:

<http://www.w3.org/TR/2014/REC-html5-20141028/>

Latest Published Version:

<http://www.w3.org/TR/html5/>

Latest Version of HTML:

<http://www.w3.org/TR/html/>

Latest Editor's Draft of HTML:

<http://www.w3.org/html/wg/drafts/html/master/>

Previous Version:

<http://www.w3.org/TR/2014/PR-html5-20140916/>

Previous Recommendation:

<http://www.w3.org/TR/1999/REC-html401-19991224/>

Editors:

WHATWG:

[Ian Hickson](#), Google, Inc.

W3C:

[Robin Berjon](#), W3C

[Steve Faulkner](#), The Paciello Group

[Travis Leithead](#), Microsoft Corporation

[Erika Doyle Navara](#), Microsoft Corporation

[Theresa O'Connor](#), Apple Inc.

[Silvia Pfeiffer](#)

[\[Docs\]](#) [\[txt|pdf\]](#) [\[draft-ietf-http...\]](#) [\[Tracker\]](#) [\[Diff1\]](#) [\[Diff2\]](#) [\[IPR\]](#) [\[Errata\]](#)

PROPOSED STANDARD

Errata Exist

Internet Engineering Task Force (IETF)
Request for Comments: 7540
Category: Standards Track
ISSN: 2070-1721

M. Belshe
BitGo
R. Peon
Google, Inc
M. Thomson, Ed.
Mozilla
May 2015

Hypertext Transfer Protocol Version 2 (HTTP/2)

Abstract

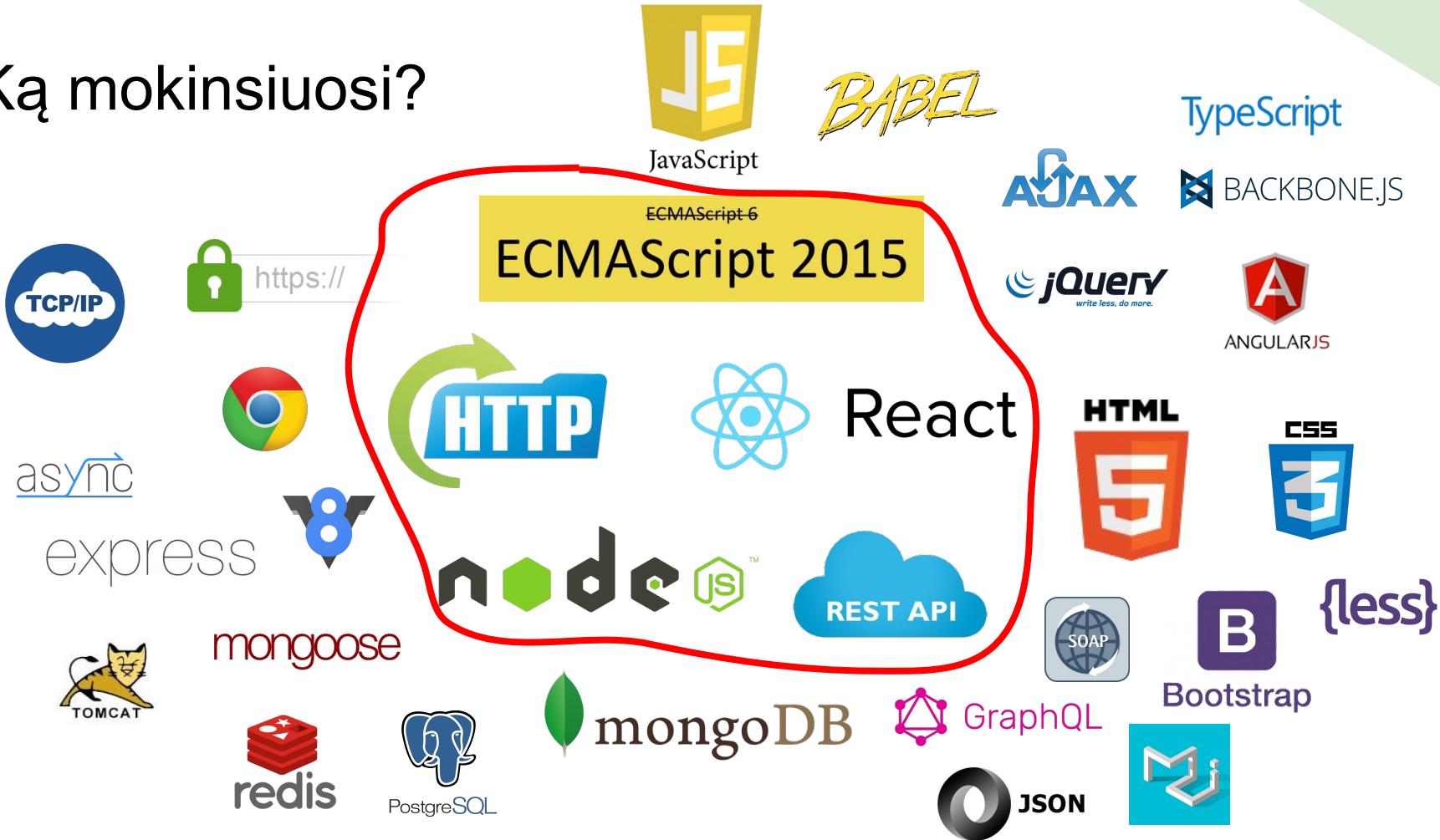
This specification describes an optimized expression of the semantics of the Hypertext Transfer Protocol (HTTP), referred to as HTTP version 2 (HTTP/2). HTTP/2 enables a more efficient use of network resources and a reduced perception of latency by introducing header field compression and allowing multiple concurrent exchanges on the

Web sistemos

- Sistemos, kurios naudoja interneto ir web technologijas paslaugoms ar informacijai teikti:
 - Internautams (naršyklė / app'sas -> sistema)
 - Kitoms sistemoms (sistema -> sistema)



Ką mokinsiuosi?

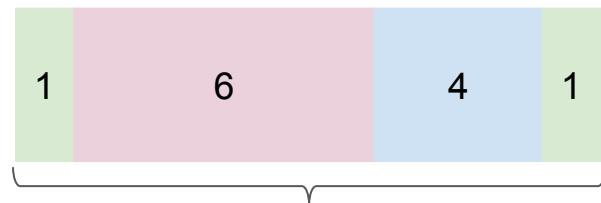


Mykolas Molis

- Mokėsi kompiuterių mokslą (Computer Science) bei programų sistemų inžineriją (Software Engineering)
- IT industrijoje darbuojuosi daugiau nei 12 metų:
 - ~60 000 darbuotojų turinčiame investiciniame banke (JAV, JK)
 - ~2000 darbuotojų turinčiame komerciniame banke (Lietuva)
 - ~700 darbuotojų turinčioje tarpautinėje IT paslaugų kompanijoje (JAV, Lietuva)
 - ~200 darbuotojų turinčiame valstybiniame registre (Lietuva)
 - ~50 iki 100 išaugusiame tarptautiniame startupe užsiimančiame saugumu (JAV, Prancūzija)
 - ~20 darbuotojų turinčioje IT kompanijoje užsiimančioje el. parašu (Lietuva)
 - 15 iki 40 išaugusioje tarpautinėje “Silicon Valley” stiliaus IT pasaugų kompanijoje (JK, Lietuva)
 - 3 iki 10 išaugusiame tarptautiniame startupe užsiimančiame web sistemų kokybe (JK, Lietuva)
 - 1 laisvai samdomas darbuotojas (freelancer) pastato saugumo sistemai kurti (Vokietija)
- ~90% sistemų, kurias vysčiau, buvo **Web sistemos**
- Vedantysis programuotojas (lead developer)

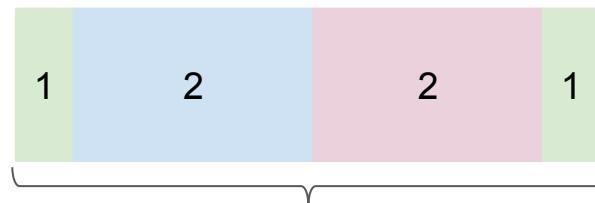
Kiek mokinsiuosi?

Teorija (T)



12
paskaitų

Praktika (P)



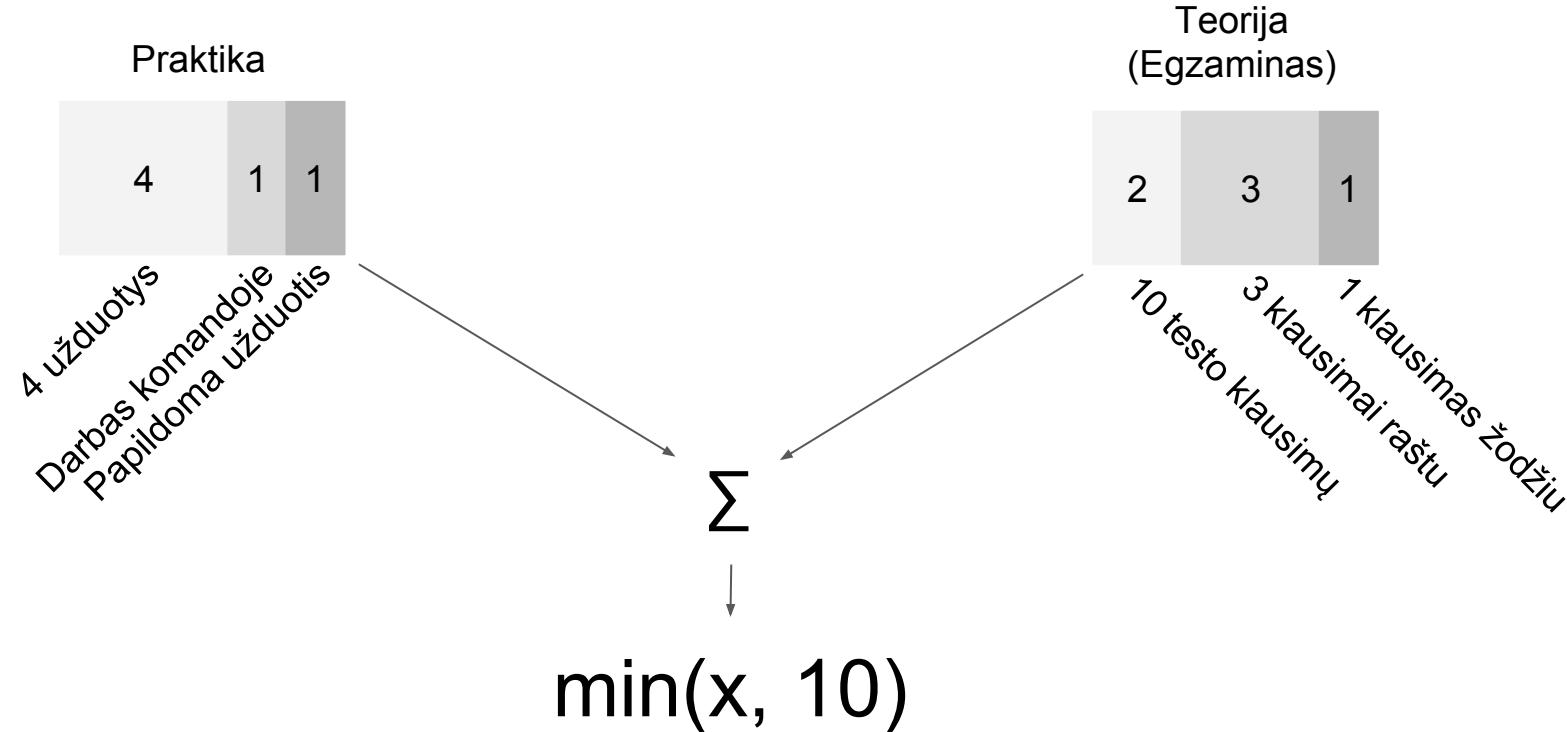
6
užsiėmimai

Bendram išsilavinimui

Reikia dėmesio

Reikia sukaupto dėmesio

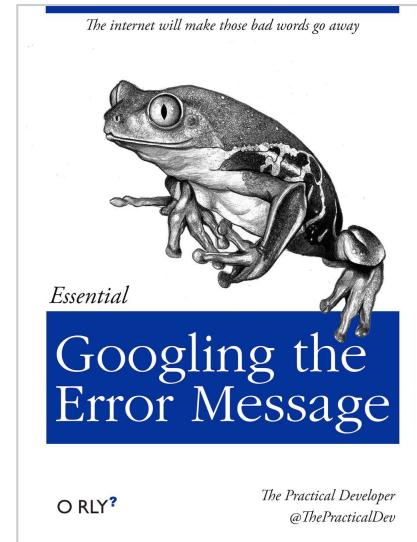
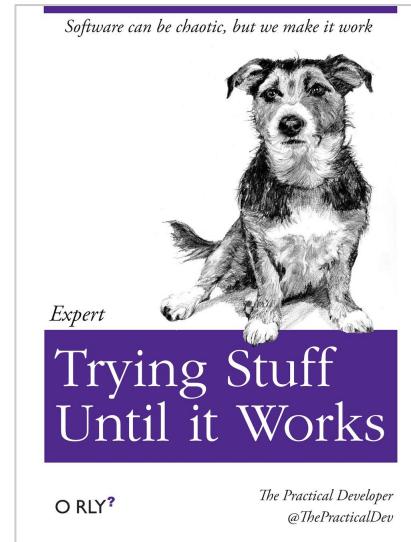
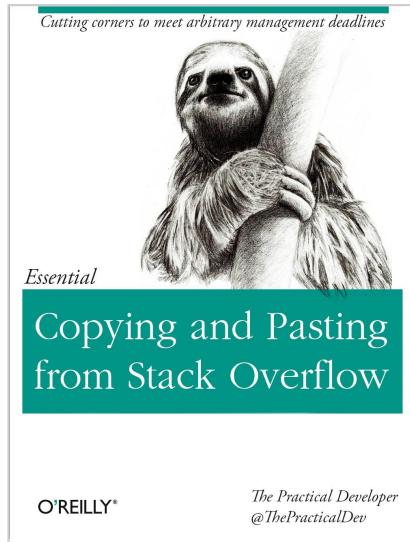
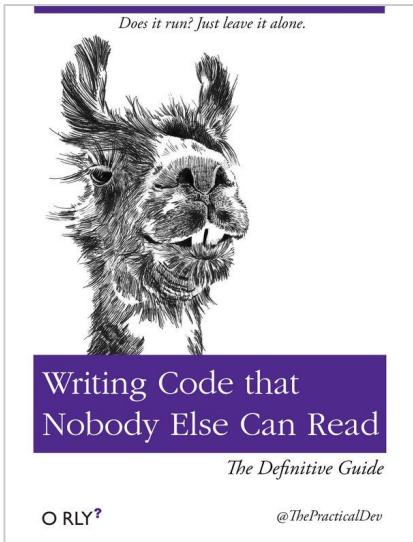
Kaip gerai mokinsiuosi?



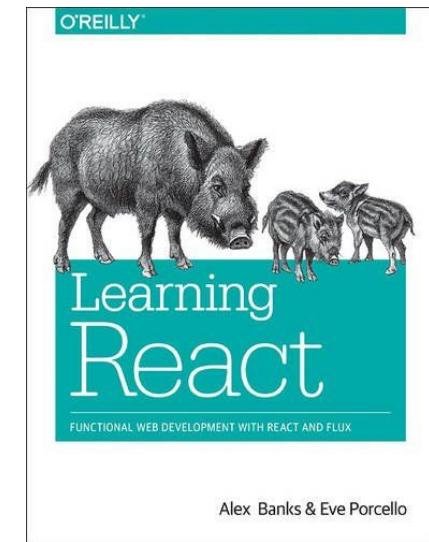
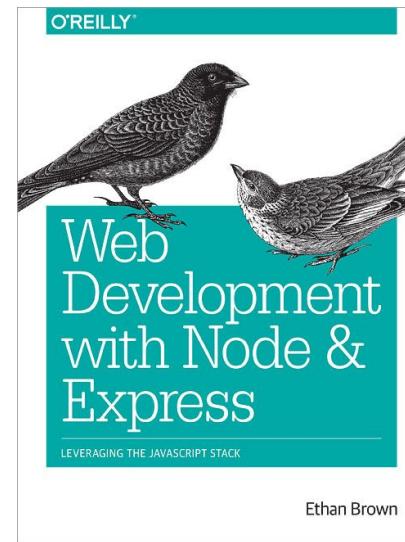
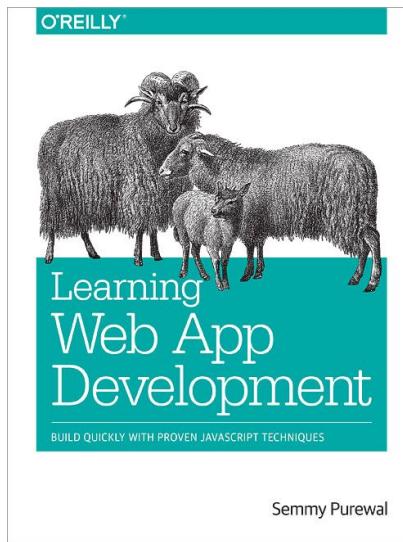
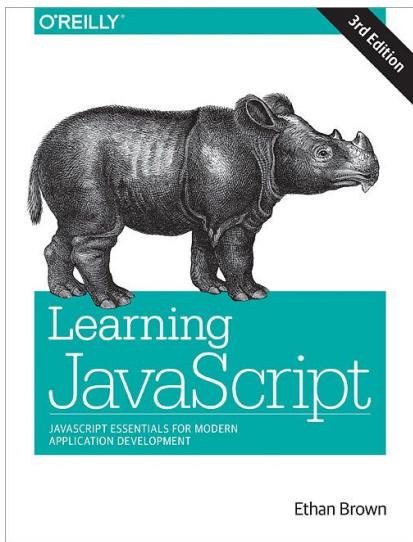
Praeitų metų kursas

- 48 studentai
 - 42 atėjo į egzaminą arba į perlaikymą ir jį išlaikė
 - 2 egzamine ir perlaikymuose nepasirodė
 - 4 perkélė studijas
- Tiems, kurie produktyviai išnaudojo praktikos laiką pavyko gauti bent 5 iš 6 balų (iš egzaminą éjo ji jau išlaikę)
- Nebuvo nei vieno, kuris atsakės į klausimus raštu gavo visus 3 balus
- Egzamine žodžiu tik VIENAM nepavyko gauti papildomų balų

Ką skaitysiu?



Ką skaitysiu?



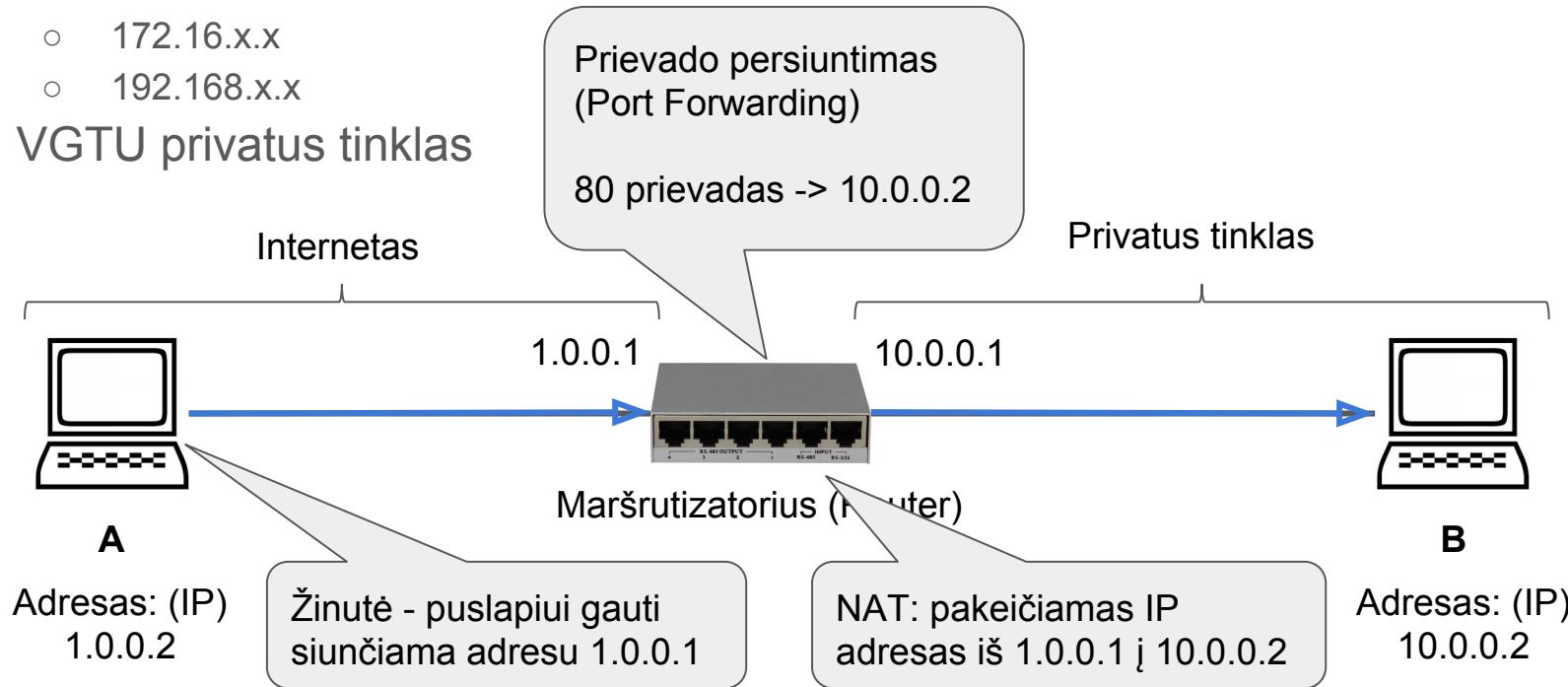
Santrauka Web sistemų kūrėjui



- Lygmenis palaiko Operacinę Sistema (pvz.: Linux, OS X, Windows)
- Ryšio lygmuo web sistemų kūrėjui dažiausiai nėra aktualus
- Interneto lygmenyje esantis IP adresas:
 - Naudotojui leidžia kreiptis į norimą adresatą internete (pvz.: naršyklė -> Google)
 - Web sistemai leidžia kreiptis į kitą norimą adresatą internete (pvz.: Google -> DB serverį)
- Transporto lygmenyje esantis prievadas:
 - Leidžia pasiekti įrenginyje esančia programą (pvz.: web sistemą, duomenų bazę)
 - Naudojant UDP protokolą galima greičiau siųsti srautinę informaciją (pvz.: tv, radiją)
 - Naudojant TCP protokolą galima siųsti informaciją patikimu būdu (pvz.: failus, dokumentus)

Santrauka Web sistemų kūrėjui - IP adresai

- Privataus tinklo IP adresas pasiekiamas tik tokio tinklo viduje:
 - 10.x.x.x
 - 172.16.x.x
 - 192.168.x.x
- VGTU privatus tinklas



Santrauka Web sistemų kūrėjui - prievedai (ports)

- Prievedo reikšmė apribota nuo 0 iki 65535 ($2^{16}-1$)
- Gerai žinomi prievedai (well-known ports) yra nuo 0 iki 1023. Jų naudoti savo kuriamam aplikacijos sluoksnio protokolui nepatartina.
 - 25 - Simple Mail Transfer Protocol (SMTP) - el. laiškams siųsti
 - 80 - Hypertext Transfer Protocol (HTTP) - web naršymui
 - 110 - Post Office Protocol Version 3 (POP3) - el. laiškams gauti
- Savo kuriamam aplikacijos sluoksnio protokolui patartina naudoti prievedus nuo 1024 iki 65535
 - 5432 - standartinis PostgreSQL duomenų bazės prievedas
 - 6881–6887 - dažnai naudojamas BitTorrent prievedų intervalas
 - 27017 - standartinis MongoDB prievedas



P1 - Atviro kodo ‘organizacija’ žiniatinklio
(WEB) sistemoms kurti

Atviro kodo ‘organizacija’

- ‘Organizacija’ yra tikros IT organizacijos simuliacija:
 - Komandinis darbas IT srityje (vietoje ir nuotoliniu būdu)
 - IT sistemos kūrimo proceso taikymas
 - IT sistemos projektavimo metodologijos taikymas
 - IT sistemos kūrimo įrankių ir pateikimo (serve) platformų taikymas
- Praktikos dalyviai tampa ‘organizacijos’ nariais
- Išrinkime ‘organizacijos’ pavadinimą

Atviro kodo ‘organizacijos’ ‘steigimas’

- ‘Organizacija’ ‘įregistruojama’ GitHub platformoje:
<https://github.com/organizations/new>
- Prisijunkite prie ‘organizacijos’ GitHub platformoje
- Sukuriama ‘organizacijos’ Slack komanda: <https://slack.com/create#email>
- Prisijunkite prie komandos Slack platformoje
 - Pasirinktinai galima atsisiu̇ti Slack Aplikaciją kompiuteriui ar telefonui
 - DĒMESIO! Naudojant Slack nemokamai yra ribojamas saugomų žinučių skaičius - kad neprarasti duomenų reiktų juos saugoti pvz GitHub platformoje

Atviro kodo ‘organizacijos’ veikla

- ‘Organizacija’ kuria atviro kodo web sistemas

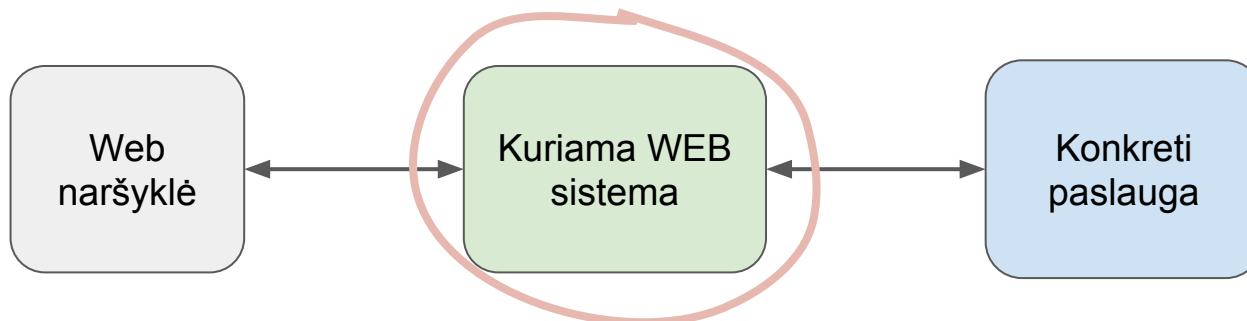
The image displays two side-by-side screenshots of web-based applications.

Left Screenshot: A trading platform interface. At the top, it shows a balance of +500 and €21,445.28 Available. Below this is a search bar labeled "Search our Instruments". The main area shows a table of instruments with columns for "Instrument", "Change", "Sell", and "Buy". Three rows are listed: EUR/GBP (-0.28 %), EUR/USD (-0.26 %), and EUR/AUD (-0.14 %). To the left of the main content is a sidebar with links: "Menu", "Trade" (selected), "Open positions", "Orders", "Closed positions", and "Real money". Below the sidebar is a chart for EUR/GBP, showing a downward trend from approximately 0.87658 to 0.87674. The chart has timeframes from "Tick" to "1W". A purple horizontal line marks the "Current sell rate". At the bottom, it says "Demo Account | mykolas.molis@gmail.com" and "Switch to real money".

Right Screenshot: A Google Photos album titled "Sat, 30 Dec 2017 Vilnius". The album contains 15 images showing people walking in a snowy forest. The interface includes standard Google Photos navigation elements like "Assistant", "Photos", "Albums", and "Sharing".

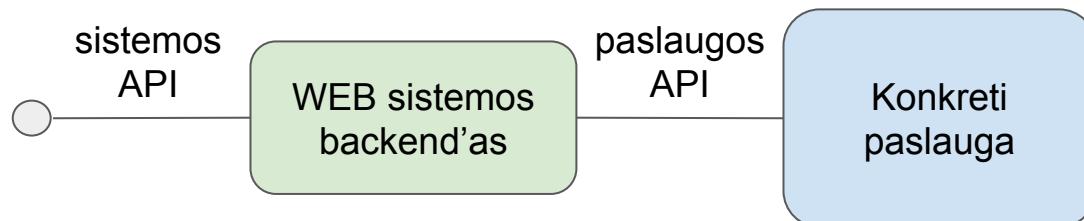
‘Organizacijos’ nario tikslas

- Kiekvienas ‘organizacijos’ narys:
 - pasirenka sritį ir tos srities **konkrečią paslaugą** iš sąrašo:
<https://github.com/toddmotto/public-apis>
 - išnagrinėja **konkrečios paslaugos** API (Application Programming Interface) ir aprašo API pagrindu kuriamą WEB sistemą (reikalavimai užduočiai pateikti kitoje skaidrėje)
 - iki semestro galo vysto WEB sistemą ir ją (dalį jos) atsiskaito iki paskutinio praktikos užsiémimo (dėl laiko stokos perlaikinėtojai galės atsiskaityti iki maks. 2 balų)



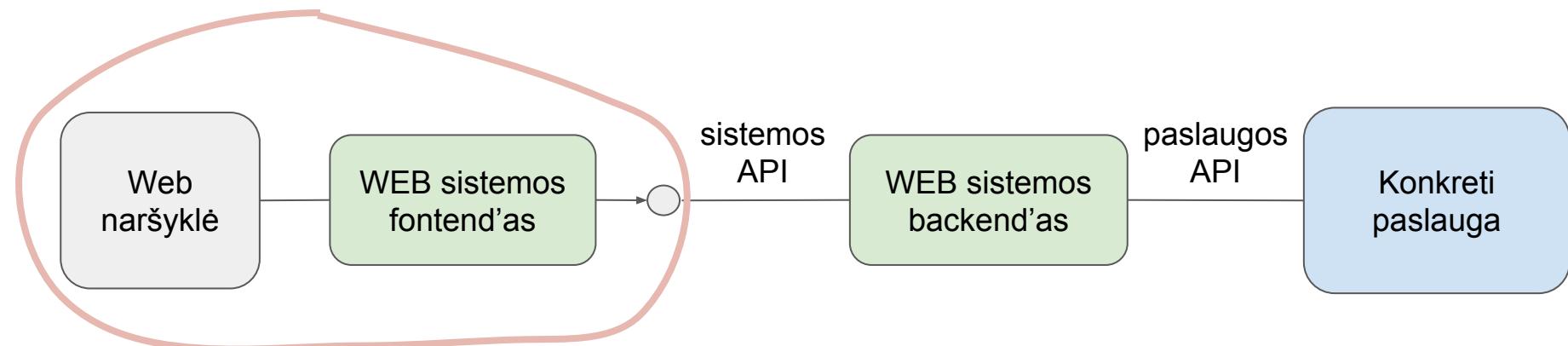
Reikalavimai WEB sistemos backend'ui

- WEB sistema turi turėti backend'ą, kuris per savo REST API (**sistemos API**) turi pateikti ir modifikuoti:
 - konkrečios paslaugos duomenis (per **paslaugos API**)
 - savo duomenų bazės duomenis (pradinė duomenų bazė gali būti masyvas (array))



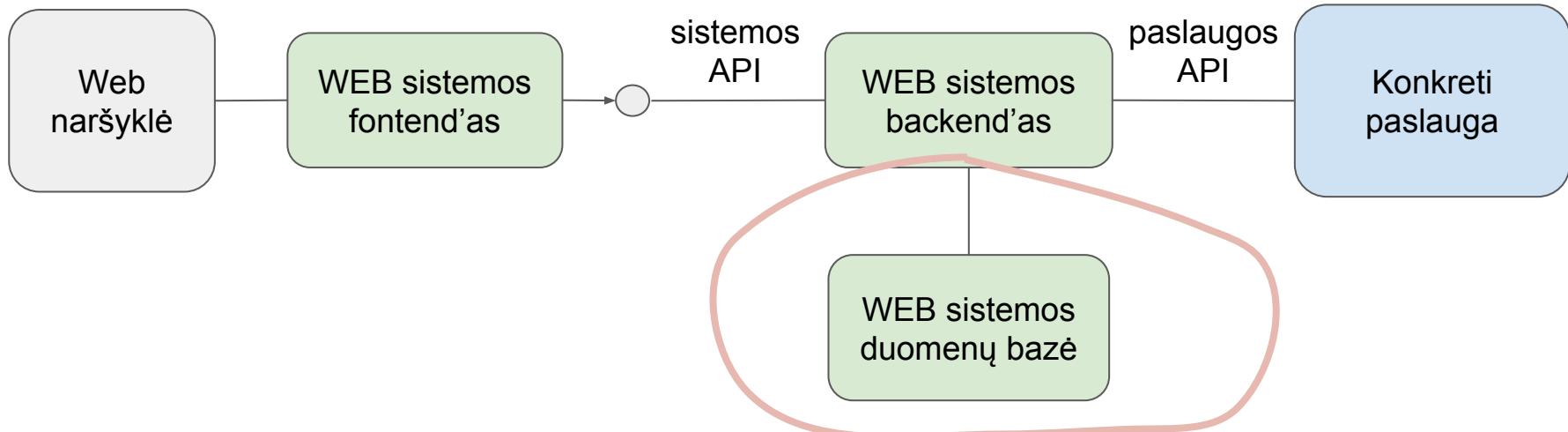
Reikalavimai WEB sistemos frontend'ui

- WEB sistema turi turėti frontend'ą, kuris naršyklėje:
 - pavaizduotų iš **sistemos API** gautus duomenis
 - susietų naudotojo veiksmus (pvz.: duomenų kūrimą ar trynimą) su **sistemos API** metodais



Reikalavimai WEB sistemos duomenų bazei

- WEB sistema turi turėti duomenų bazę, kuri:
 - saugotu papildomus duomenis, susijusius su **konkrečia paslauga**, tačiau kurių **konkreti paslauga** išsaugoti neleidžia



Reikalavimai WEB sistemos aprašymui

- Visi 3 anksčiau paminėti komponentai turi būti aprašyti GitHub repozitorijos (repository) užduotyje (issue) pagal pateiktą šabloną:
 - a. pasirinkta bent viena konkrečios paslaugos esybė (pvz.: dog, beer, artist) ir bent 5 jos atributai
 - b. aprašyti bent 4 sistemas (backend'o) API metodai apibūdinantys esybę:
 - vienai ir kelioms esybėms gauti
 - esybei sukurti
 - esybei atnaujinti
 - esybei ištrinti
 - c. aprašyti bent 1 frontend'o vaizdą (page) ir bent 2 vaizde esantys vizualūs komponentai:
 - kelių esybių duomenims pateikti su galimybe pasirinktą esybę pašalinti ir redaguoti
 - naujos esybės kūrimui / pasirinktos esybės atnaujinimui (redagavimui)

WEB sistemos aprašymo plano pavyzdys

- a. Esybė - sweet (id (skaičius), name (max 20 simbolių), image, calories, price)
- b. REST metodai:
 - i. GET /api/sweet/{id} - vienam (pagal ID) / visiems saldumynams grąžinti
 - ii. POST /api/sweet - naujam saldumynui sukurti
 - iii. PUT /api/sweet/{id} - esamam saldumynui (pagal ID) atnaujinti
 - iv. DELETE /api/sweet/{id} - esamam saldumynui (pagal ID) ištrinti
- c. Vaizdas su web sistemos pavadinimu ir:
 - i. Sąrašu, kuris pateikia saldumynus su jų atributais bei atnaujinimo ir trynimo mygtukais
 - ii. Naujo arba atnaujinamo saldumyno forma su laukais kiekvienam atributui ir išsaugojimo bei atšaukimo mygtukais

4 praktikos užduotys

1. Apsilankyti bent 5 iš 6 pratybų (komandinis darbas) (1 balas)
2. Aprašyti kuriamą WEB sistemą (1 balas)
3. Sukurti WEB sistemos backend'ą (1 balas)
4. Sukurti WEB sistemos frontend'ą (1 balas)
5. WEB sistemoje panaudoti duomenų bazę (1 balas)
6. Papildomai (1 balas)
 - a. naudoti konkrečią paslaugą apsaugotą OAuth būdu
 - b. sudiegti sistemą į debesį (Cloud)
 - c. panaudoti daugiau nei vieną konkrečią paslaugą
 - d. panaudoti daugiau nei vieną esybę arba sukurti daugiau nei vieną vaizdą (page)
 - e. parašyti testus sistemos API arba vizualiems komponentams
 - f. padaryti prisijungimą prie sistemos
 - g. (kita - prieš tai suderinus su dėstytoju)

Konkrečios paslaugos nagrinėjimas

- Chrome Restlet Client addon (jskiepis)

The screenshot shows the Chrome Restlet Client addon interface. At the top, there are tabs for 'CLIENT', 'REQUESTS' (which is selected), and 'SCENARIOS'. The main area is titled 'Test 1'. It displays a 'METHOD' dropdown set to 'GET', a 'SCHEME' dropdown with 'https://api.abalin.net/namedays?day=20&month=6', and a 'Send' button. Below this, under 'QUERY PARAMETERS', there are two entries: 'day' with value '20' and 'month' with value '6'. A 'Save' button is located at the top right of the main form. Below the main form, there are sections for 'HEADERS' and 'BODY'. The 'BODY' section notes: 'XHR does not allow payloads for GET request.' At the bottom, there is a 'HISTORY' tab showing three recent requests:

| Date | Method | URL | Status | Cache | Time |
|---------------------|--------|--|--------|-------|-------|
| 2018 Feb 4 18:25:36 | GET | https://api.abalin.net/namedays?day=20&month=6 | 200 | cache | 441ms |
| 2018 Feb 4 18:24:40 | GET | https://api.abalin.net/namedays?day=19&month=6 | 200 | cache | 629ms |
| 2018 Feb 4 18:24:12 | GET | https://api.abalin.net/namedays?day=18&month=6 | 200 | cache | 609ms |

Rezultatas

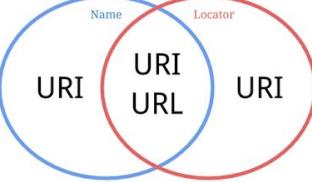
- Išrinktas ‘organizacijos’ pavadinimas
- ‘Organizacija’ ‘jsteigta’ GitHub ir Slack platformose
- Pristatyta užduotis kiekvienam ‘organizacijos’ nariui
- Šių pratybų tikslas:
 - išsirinkti sritį ir pradėti nagrinėti pasirinktą **konkrečią paslaugą**
- Kitų pratybų tikslas:
 - aprašyti pasirinktą **konkrečią paslaugą** pagal pateiktą šabloną



T2 - HTTP protokolas

```
<!DOCTYPE html>
<html>
<!-- created 2010-01-01 -->
<head>
<title>sample</title>
</head>
<body>
<p>Voluptatem accusantium  
totam rem aperiam.</p>
</body>
</html>
```

HTML



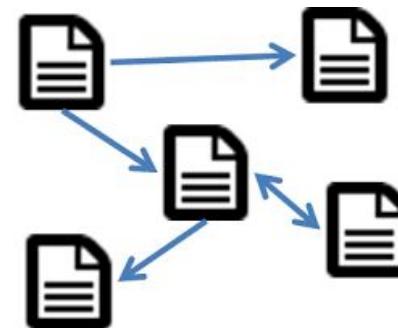
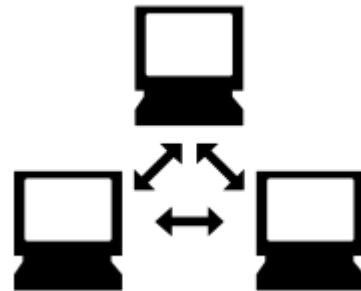
http://

Iki 1990-ų spalio, Timas sukuria tris fundamentalias technologijas, kurios išlieka šiuolaikinio žiniatinklio (web) pagrindu

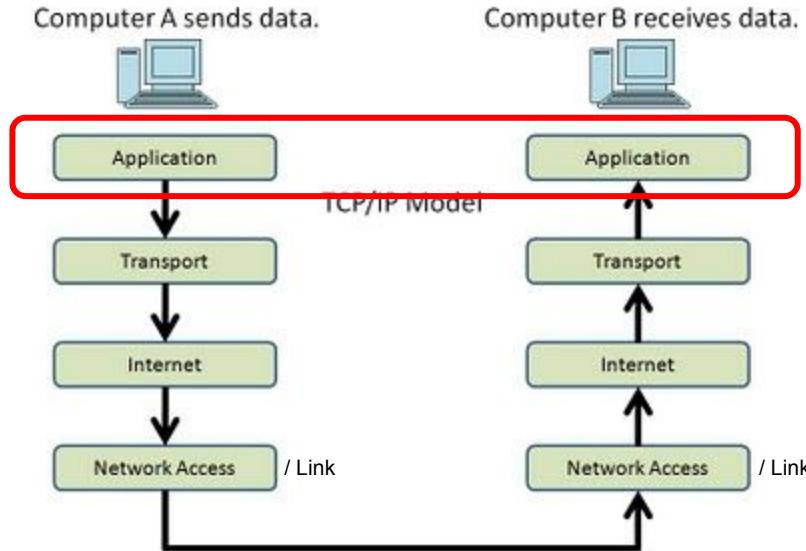


Internetas ir Žiniatinklis

- Internetas - susijungusių kompiuterių tinklų globali sistema
- Žiniatinklis (WWW arba Web) - globalus rinkinys dokumentų ir kitų resursų sujungtų nuorodomis (hyper links) ir uniforminiai resursų identifikatoriai (uniform resource identifiers (URIs))
- Šiame kurse nagrinėsime technologijas susijusias su žiniatinkliu (Web)



TCP/IP Modelis - Aplikacijos Sluoksnis

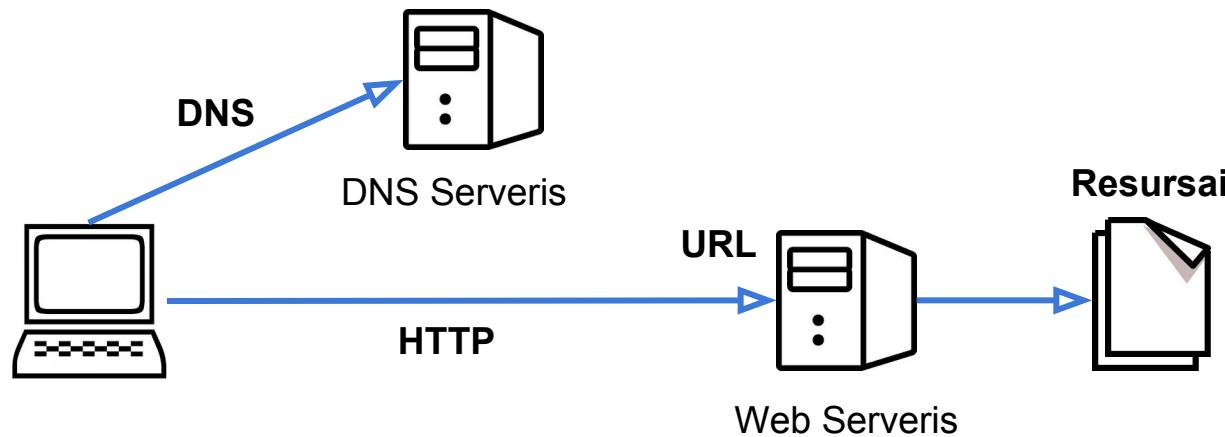


Aplikacijos sluoksnis (Application Layer)

- Aplikacijos sluoksnio protokolai naudoja žemiau esančių sluoksniių protokolus (dažniausiai TCP, UDP ir IP) sistemos duomenims perduoti. Pvz.:
 - Failų perdavimo protokolas (File Transfer Protocol (**FTP**)) failams pasidalinti ir atsisiųsti:
 - Sistemos: FTP serveris ir FTP klientas
 - Transporto sluoksnio protokolas: TCP (standartiniai prievadai: 20 (duomenų perdavimas), 21 (komandų vykdymas))
 - Interneto sluoksnio protokolas: IP
 - Dinaminio hosto configūravimo protokolas (Dynamic Host Configuration Protocol (**DHCP**)) naudojamas naujam tinklo įrenginiui konfigūruoti
 - Sistemos: DHCP serveris ir DHCP klientas
 - Transporto sluoksnio protokolas: UDP (prievedai: 67 (serveriui) ir 68 (klientui))
 - Interneto sluoksnio protokolas: IP

Žiniatinklis (Web)

- Globalus rinkinys dokumentų ir kitų resursų sujungtų nuorodomis (hyper links) ir uniforminiai resursų identifikatoriai (Uniform Resource Identifiers (URIs))
 - Resursai: HTML, CSS, JS, PNG, SVG, JSON, XML,...
 - Nuorodos / Identifikatoriai: URI, URN, URL,...
 - Protokolai: DNS, HTTP,....



Resursai

HTML



CSS

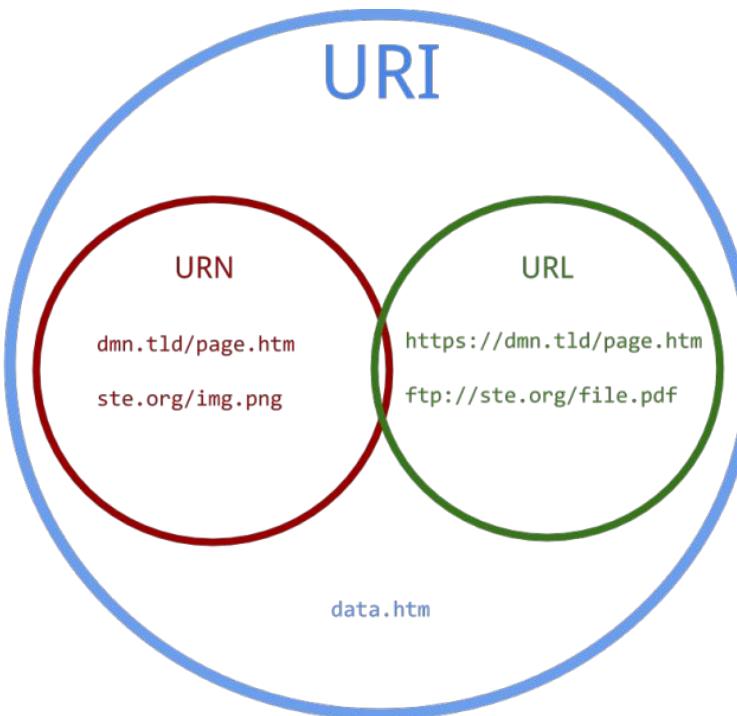


JavaScript



...

Nuorodos: URI, URN, URL



URL

TCP / UDP prievedas

scheme://[user:password@]host[:port][/]path[?query][#fragment]

The diagram illustrates the structure of a URL. It starts with 'scheme://' at the beginning, which is bracketed under the label 'protokolas'. This is followed by optional credentials '[user:password@]'. The next part is 'host', which can include a port number ':port'. A bracket above 'host' and ':port' points to the label 'IP adresas / srities vardas (domain name)'. After the host, there's an optional separator '/', then 'path', then a question mark '?' for 'query', and finally a hash '#' for 'fragment'.

protokolas IP adresas / srities vardas (domain name)

Pavyzdžiai:

<http://google.lt/index.html>

<https://petras:raktas@failai.lt:1021/dokumentai/filmas.mp4?kokybe=gera#2dal is>

<mongodb://petras:raktas@ds1.mlab.com:39187/mydb>

HTTP URL

https://www.google.lt/search?q=labas

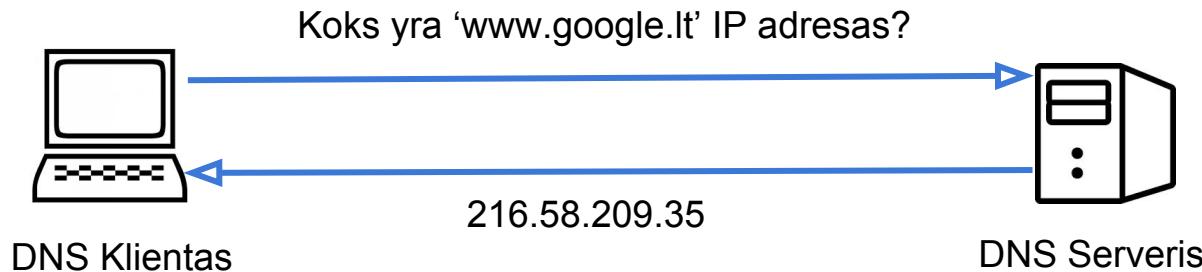
HTTP
protokolas

sritys vardas (domain name)

path ir query

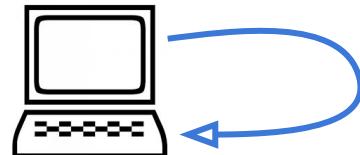
DNS protokolas / sistema

- Žmonėms yra sunku atsiminti IP adresus, bet lengva vardus
- Srities vardų struktūra (Domain Name System)
- Ieškomam srities vardui (domain name) gražina IP adresą
- Naudoja UDP Transporto sluoksnio protokolą (prievedas 53)
- Naudoja IP Interneto sluoksnio protokolą



DNS protokolas / sistema - OS DNS klientas

- Kai savo naršyklėje suvedate: “<http://www.google.lt>” ir paspaudžiat “Enter”:
 - Naršyklė iš paduoto URL pasiima srities vardą: “www.google.lt”
 - Naršyklė naudoja Operacinės Sistemos DNS klientą
 - DNS klientas gali jau turėti adresą savo slėptuvėje / keše (cache) ir jį grąžinti
 - Kitu atveju prie srities vardo yra pridedamas “.” (šaknis (root)) ir nuo šaknies yra pradedama IP adreso paieška (pvz.: “www.google.lt” -> “www.google.lt.”)

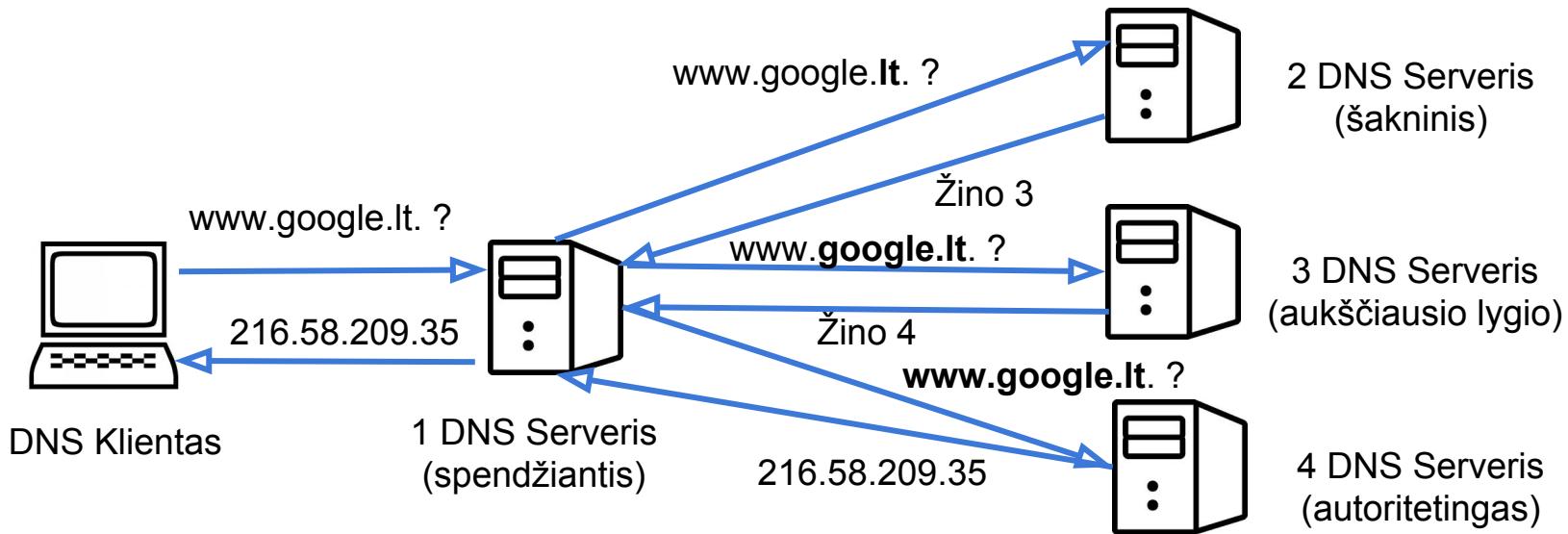


Koks yra ‘www.google.lt’ IP adresas?

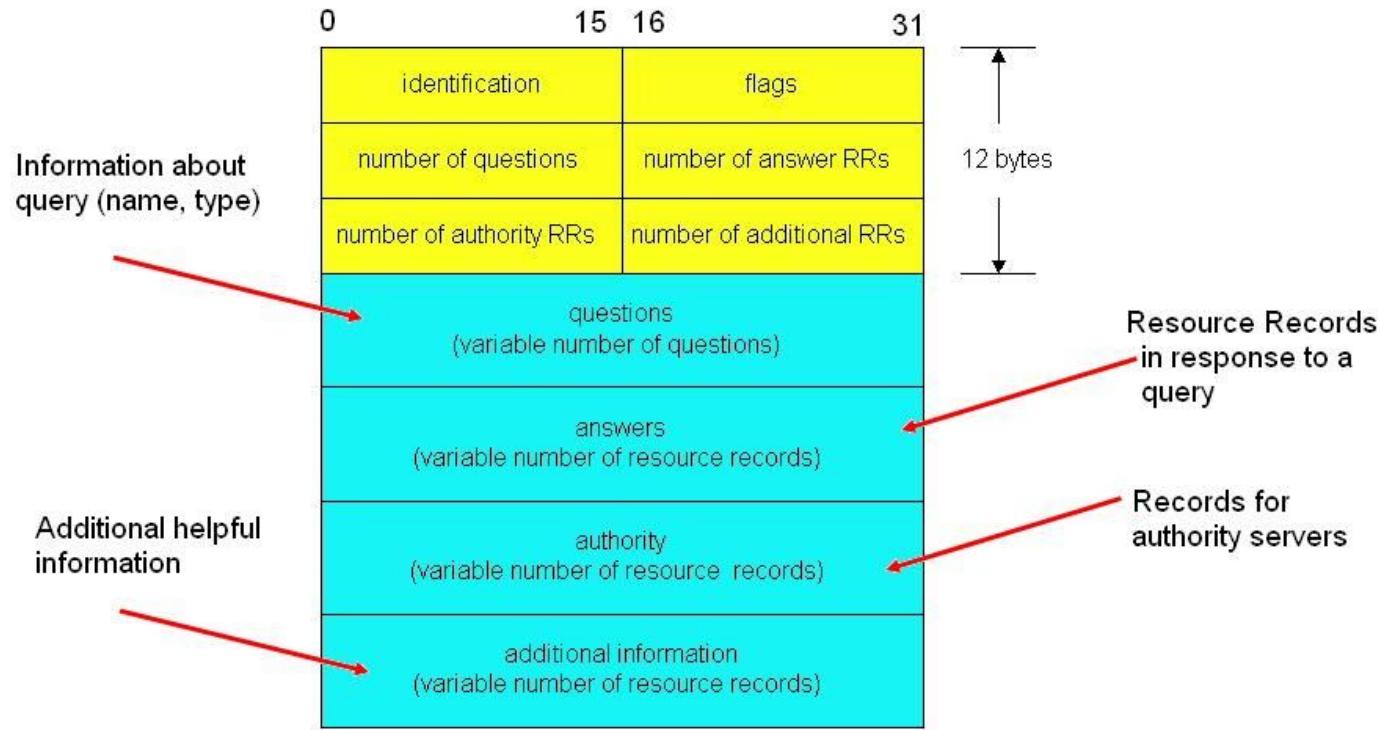
OS DNS Klientas

DNS protokolas / sistema - DNS serveriai

- Srities vardas turi hierarchinę struktūrą, pvz.: www.google.lt. struktura yra
 - a. <tuščias vardas> (šakninis vardas (root name))
 - lt (aukščiausio lygio domenas (Top Level Domain (TLD)))
 - google (autoritetingas vardas(Authoritative Name))
 - www



DNS protokolo žinutė



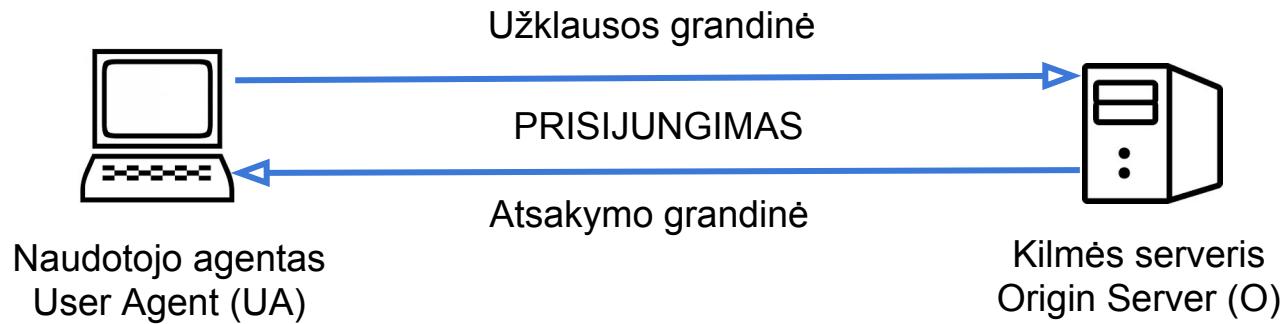
HTTP

- Hiperteksto perdavimo protokolas (Hypertext Transfer Protocol)
- Istoriskai pavadinimas nusako, kad perduodamas tik Hipertekstas, tačiau šiais laikais gali būti perduodama bet kokia informacija (į abi puses), pvz.:
 - paveikslėliai (PNG, JPG, ...)
 - video (MP4, ...)
 - programų skriptai (JavaScript, ...)
 - web svetainės stilius (CSS, ...)
 - archyvai (ZIP, GZIP, ...)
- Standartiskai naudoja TCP Transporto sluoksnio protokolą (prievedas 80)
- Naudoja IP Interneto sluoksnio protokolą

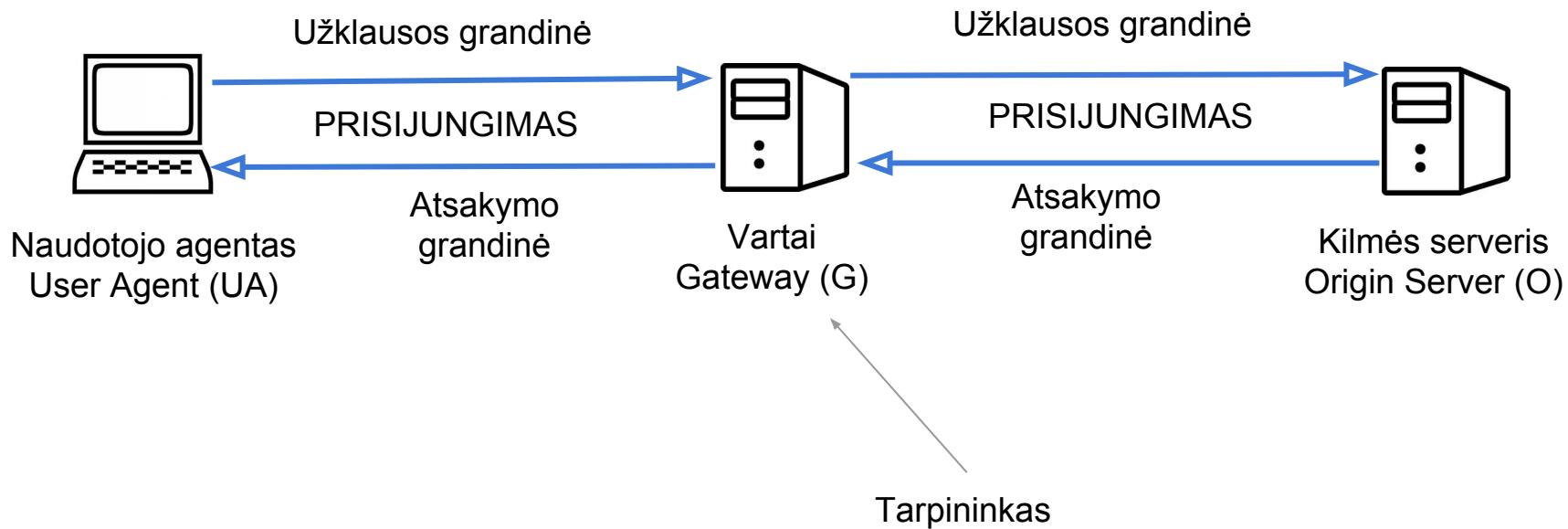
HTTP - standartas

- Internet Engineering Task Force (IETF)
- Hypertext Transfer Protocol (HTTP/1.1):
 - **RFC7230 - "Message Syntax and Routing"**
 - **RFC7231 - "Semantics and Content"**
 - RFC7232 - "Conditional Requests"
 - RFC7233 - "Range Requests"
 - RFC7234 - "Caching"
 - RFC7235 - "Authentication"
- **Hypertext Transfer Protocol Version 2 (HTTP/2)**

HTTP - užklausos ir atsakymo protokolas



HTTP - tarpininkas



HTTP protokolo žinutė

- Užklausa:

```
GET http://localhost/api/ping HTTP/1.1
Host: localhost
Connection: keep-alive
User-Agent: Chrome/55.0.2883.87
Accept: */*
Referer: http://localhost/app/index.html
Accept-Encoding: gzip, deflate, sdch, br
Accept-Language: en-US,en;q=0.8,lt;q=0.6
If-None-Match: W/"12-rsPRhBP8aOVLxyWjUzLLlQ"

```
- Atsakymas:

```
HTTP/1.1 200 OK
X-Powered-By: Express
Content-Type: text/plain; charset=utf-8
Content-Length: 19
ETag: W/"4-b9sIeqP7+8uCh6WToJGeYQ"
Date: Tue, 20 Dec 2016 10:23:03 GMT
Connection: keep-alive

```

My response message

Demo - Chome naršyklės Pl kūréjo įrankiai (dev tools)

The screenshot shows the Network tab in the Chrome DevTools developer tools. The main pane displays a list of network requests for the domain `www.google.lt`. One specific request is highlighted: `googlelogo_color_120x44dp.png`, which has a status code of 200. The right-hand panel provides detailed information about this request, including Headers, Response, and Timing sections.

Request URL: `https://www.google.lt/`
Request Method: GET
Status Code: 200
Remote Address: 64.233.164.94:443

Response Headers:

- `alt-svc: quic=:443; ma=2592000; v="35,34"`
- `cache-control: private, max-age=0`
- `content-encoding: gzip`
- `content-type: text/html; charset=UTF-8`
- `date: Sun, 08 Jan 2017 10:28:29 GMT`
- `expires: -1`
- `server: gws`
- `status: 200`
- `x-frame-options: SAMEORIGIN`
- `x-xss-protection: 1; mode=block`

Request Headers:

- `⚠ Provisional headers are shown`
- `Upgrade-Insecure-Requests: 1`
- `User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/55.0.2883.87 Safari/537.36`

HTTP - metodai

```
GET /api/ping HTTP/1.1
host: localhost:3001
...
...
```

Veiksmas

- **GET** - web resursui gauti (užklausa kūno neturi)
- **POST** - duomenims į serverį nusiųsti
- **OPTIONS** - informacija apie komunikacijų opcijas
- **HEAD** - tas pats kaip GET tik be kūno (body) atsakyme
- **PUT** - resursui serveryje išsaugoti arba atnaujinti
- **DELETE** - resursui serveryje pašalinti (užklausa kūno neturi)
- **TRACE** - sužinoti, ką gavo serveris (kai siunčiama per tarpininkus)
- **CONNECT** - naudojamas su proksiu (proxy)
- **PATCH** - atnaujinti resurso dalį (papildomas metodas apibrėžtas atskiroje specifikacijoje)

HTTP - saugūs (safe) metodai

Metodai, kurie nekeičia serverio būsenos, tik gražina duomenis yra saugūs (safe):

- GET
- HEAD
- OPTIONS
- TRACE

Metodai, kurie keičia serverio būseną, pvz.:

- POST
- PUT
- DELETE
- PATCH

HTTP - silpni (idempotent) metodai

Metodai kurių rezultatas kviečiant daug kartų yra tas pats kaip kviečiant vieną kartą yra silpni (idempotent):

- GET
- HEAD
- PUT
- DELETE
- OPTIONS
- TRACE

Metodas, kurio kiekvienas kvietimas turi unikalų rezultatą, pvz.:

- POST
- PATCH

HTTP - metodų santrauka

| HTTP metodas | Standartinis metodas | Turi užklausos kūną | Turi atsakymo kūną | Saugus (safe) | Silpnas (idempotent) |
|--------------|----------------------|---------------------|--------------------|---------------|----------------------|
| GET | Taip | Ne | Taip | Taip | Taip |
| HEAD | Taip | Ne | Ne | Taip | Taip |
| POST | Taip | Taip | Taip | Ne | Ne |
| PUT | Taip | Taip | Taip | Ne | Taip |
| DELETE | Taip | Ne | Taip | Ne | Taip |
| CONNECT | Taip | Taip | Taip | Ne | Ne |
| OPTIONS | Taip | Taip arba Ne | Taip | Taip | Taip |
| TRACE | Taip | Ne | Taip | Taip | Taip |
| PATCH | Ne | Taip | Taip | Ne | Ne |

HTTP - užklausos URI

- URI formos:

- Origin form (kilmės) - naudojama tiesioginėms užklausoms (be tarpininko)
- Absolute form (absoliuti) - naudojama užklausoms per tarpininką (proxy)
- Authority form (įgaliojimų) - naudojama tik su CONNECT užklausa tuneliui sukurti (tunnel):
 - CONNECT `www.example.com:8080` HTTP/1.1
 - ...
- Asterisk form (asterikso) - naudojama tik su OPTIONS užklausa serverio mastu (palyginus su kitomis užklausomis yra resurso mastu):
 - OPTIONS `*` HTTP/1.1
 - ...

```
GET /api/ping HTTP/1.1
Host: localhost:3001
...
...
```

HTTP - užklausos URI kilmės forma (origin form)

- http://www.example.com/path/resource.html:
 - GET /path/resource.html HTTP/1.1
 - ...
- http://www.example.com/path/book?id=2:
 - GET /path/save?id=2 HTTP/1.1
 - ...
- http://www.example.com:
 - GET / HTTP/1.1
 - ...

HTTP - užklausos URI absoluti forma (absolute form)

- `http://www.example.com/path/resource.html`:
 - `GET http://www.example.com/path/resource.html HTTP/1.1`
 - ...
- `http://www.example.com/path/book?id=2`:
 - `GET http://www.example.com/path/save?id=2 HTTP/1.1`
 - ...
- `http://www.example.com`:
 - `GET http://www.example.com/ HTTP/1.1`
 - ...

HTTP - versija

- HTTP/<major>.<minor>
 - Versija 0.9 išleista 1991 metais
 - Versija 1.0 išleista 1996 metais
 - Versija 1.1 išleista 1997 metais (papildymas 1999 ir 2014 metais)
 - Versija 2.0 išleista 2015 metais
 - Papildo 1.1 versiją susitarimais kaip efektyviau ir greičiau siužti informaciją

```
GET /api/ping HTTP/1.1
Host: localhost:3001
X-Powered-By: Express

...
```

HTTP/1.1 200 OK
X-Powered By: Express
Content-Type: application/json
Content-Length: 123
Date: Mon, 25 Jul 2022 10:45:21 GMT
Connection: keep-alive
ETag: "5f1d0140-7d8"
Vary: Accept
...
{
 "id": 1,
 "title": "Cardiome",
 "body": "Cardiome is a medical device company that develops software solutions for heart monitoring.",
 "category": "Medical Devices",
 "rating": 4.5
}

HTTP - būsenos kodas ir priežastis

- **1xx** - Informacinis (Informational): užklausa buvo gauta, tēsiamas apdorojimas
- **2xx** - Sėkmingas (Successful): užklausa buvo sėkmingai gauta, suprasta ir priimta
- **3xx** - Peradresavimas (Redirection): Reikalingas kitas veiksmas norint kad užklausa būtų užbaigta
- **4xx** - Kliento klaida (Client Error): Užklausa turi klaidą arba negali būti išpildyta
- **5xx** - Serverio klaida (Server Error): Serverui nepavyko įvykdyti validžios užklausos

HTTP - 2xx būsenos kodas ir priežastis

- 2xx - Sėkmingas (Successful):
 - 200 OK - užklausa pavyko
 - 201 Created - užklausa pavyko, buvo sukurtas resursas
 - 202 Accepted - užklausa pavyko, tačiau dar nebuvo baigtas vykdyti
 - 204 No Content - užklausa pavyko, tačiau atsakymas yra tuščias

HTTP - 4xx būsenos kodas ir priežastis

- 4xx - Kliento klaida (Client Error):
 - 400 Bad Request - serveris negali apdoroti užklausos dėl jos nevalidumo
 - Trūksta informacijos
 - Neteisingas duomenų tipas
 - 403 Forbidden - užklausa suprasta, tačiau klientui draudžiamas jos vykdymas
 - Naudotojas tam neturi teisių
 - Neteisingas slaptažodis / slapukas
 - 404 Not Found - nerastas užklausiamas resursas
 - 405 Method Not Allowed - metodas nepalaukomas su nurodomu resursu
 - 408 Request Timeout - serveris nespėjo sulaukti pilnos užklausos per tam skirtą laiką

HTTP - 5xx būsenos kodas ir priežastis

- 5xx - Serverio klaida (Server Error):
 - 500 Internal Server Error - serveryje įvyko nenumatyta klaida
 - Nepavyko ištraukti informacijos iš duomenų bazės
 - Baigėsi laikinoji atmintis
 - 501 Not Implemented - serveris dar nepalaiko norimos užklausos
 - 503 Service Unavailable - nerastas užklausiamas resursas
 - Serversi perkrautas užklausomis
 - Paslauga kvietimo metu yra atnaujinama

HTTP - antraštė

CET /api/ping HTTP/1.1
Host: localhost:3001
HTTP/1.1 200 OK
X-Powered-By: Express

- Rakto-reikšmės sąrašas (key-value list)
 - Raktas: tarp didžiųjų ir mažujų raidžių nėra skirtumo (case insensitive)
- Galimos sukurtos naujos antraštės, pvz.:
 - Token
 - The-Address-Of-A-Page
- Naudojamos standartinės antraštės skirstomos į:
 - Užklausos ir atsakymo antraštės kategorijas - antraštės naudojamos tiek užklausos, tiek ir atsakymo žinutėse
 - Užklausos antraštės kategorijas - antraštės naudojamos tik užklausos žinutėje
 - Atsakymo antraštės kategorijas - antraštės naudojamos tik atsakymo žinutėje

HTTP - užklausos ir atsakymo antraštės

- Užklausos ir atsakymo antraštės kategorijos:
 - Connection - prisijungimo (TCP) valdymas. Pvz.:
 - Connection - prisijungimo indikacija (pvz.: keep-alive - likti prisijungus, close - uždaryti)
 - Representation - apibūdina resursą kaip abstrakciją. Pvz.:
 - **Content-Type** - kūno duomenų tipas (pvz.: text/html - HTML dokumentas)
 - Content-Encoding - kūno duomenų užkodavimas (pvz.: compress - suspaudimas)
 - Content-Language - kūno duomenų kalba (pvz.: en-US - Amerikos anglų)
 - Payload - apibūdina resursą kaip duomenis. Pvz.:
 - Content-Length - kūno duomenų ilgis baitais
 - Transfer-Encoding - kūno duomenų užkodavimas (pvz.: chunked - transportavimas kūnų suskaidžius į dalis)

HTTP - Content-Type antraštė

Svarbiausia antraštė web app'so kūrėjui:

- Nurodo siunčiamo ir gaunamo kūno tipą MIME formatu <tipas>/<potipis>.

Pvz.:

- text/html - tipas 'text' nurodo, kad kūne yra tekstas, o potipis 'html' - tai HTML tekstas
- text/css - CSS stiliaus tekstas
- image/png - PNG paveikslėlis
- image/jpeg - JPEG paveiklėlis
- video/mp4 - video MP4 formatu
- audio/x-aac - audio AAC formatu
- application/javascript - JavaScript aplikacijos kodas
- application/json - aplikacijai skirti duomenys JSON (JavaScript Object Notation) formatu

HTTP - užklausos antraštės

- Užklausos antraštės kategorijos:
 - Controls - nurodo kaip reiktų užklausą valdyti. Pvz.:
 - **Host** - nurodo domeną, kuriam siunčiama žinutė. Vienintelė privaloma antraštė!
 - Conditionals - nurodo kaip reiktų pateikti resurso duomenis
 - Content Negotiation - nurodo, kokį atsakymo turinį priimtų / galėtų priimti (prioriteto tvarka). Pvz.:
 - **Accept** - priimamas turinio tipas (pvz.: text/plain; q=0.5, text/html)
 - Accept-Charset - priimamas simbolių kodavimas (pvz.: iso-8859-5, unicode-1-1;q=0.8)
 - Accept-Encoding - priimamas žinutės kodavimas (pvz.: compress;q=0.5, gzip;q=1.0)
 - Accept-Language - priimama kalba (pvz.: lt, en-gb;q=0.8, en;q=0.7)
 - Authentication Credentials - nurodo identifikacinius duomenis
 - Request Context - apibūdina užklausos kontekstą. Pvz.:
 - Referer - URI resurso iš kurio buvo gauta nuoroda, kuria vykdoma užklausa
 - User-Agent - užklausą vykdantis klientas (pvz.: Chrome/55.0.2883.87)

HTTP - Host antraštė

Kartais, kai HTTP klientas siunčia žinutę adresu: `http://example.com/index.html` ,
HTTP serveris gauna tokią žinutęs antraštę: `GET /resource.html HTTP/1.1`

Žinutė savyje neturi example.com domeno ir jei HTTP serveris teikia paslaugą
keliems domenams (virtual hosting), jis nežino kurio domeno resursą pateikti.

Host antraštėje yra nurodytas norimo resurso domenas ir prievadas (pasirinktinai):

- `example.com`
- `example.com:8080`
- `sample.lt`
- `sample.lt:3000`

HTTP - Accept antraštė

Priimamas (norimas) turinio tipas (MIME formatu) prioriteto tvarka

- jei antraštės nėra - tinka bet koks atsakymo formatas
- */* - tinka bet koks formatas
- tipas/* - tinka bet koks nurodyto tipo potipis
- tipas/potipis - tinka nurodytas tipas ir potipis
- tipas/potipis; q=0.8 - tinka nurodytas tipas ir potipis su svoriu 0.8 ($0 \leq q \leq 1$)

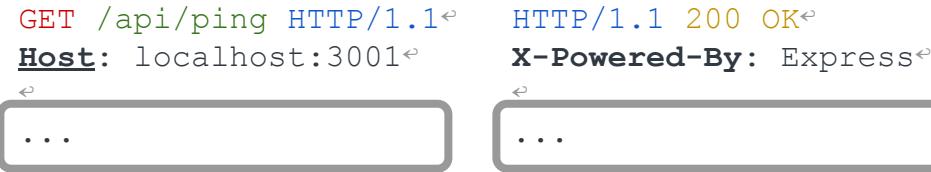
Pvz:

- text/plain; q=0.5, text/html, text/x-dvi; q=0.8, text/x-c - norimas
text/plain arba text/x-c formatas, kitu atveju - text/x-dvi (nes svoris $0.8 > 0.5$),
galiausiai priimamas text/plain formatas

HTTP - atsakymo antraštės

- Atsakymo antraštės kategorijos:
 - Control data - papildo atsakymą metaduomenimis apie resursą. Pvz.:
 - Date - žinutės data ir laikas (pvz.: Tue, 15 Nov 1994 08:12:31 GMT)
 - Location - URI resurso susijusio su užklausos grąžintu atsakymu
 - Retry-After - laikas po kurio vykdyti kitą užklausą (pvz.: 120)
 - Validator Header Fields - pasirinktos reprezentacijos metaduomenys
 - Authentication Challenges - nurodomas autentifikavimosi būdas
 - Response Context - apibūdina atsakymo kontekstą. Pvz.:
 - Allow - išvardina leidžiamus kvesti metodus (pvz.: GET, HEAD, PUT)
 - Server - užklausą aptarnavęs klientas (pvz.: Apache/2.2.22)

HTTP - kūnas



Bitai - jų reikšmė priklauso nuo antraščių:

1. Transfer-Encoding - kūno duomenų transportavimas
 - a. Jei nėra nurodyta - tuomet duomenų ilgį nusako Content-Length antraštė
2. Content-Encoding - kūno duomenų užkodavimas
 - a. Jei nėra nurodyta - tuomet duomenys nėra užkoduojami
3. Content-Type - kūno duomenų interpretavimas
 - a. Jei nėra nurodyta - tuomet gavėjas
 - i. Arba interpretuoja duomenis kaip application/octet-stream (dvejetainj failą (binary file))
 - ii. Arba bando formatą nustatyti

HTTP - kūno persiuntimas

1. Siuntėjas

- a. Gauna duomenų bitus pagal norimą formatą ir nurodo tipą `Content-Type` antraštėje
- b. Jei nori bitus užkoduoti, tuomet tai padaro ir kodavimą nurodo `Content-Encoding` antraštėje
- c. Jei nori (užkoduotus) bitus transportuoti specifiniu būdu tuomet ji nurodo `Transfer-Encoding` antraštėje ir vykdo bitų siuntimą pasirinktu specifiniu būdu, kitu atveju nurodo duomenų ilgį `Content-Length` antraštėje

2. Gavėjas

- a. Jei nurodyta `Transfer-Encoding` antraštė - tuomet gautus bitus gavėjas priima nurodytu būdu, kitu atveju nuskaito bitų skaičių nurodytą `Content-Length` antraštėje
- b. Jei nurodyta `Content-Encoding` antraštė, tuomet gavėjas nuskaitytus bitus atkoduoja antraštėje nurodytu būdu
- c. (Atkoduotas) bitus gavėjas interpretuoja kaip `Content-Type` antraštėje nurodytą formatą

HTTP - kūno persiuntimo 1 pavyzdys

1. Siuntėjas
 - a. Gauna siunčiamo DOC failo bitus ir nurodo Content-Type: application/msword antraštę
 - b. Bitų nekoduojant todėl nenurodo Content-Encoding antraštės
 - c. Suskaišiuojant bitus (kaip baitus), nurodo Content-Length: 1024 antraštę ir vykdo užklausą
2. Gavėjas
 - a. Perskaito Content-Length: 1024 antraštę ir nuskaito lygiai 1024 baitus kūno duomenų
 - b. Neranda Content-Encoding antraštės todėl gautų bitų atkoduoti nereikia
 - c. Perskaito Content-Type: application/msword antraštę ir interpretuoja išarchyvuotus bitus kaip DOC failą (pvz. atidaro juos su MS Word programa)

HTTP - kūno persiuntimo 2 pavyzdys

1. Siuntėjas
 - a. Gauna siunčiamo DOC failo bitus ir nurodo Content-Type: application/msword **antraštę**
 - b. Užkoduojas bitus (suarchyuoja) ir nurodo Content-Encoding: gzip **antraštę**
 - c. Nurodo Transfer-Enconding: chunked **antraštę**, skaido užkoduotus bitus dalimis ir vykdo užklausą (siunčia dalis vieną po kitos, su galimomis pauzėmis tarp dalių)
2. Gavėjas
 - a. Perskaito Transfer-Enconding: chunked **antraštę**, gautus bitus gavėjas skaito dalimis ir sujungia (jungimas baigiamas, kuomet yra gaunama tuščia dalis)
 - b. Perskaito Content-Encoding: gzip **antraštę** ir sujungtus bitus išarchyuoja
 - c. Perskaito Content-Type: application/msword **antraštę** ir interpretuoja išarchyvuotus bitus kaip DOC failą (pvz. atidaro juos su MS Word programa)

HTTP - kelios svarbiausių HTTP 1.1 ypatybių

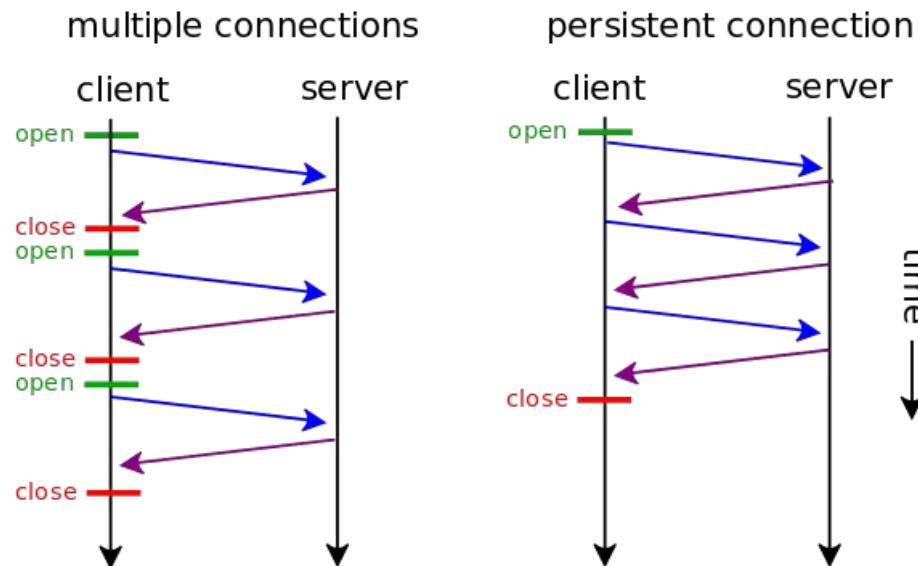
- Išliekantis prisijungimas (Persistent connection)
 - vieno prisijungimo panaudojimas daugiau nei vienai HTTP užklausai vykdyti
- Lygiagretinimas (Pipelining)
 - daugiau nei vienos užklausos siuntimas nelaukiant atsakymo

HTTP - Išliekantis prisijungimas

- HTTP protokolas siunčia užklausą ir atsakymą prieš tai atlikus prisijungimą (praktiškai visuomet TCP prisijungimą)
- Atlikti naują prisijungimą kiekvienai užklausai siūsti ir atsakymui gauti nėra efektyvu
- Todėl atlikus prisijungus jis išlieka tam tikram laikui, kad per tą laiką būtų galima atlikti daugiau nei viena užklausą ir sulaukti atsakymų

HTTP - Išliekantis prisijungimas

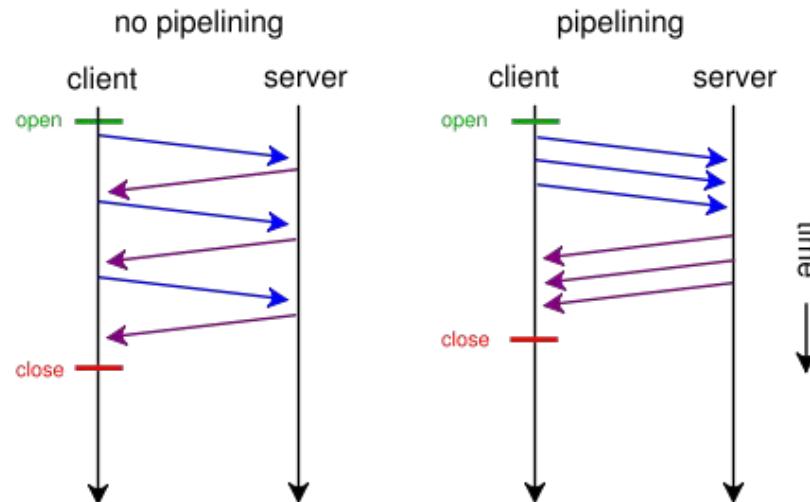
HTTP protokolas siunčia užklausą ir atsakymą prieš tai atlikus prisijungimą (praktiškai visuomet TCP prisijungimą)



HTTP - Lygiagretinimas

HTTP protokolas kiekvienai užklausai visuomet sulaukia atsakymo

SVARBU! Dėl eilės blokavimo ir kitų problemų ši HTTP 1.1 savybė yra praktiškai nenaudojama (žiūrėti HTTP 2 protokolo multipleksavimo savybę)

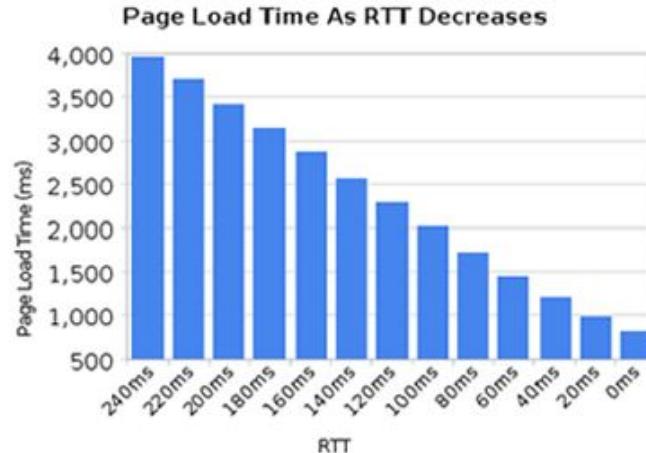


HTTP 1.1 problemas

- Nemaža specifikacija:
 - HTTP 1.0 - 60 puslapių
 - HTTP 1.1 - 176 pusliai
 - HTTP 1.1 (atnaujintas) - $88 + 100 + 27 + 24 + 42 + 18$ puslapių
- Daug neprivalomų dalykų
 - Beveik niekas nepalaiko visko
- Nepakankamai išnaudojamas TCP protokolas (app'sas užkraunamas lėčiau)
 - Brangu patvirtinti kiekvieną TCP paketą (max paketo dydis - 64 K)

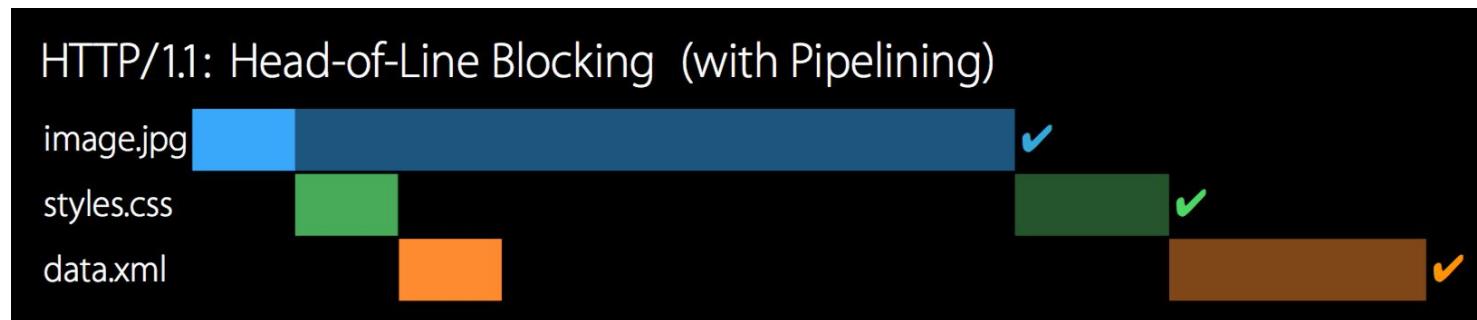
HTTP 1.1 problemos

- Latencija (latency) - delsimas iki kol pradedami siųsti duomenys
- Round-trip delay time (RTT) - laikas reikalingas duomenims nusiųsti ir patvirtinti, kad duomenys buvo sėkmingai nusiųsti



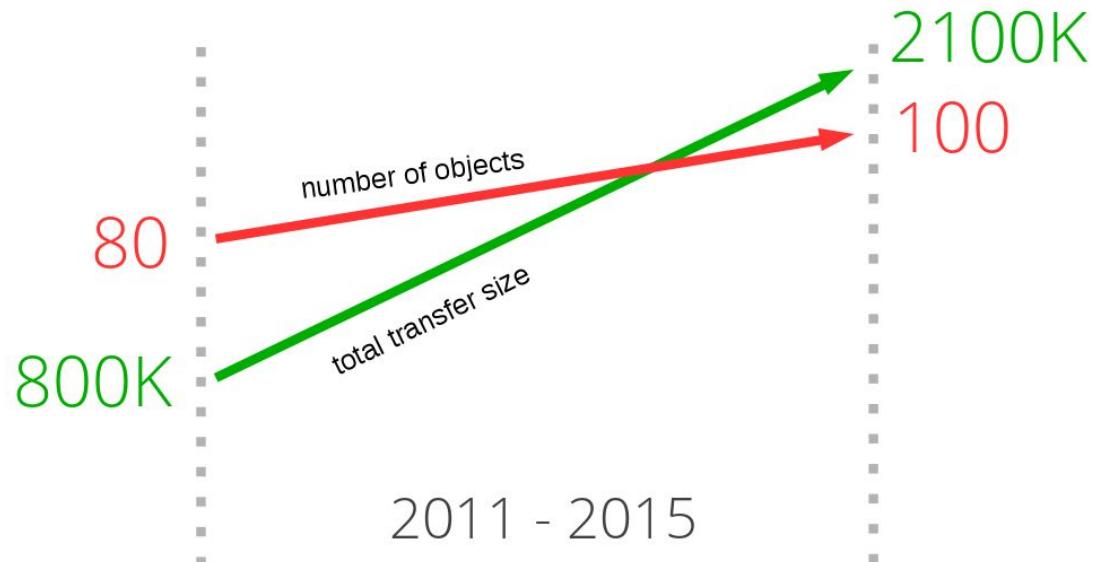
HTTP 1.1 problemos

- HTTP 1.1 vienu metu leidžia siųsti tik vieną užklausą
- Eilės blokavimas (Head of line blocking) - negalėjimas siųsti kitos užklausos, kuomet nepasibaigė dabartinės užklausos siuntimas



HTTP 1.1 problemas

- Populiariausių web svetainių tendencija - visko atsisiųsti reikia vis daugiau



HTTP 1.1 problemų apėjimas

- Sprintingas (sprinting) - daugybės pavaikslukų sudėjimas į vieną
 - Tačiau atskiri paveikslukai turi būti karpomi JavaScripto ir CSS priemonėmis
 - Tačiau iš cache'o vienu metu pašalinami visi paveikslukai



HTTP 1.1 problemų apėjimas

- Inliningas (inlining) - resursų duomenų įterpimas į kodą
 - Tačiau tokį kodą nepatogu skaityti

```
.icon1 {  
    background: url(data:image/png;base64,<data>) no-repeat;  
}  
  
.icon2 {  
    background: url(data:image/png;base64,<data>) no-repeat;  
}
```

Duomenys įterpiami čia

Duomenys įterpiami čia

HTTP 1.1 problemų apėjimas

- Apjungimas (concatenation) - kodo failų apjungimas į vieną kodo failą
 - Tačiau tokį kodą nepatogu skaityti

```
12
13 | holder max min multiple pattern required step".split(" "));f.inputtypes=function(a){for(var b=0,c,k=a.length;t
14 | })N[a[b]]!=!c}return N};"search tel url email datetime date month week time datetime-local number range color"
15 | ";"(e.head||e.getElementsByTagName("head")[0]).appendChild(b);c.id="modernizr";l.appendChild(c);a=c.offsetHeight
16 | ]=="function";if(typeof c[b]!="undefined")c[b]=u;c.removeAttribute(b)};return k})},G={}.hasOwnProperty,R=R=
17 | n("box-flex:1;")+"width:10px;";a.appendChild(b);l.appendChild(a);var c=b.offsetWidth==42;a.removeChild(b);
18 | }catch(c){}return false};d.touch=function(){return"ontouchstart"in i||Q("@media ("+q.join("touch-enabled"),(")+
19 | document.documentElementNode)");d.history=function(){return!!(i.history&&i.history.pushState)};d.draganddrop=function(
20 | n(){j.cssText="background:url(:),url(:),red url(:)";return/(url\$(.*?\{3\})/.test(j.background)};d.backg
21 | );d.cssanimations=function(){return n("animationName")};d.csscolumns=function(){return n("columnCount")};d.cs
22 | sform","OTransform","msTransform"]);};d.csstransforms3d=function(){var a=!!D(["perspectiveProperty","WebKitPers
23 | neet"]|c.styleSheet;b=k.hasFeature("CSS2","")?function(g){if(!(a&&g))return false;var r=false;try{a.insertRule(
24 | )}{var a=e.createElement("video"),b=!a.canPlayType;if(b){b=new Boolean(b);b.ogg=a.canPlayType('video/ogg; cod
25 | ecs="1"');b.m4a=a.canPlayType("audio/x-m4a;")||a.canPlayType("audio/aac;")return b};d.localStorage=function(
26 | element("div");a.innerHTML=<svg/>;return(a.firstChild&&a.firstChild.namespaceURI)==v.svg);d.smil=function()
27 | sName+=" "+(b?"no-":a);f[a]=b;return f});j.cssText="";E=h=null;i.attachEvent&&function(){var a=e.createElement
28 | "mary|time|video".split("|"),r=g.length,x=RegExp("<(*)|(abbr|article|aside|audio|canvas|details|figcaption|fig
29 | tElementsByTagName(g[p]),I=m.length,t=-1;++t<I;if(m[t].className.indexOf("iepp_")<0)m[t].className+=" iepp_"
30 | .replace(/\bno-ic\h_ "/)+" ic"+t.className+=" "+p.join(" ")+return f};this.document).
```

HTTP 1.1 problemų apėjimas

- Atskyrimas (sharding) - HTTP 1.1 specifikacija leidžia 1-am domenui atidaryti 6-8 jungtis (connections). Todėl tam pačiam serveriui užregistruojami keli domenai ir taip galima atidaryti daugiau jungčių ir greičiau atsisiųsti duomenis.
 - Tačiau tai apsunkina infrastruktūros valdymą

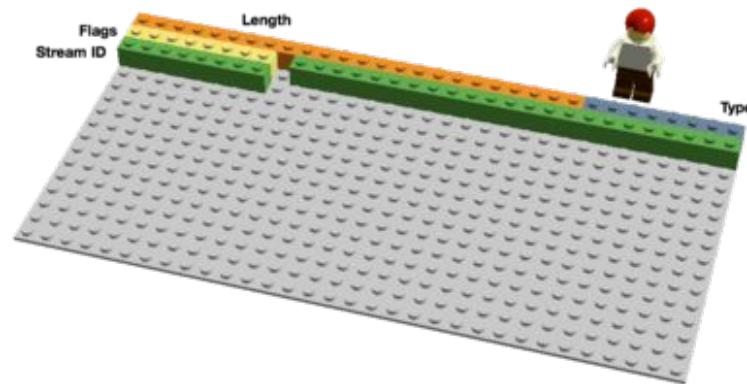
| | | | | | |
|-----------|--------------|--------------------|------|----------|----------|
| ● 200 GET | 265.jpg | x.cdn-expressen.se | jpeg | 12.09 KB | → 249 ms |
| ● 200 GET | 265.jpg | z.cdn-expressen.se | jpeg | 5.92 KB | → 167 ms |
| ● 200 GET | original.jpg | y.cdn-expressen.se | jpeg | 64.28 KB | → 192 ms |
| ● 200 GET | original.jpg | w.cdn-expressen.se | jpeg | 21.88 KB | → 106 ms |
| ● 200 GET | 540.jpg | w.cdn-expressen.se | jpeg | 18.77 KB | → 112 ms |
| ● 200 GET | 128.jpg | z.cdn-expressen.se | jpeg | 3.34 KB | → 55 ms |
| ● 200 GET | 265.jpg | x.cdn-expressen.se | jpeg | 13.00 KB | → 245 ms |
| ● 200 GET | 265.jpg | y.cdn-expressen.se | jpeg | 9.19 KB | → 194 ms |
| ● 200 GET | 540.jpg | w.cdn-expressen.se | jpeg | 13.13 KB | → 108 ms |
| ● 200 GET | 174.jpg | y.cdn-expressen.se | jpeg | 5.66 KB | → 197 ms |
| ● 200 GET | 174.jpg | z.cdn-expressen.se | jpeg | 5.56 KB | → 55 ms |
| ● 200 GET | 174.jpg | w.cdn-expressen.se | jpeg | 5.07 KB | → 111 ms |
| ● 200 GET | 174.jpg | z.cdn-expressen.se | jpeg | 6.16 KB | → 59 ms |
| ● 200 GET | 174.jpg | y.cdn-expressen.se | jpeg | 6.57 KB | → 210 ms |
| ● 200 GET | 174.jpg | y.cdn-expressen.se | jpeg | 4.58 KB | → 12 ms |
| ● 200 GET | 265.jpg | y.cdn-expressen.se | jpeg | 11.49 KB | → 173 ms |

HTTP 2

- Tokie patys URL: http://... https://...
- Tokios pačios užklausos ir atsakymai tik skiriasi jų kodavimas

HTTP 2 - Binarinis protokolas (Binary)

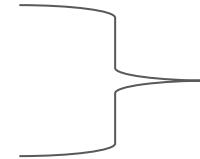
- Užklausos ir atsakymo žinutės suskaidomos į dalis (frames), kurių kiekvienas turi:
 - Length
 - Type
 - DATA
 - HEADERS
 - ...
 - Flags
 - Stream Identifier
 - **Payload** (priklauso nuo tipo (type))



Žinutė: HTTP 1.1 vs HTTP 2

- Užklausa:

```
GET http://a/b HTTP/1.1
Host: localhost
Connection: keep-alive
←
```

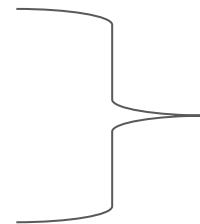


Length: ...
Type: HEADERS
Flags: ...
Stream Identifier: ...

Payload:
:method = GET
:path = /a/b
:scheme = http
Host = localhost

- Atsakymas:

```
HTTP/1.1 200 OK
X-Powered-By: Express
Content-Type: text/plain
←
My response message
```



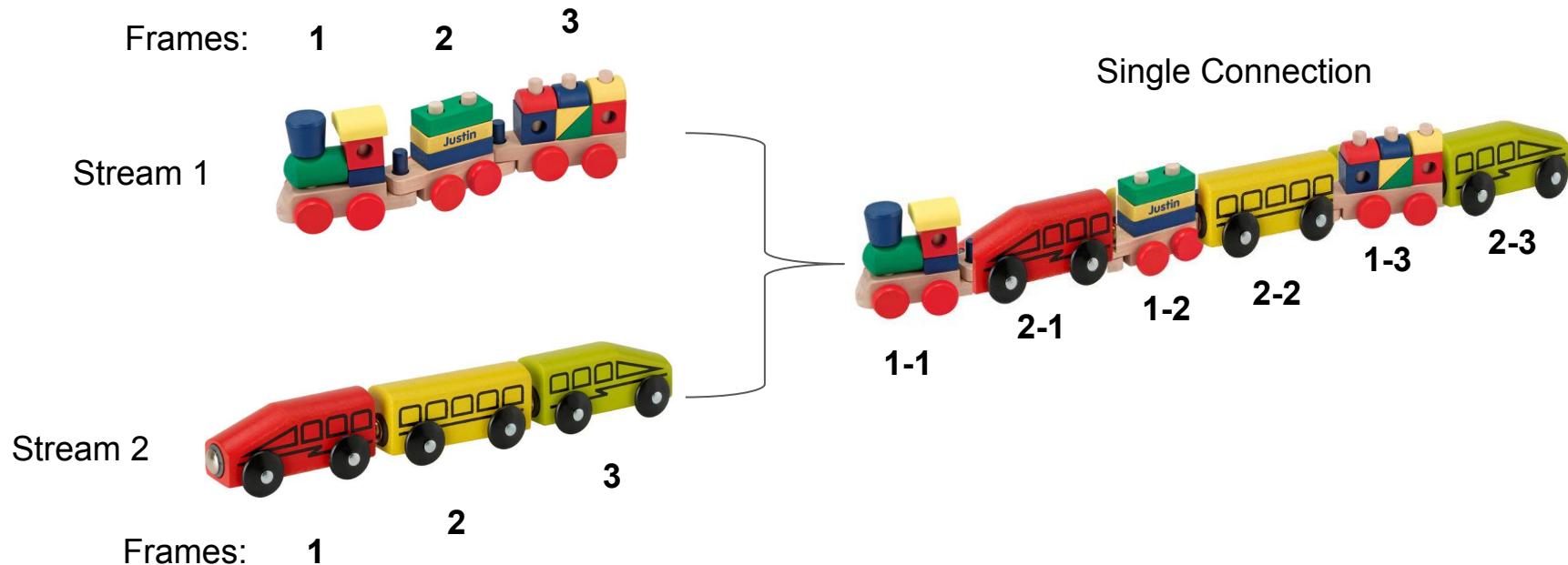
Length: ...
Type: HEADERS
Flags: ...
Stream Identifier: ...

Payload:
:status = 200
X-Powered-By = Express
Content-Type = text/plain

Length: ...
Type: DATA
Flags: ...
Stream Identifier: ...
Payload: My response message

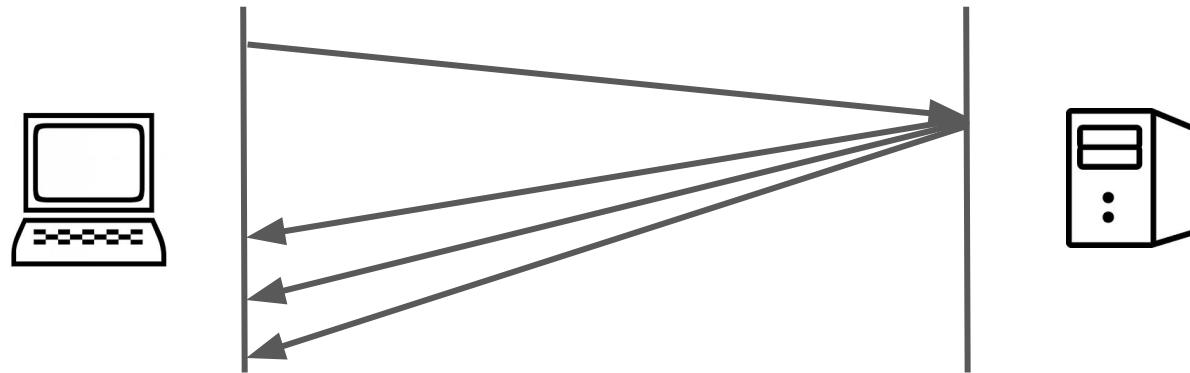
HTTP 2 - Multipleksavimas (Multiplexing)

- Keli srautai (streams) vienoje jungtyje (connection)
- Taigi užtenka vienos jungties (connection) domenui (domain)



HTTP 2 - Serverio push'as (Server push)

- Serveris gali pasiųsti porą atsakymų
 - Prašant HTML failo, kartu yra paduodami ir susiję stiliai bei paveikslukai
- Klientas gali atsisakyti priimti papildomus resursus (RST_STREAM)
 - Tarkime resursai jau egzistuoja naršyklės atmintyje (cache)



HTTP 2 - Kita

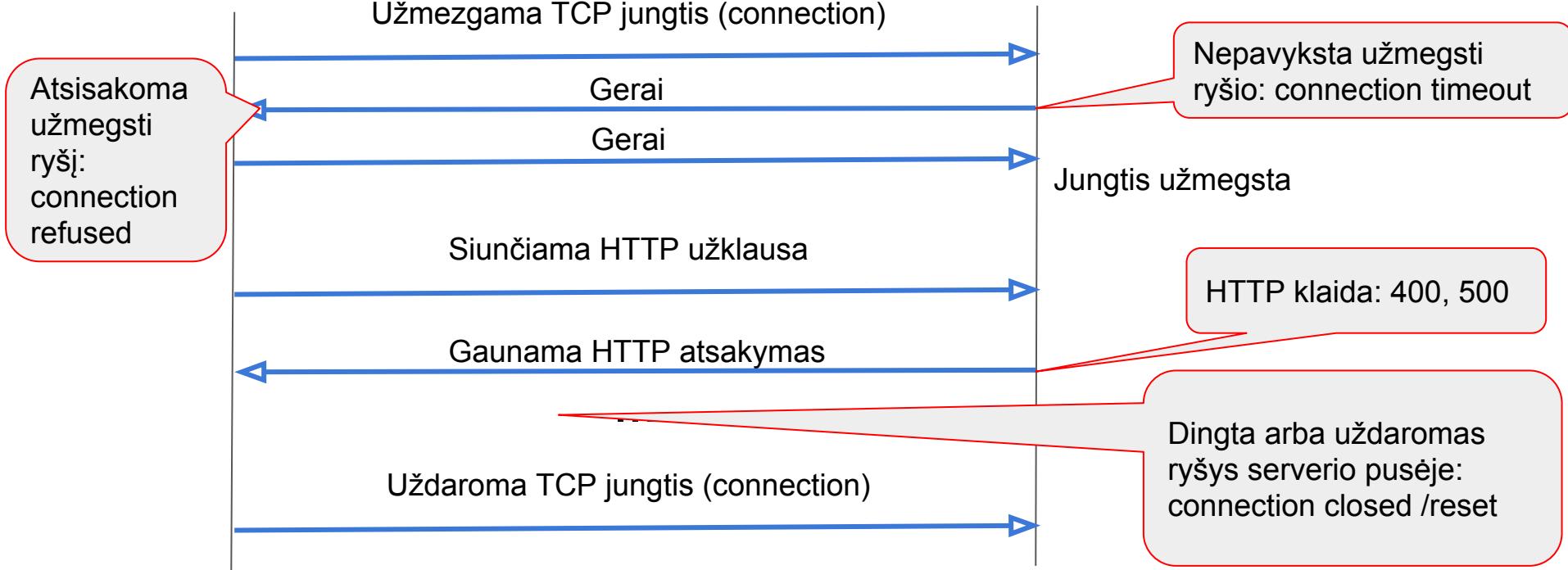
- Srauto prioritetas ir priklausomybės (priorities and dependencies)
 - HTML resursas siunčiamas aukščiausiu prioritetu
 - Stilius ir paveikslukai žemesniu prioritetu
- Antraštės kompresija (header compression)
 - Esant sausainiukui (cookie) jį reikia siųsti su kiekviena užklausa
 - Antraštės kompresija padeda sumažinti žinutės dydį (kad dažnu atveju telptų į vieną TCP paketą)
 - HPACK kompresija padeda išvengti atakų (BREACH, CRIME)
- Apsigalvojus galima užklausą atšaukti (reset)
 - HTTP 1.1 protokolo atveju nutraukiamas prisijungimas, o jo atnaujinimas kainuoja laiko
- Srauto kontrolė (flow control)
 - Panašu į TCP ar SSH protkololus



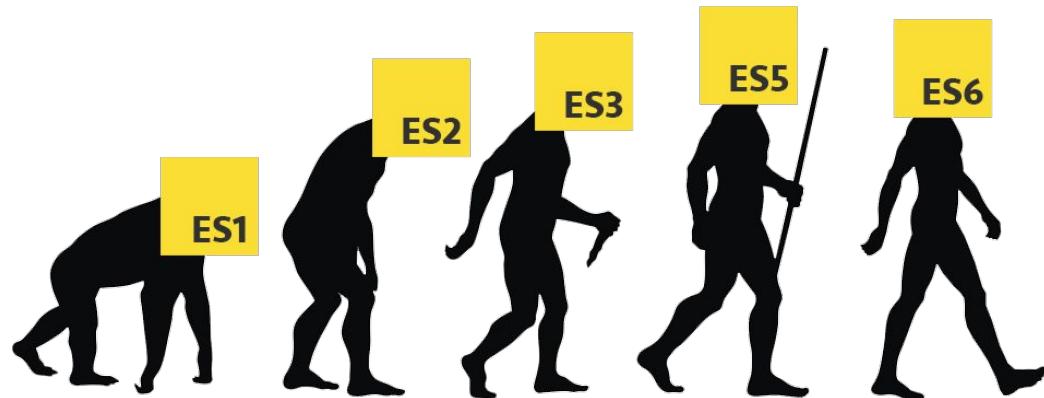
Naudotojo agentas
User Agent (UA)



Kilmės serveris
Origin Server (O)



1997 1998 1999 2009 2015

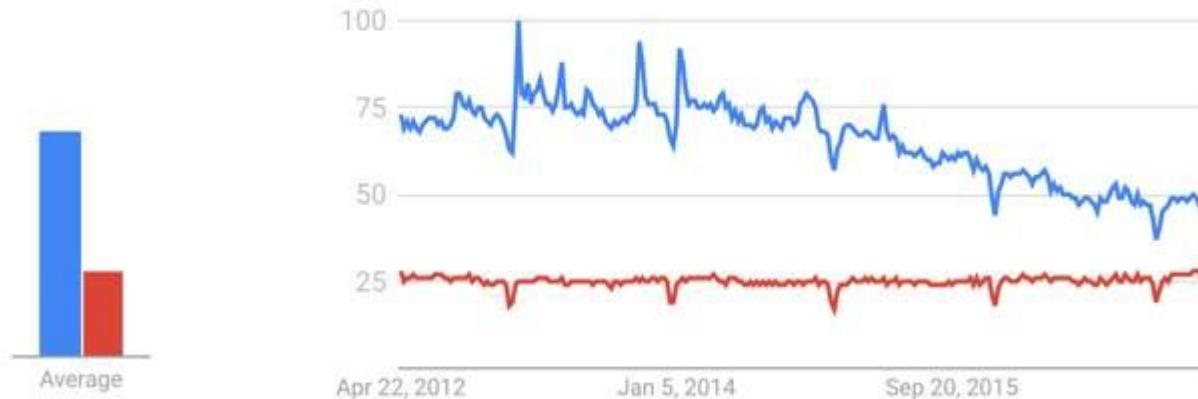


T3 - JavaScript pagrindai

Interest over time

Google Trends

● Java ● JavaScript



Worldwide. Past 5 years.

Procesas (process) prieš Giją (Thread)

- **Procesas (Process)** - operacinės sistemos dalis apibūdinanti leidžiamos programos resursus (unit of resources)
 - kiekvienam procesui paskiriamas laikinosios atminties kiekis (allocate memory) bei yra suteikiama prieiga prie resursų, tokį kaip procesoriai (CPUs) ar įvesties/išvesties (IO) įrenginiai.
- **Gija (Thread)** - proceso dalis, kuri planuoja ir leidžia programos kodą (unit of scheduling and execution):
 - Procesas visuomet turi bent vieną giją
 - Gijos dalinasi procesui paskirtus resursus
 - Kiekviena gija turi savo leidimo steką (stack)

Procesas (process) prieš giją (thread)

Operacinė Sistema

Procesas 1 Gija 1

Procesas 2 Gija 1

Gija 2

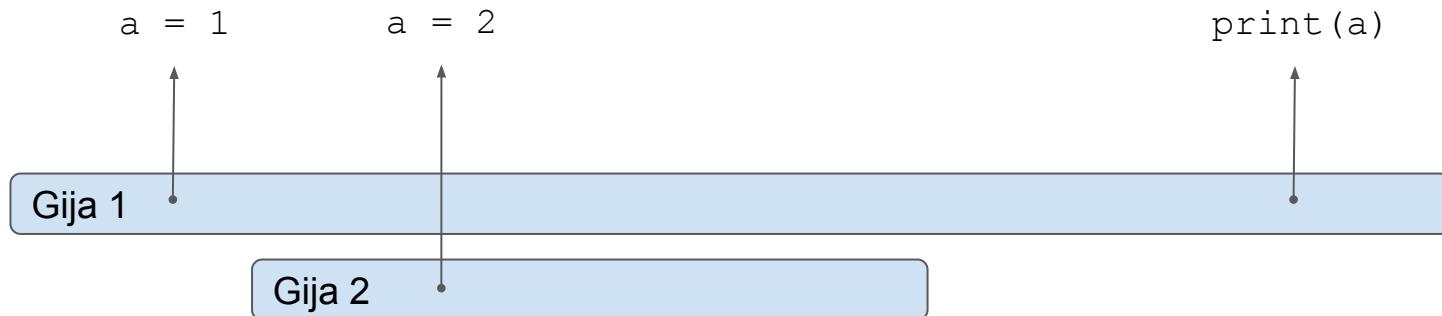
Gija 3

Gija 3

Gija 4

Kur gijos gali kelti problemų?

- Gija gali prieiti prie bendrų programos kintamujų
- Jei gija pakeičia bendro kintamojo reikšmę, o kita gija apie tai nieko nežino programa gali gražinti neprognozuojamą rezultatą (race condition)

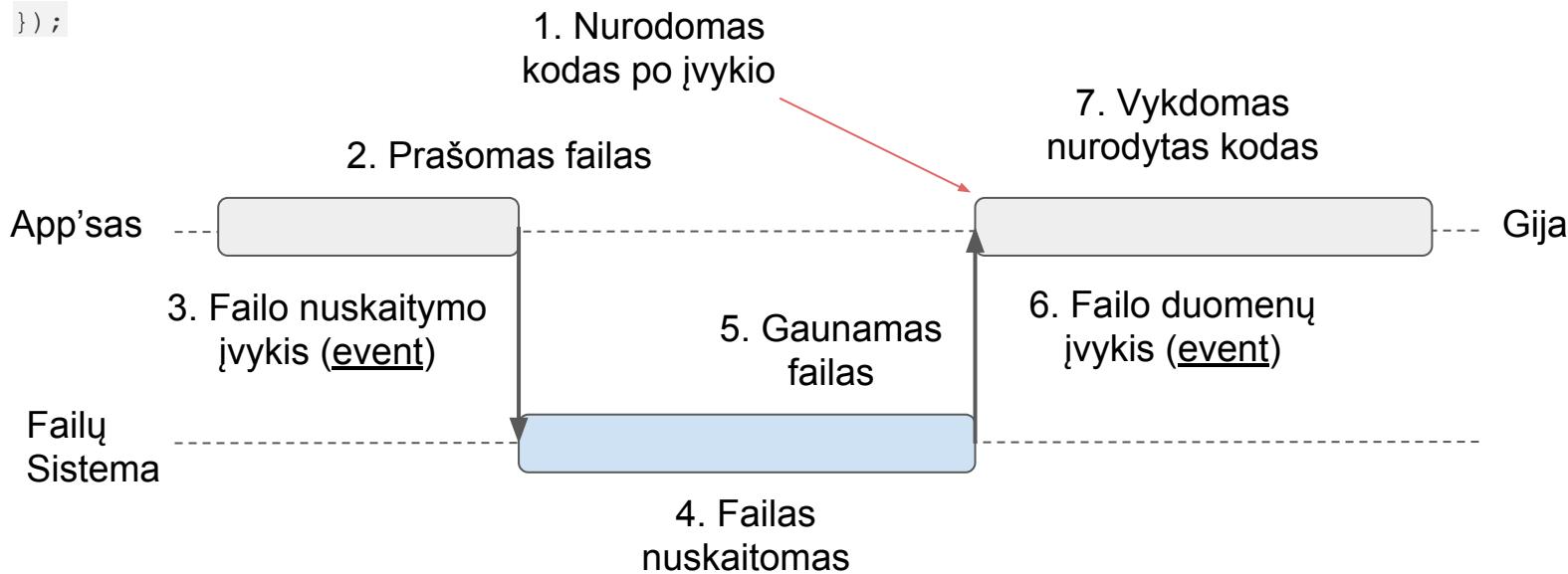


Įvykiais-varomas (event-driven) programavimas

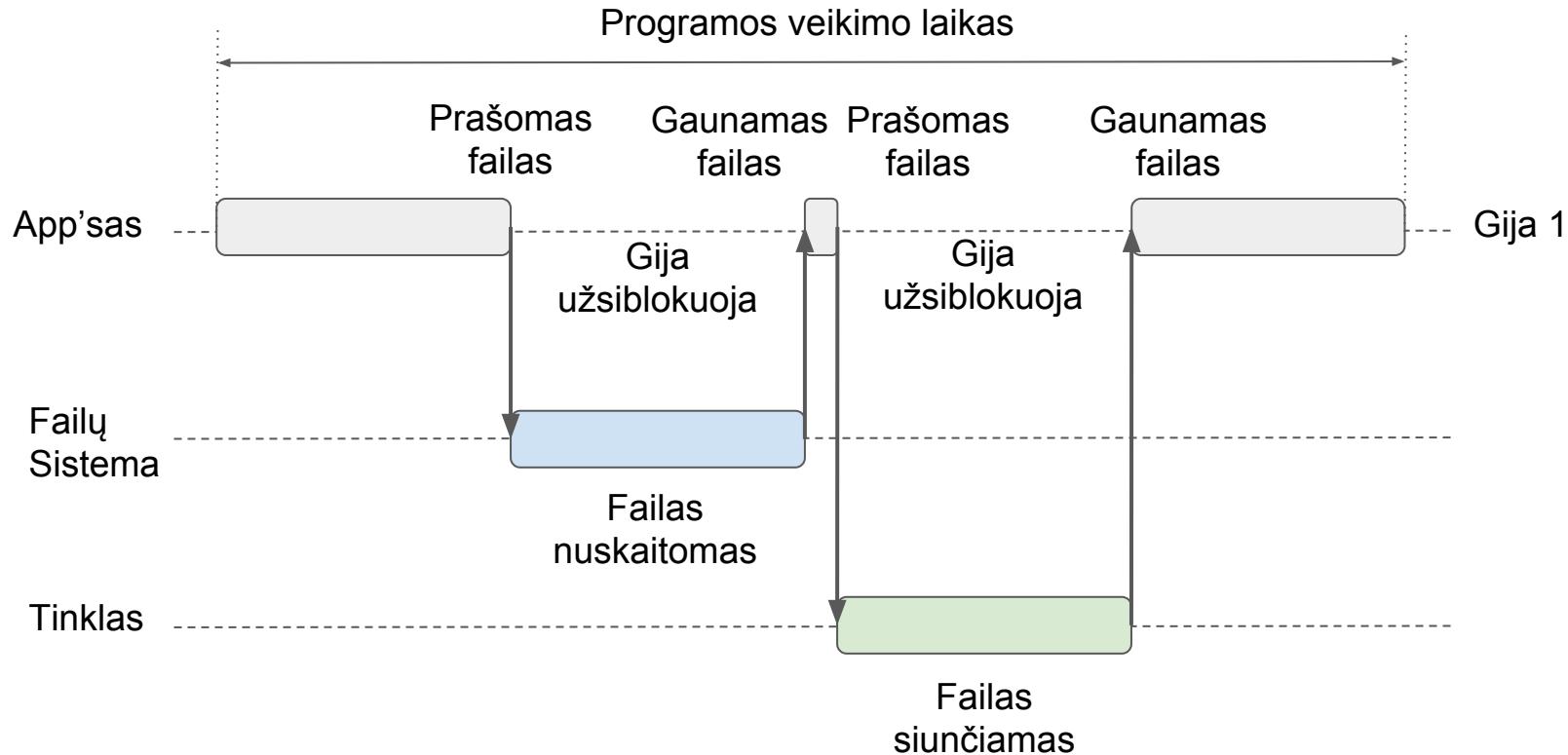
- Programavimo paradigma, kurios metu programos vykdymo seka yra nusprendžiama įvykių, pvz.:
 - Klaviatūros, pelės mygtukų paspaudimo įvykių
 - Laikrodžio tikslėjimo įvykių
 - Web užklausos atsakymo gavimo įvykių
 - Failo nuskaityto turinio gavimo įvykių

Įvykiai-varomas (event-driven) programavimas

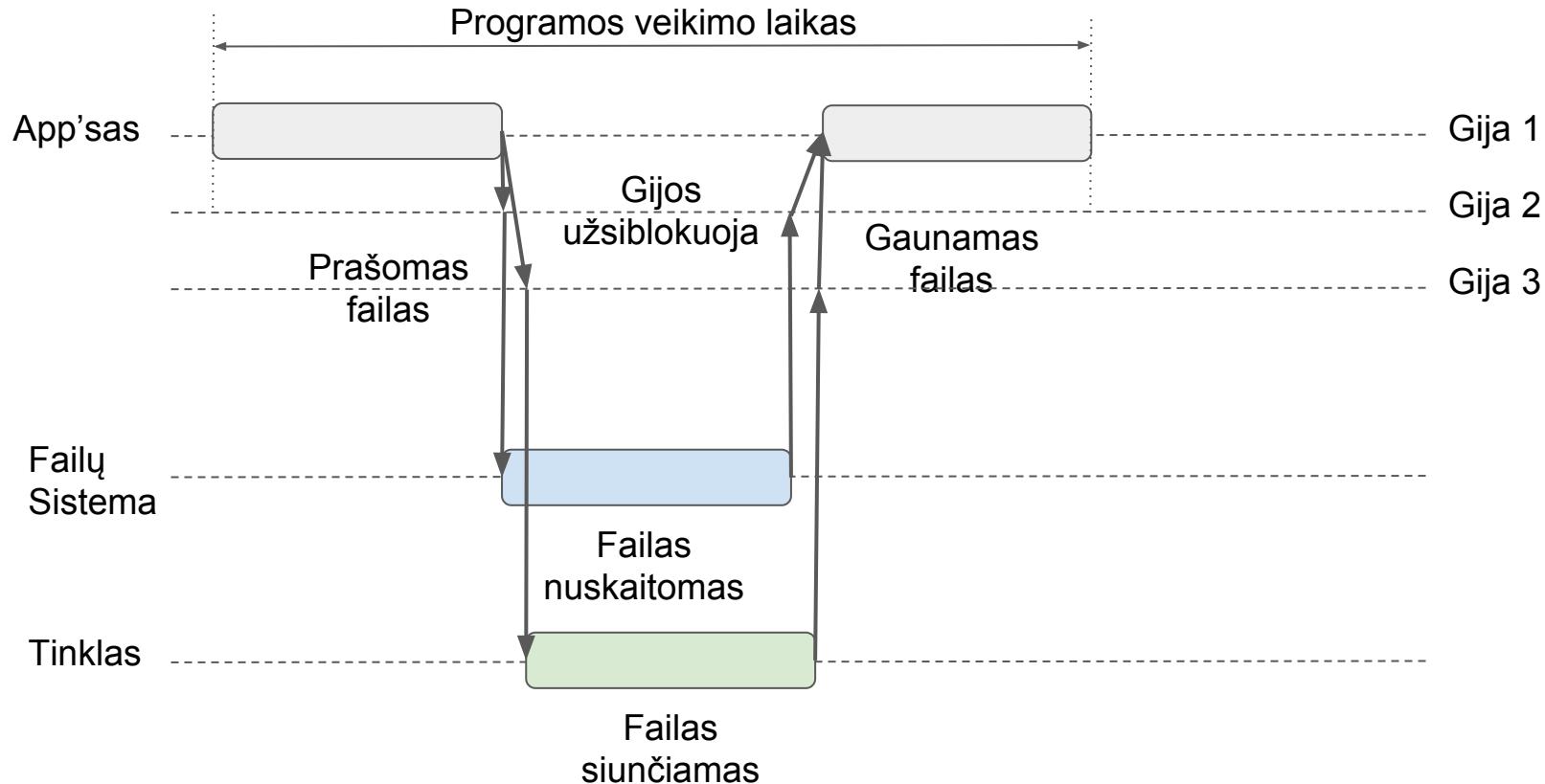
```
fs.readFile('/etc/passwd', (err, data) => {
  if (err) throw err;
  console.log(data);
});
```



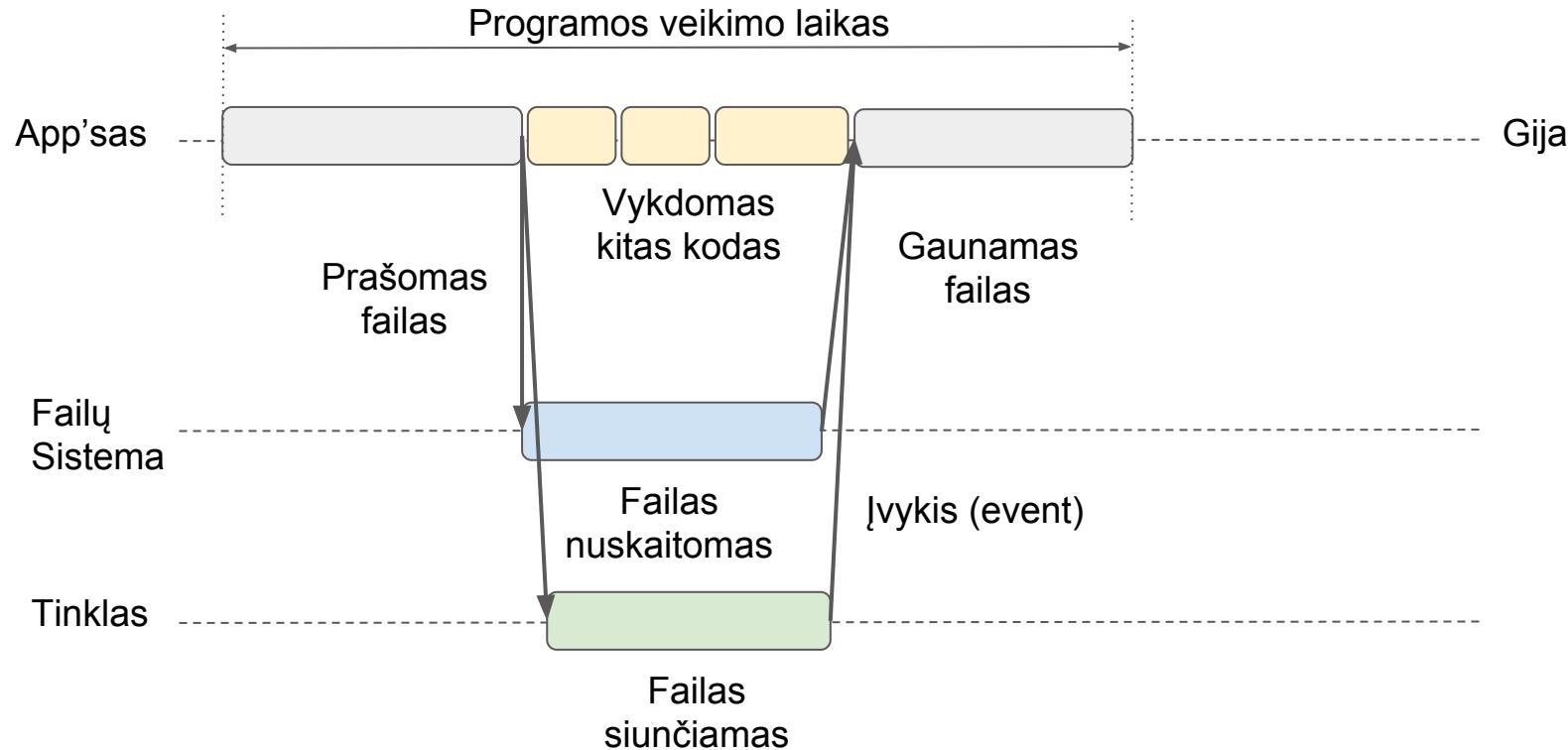
Blocking Input/Output (IO)



Blocking Input/Output (IO)



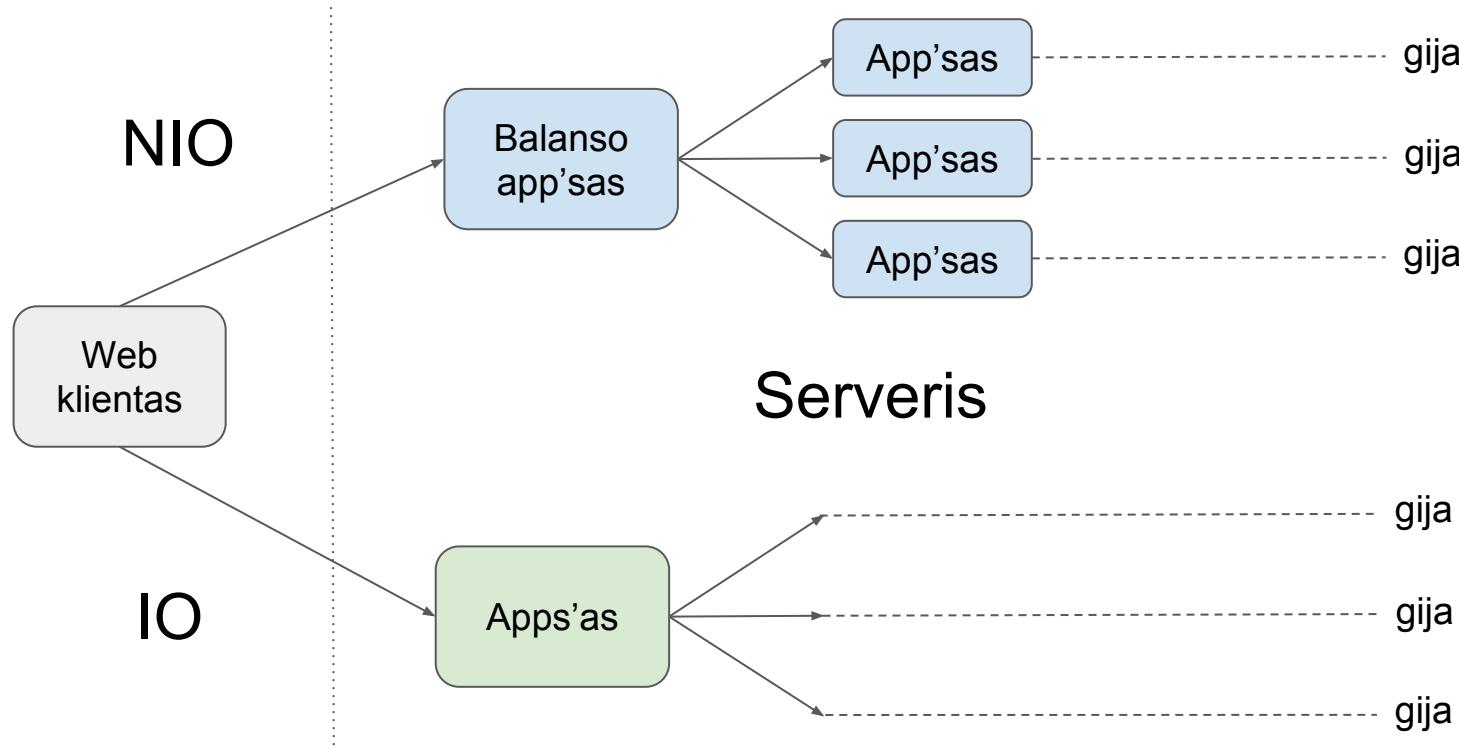
Non-blocking Input/Output (NIO)



IO prieš NIO

- Naudojant NIO, gijų valdymu pasirūpina NIO karkasas, todėl programuotojui atrodo, kad jo programa veikia paraleliai, bet nereikia rūpintis, kad daugiau nei viena gija vienu metu leis tą pačią kodo vietą
 - IO atveju, norint atlikti veiksmus paraleliai, reikia patiems pasirūpinti gijų valdymu
- Naudojant NIO, galima atlikti daugiau veiksmų, nes gijos panaudojamos tik įvesčiai/išvesčiai: reikia mažiau gijų: naudojama mažiau atminties
 - IO atveju gijos naudojamos ne tik su įvestimi/išvestimi (IO) susijusiam kodui leisti
- Naudojant IO, paleidus vieną programą, galimą išnaudoti visus procesorius (cores), nes skirtinges gijos gali naudoti skirtinges procesorius
 - NIO atveju reiktų paleisti tiek programų, kiek kompiuteryje yra procesorių (cores), o užklausoms į programas balansuoti reikalinga dar viena programa (load balancing)

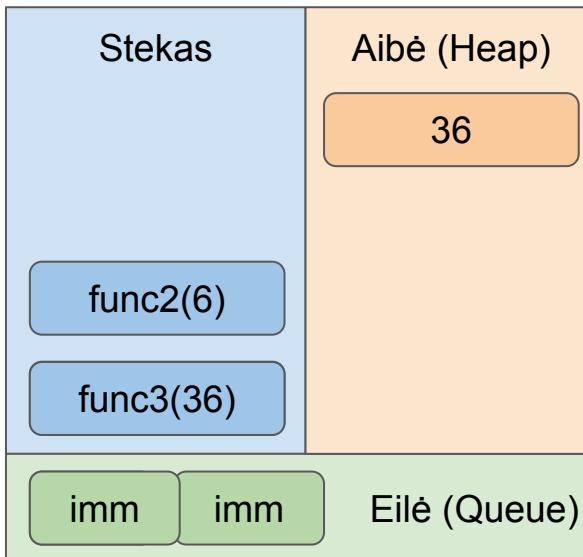
IO prieš NIO web kontekste



JavaScript “variklis” (engine)

- Programos JavaScript kodą leidžia proceso vienoje gijoje (thread)
 - Įvesties/išvesties kodą leidžia atskirose gijoje (izoliuota nuo programuotojo veiksmų)
- Naudoja NIO (non-blocking input/output)
 - Programuotojui atrodo, kad jo kodas leidžiamas paraleliai, tačiau nereikia rūpintis, kad daugiau nei viena gija vykdys tą pačią kodo vietą
- Naudoja įvykiais-varomą (event-driven) paremtą programavimą
 - Apie baigtą vykdyti įvesties/išvesties veiksma yra informuojama įvykiu, kuris iniciuoja nurodyto programos kodo vykdymą

JavaScript “variklis” (engine)



The screenshot shows a code editor window with the following code in `app.js`:

```
app.js  x
1 const func3 = (b) => {
2   console.log(b);
3 }
4
5 const func2 = (a) => {
6   return a * a;
7 };
8
9 const func1 = (a) => {
10  const b = func2(a + a);
11  setImmediate(func3, b);
12 };
13
14 func1(3);
15 |
```

The code defines three functions: `func1`, `func2`, and `func3`. `func1` calls `func2` with an argument of `a + a`, and then uses `setImmediate` to call `func3` with the result of `func2`. Finally, it calls `func1` with the argument `3`. The code editor interface includes tabs, a search bar, and a file menu.

JavaScript “varikliai” (engines)



- **V8** - sukompiliuoja JS kodą į mašininį kodą ir tada leidžia

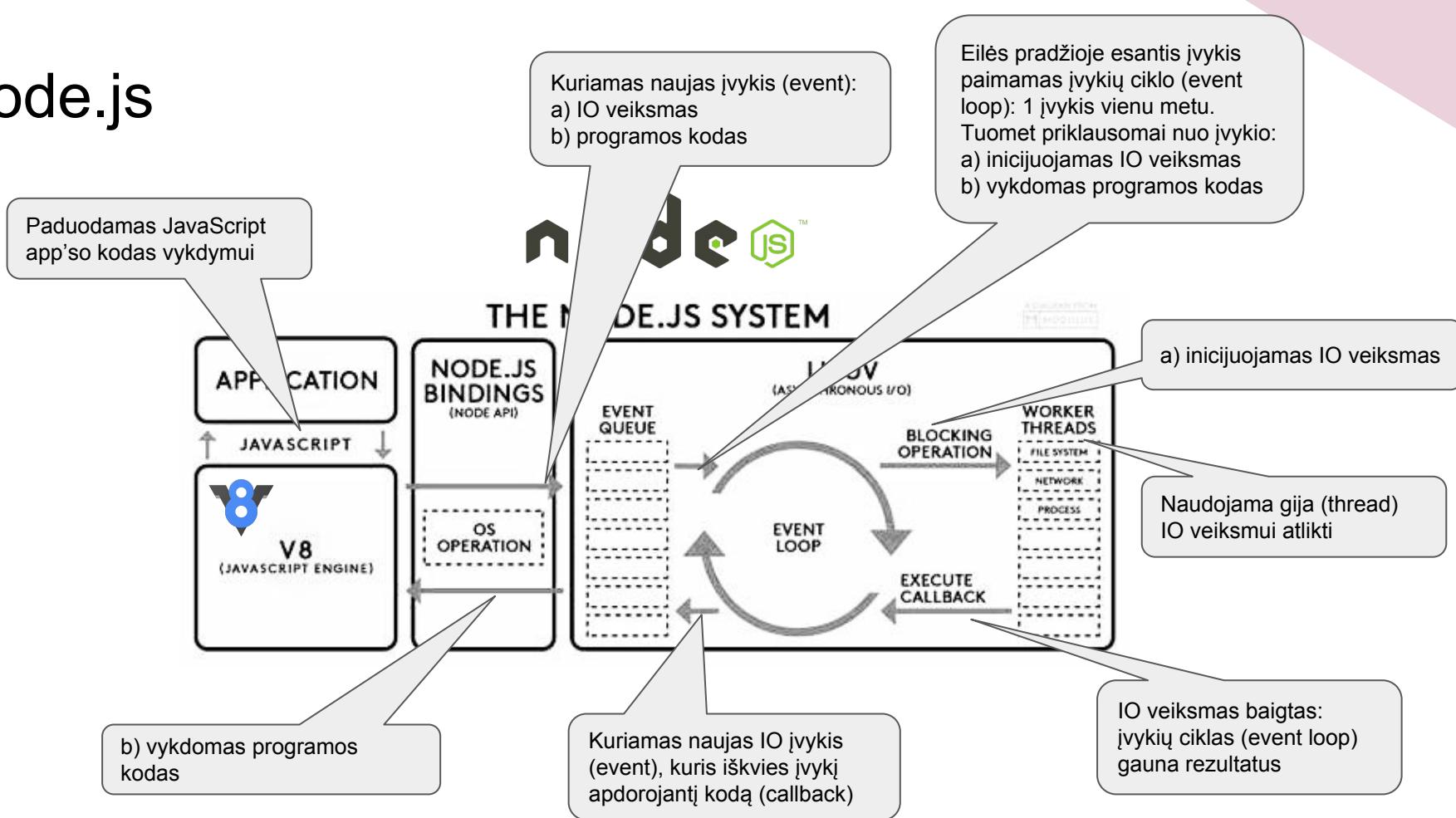


- **SpiderMonkey** - sukompiliuoja JS kodą į baitų kodą (byte code) ir tada jį interpretuoja



- **Rhino / Nashorn** - sukompiliuoja JS kodą į Java baitų kodą (byte code) ir tada leidžia su Java Virtualia Mašina (JVM)

Node.js



Node.js

- Paleidžia dabartinėje direktoriijoje esantį "app.js" failą su JS kodu:
 - node app

```
mykolas@mykolas ~/projects/vgtu/http-client $ node app
1. Connected to Server
2. Sending request message
3. Received:
HTTP/1.1 301 Moved Permanently
Location: http://www.google.lt/
Content-Type: text/html; charset=UTF-8
Date: Wed, 01 Mar 2017 11:54:16 GMT
Expires: Fri, 31 Mar 2017 11:54:16 GMT
Cache-Control: public, max-age=2592000
Server: gws
Content-Length: 218
X-XSS-Protection: 1; mode=block
X-Frame-Options: SAMEORIGIN

<HTML><HEAD><meta http-equiv="content-type" content="text/html;charset=utf-8">
<TITLE>301 Moved</TITLE></HEAD><BODY>
<H1>301 Moved</H1>
The document has moved
<A HREF="http://www.google.lt/">here</A>.
</BODY></HTML>
```

Java vs JavaScript

app.java

```
1 public class HelloWorld {  
2  
3     public static void main(String[] args) {  
4         System.out.println("Hello, World");  
5     }  
6  
7 }  
8
```

app.js

```
1 console.log("Hello, World");  
2 |
```

Java vs JavaScript

app.java

```
1  public class HelloWorld extends Object {  
2  
3      private String name;  
4  
5      public HelloWorld(String name) {  
6          this.name = name;  
7      }  
8  
9      public void printMyName() {  
10         System.out.println(name);  
11     }  
12  
13 }  
14
```

app.js

```
1  class HelloWorld extends Object {  
2  
3      constructor(name) {  
4          this.name = name;  
5      }  
6  
7      printMyName() {  
8          console.log(this.name);  
9      }  
10  
11 }  
12
```

ECMAScript 6 (2015)

- ECMAScript 5 (versija prieš ECMAScript 6) palaiko visos modernios naršyklės - ši versija dar vadinama “JavaScript asembleris”
- ECMAScript 6 standarto versija vis dar nėra palaikoma visų naršyklių
- ECMAScript 6 turi labai reikalingus kalbos patobulinimus, kurie palengvina jos naudojimą
- Naudosime Babel kompiliatorių, kuris ES 6 kodą sukompiliuos į ES 5 kodą, kurį supras visos modernio naršyklės
- Serverio pusėje naudosime tik tas ES 6 ypatybes, kurias palaiko mūsų naudojama Node.js versija

Kintamieji ir konstantos

```
1 let a = 0;          1 {           1 // camel case
2 const b = 1;        2   let a = 0;    2 let myName = 'Mykolas';
3 var c = 2;          3   const b = 1;  3 // snake case
                           4   var c = 2;    4 let my_name = 'Mykolas';
                           5   console.log(a); 5 const MY_NAME = 'Mykolas';
                           6   console.log(b);
                           7   console.log(c);
                           8 }
                           9   console.log(c);
                           10  console.log(b);
                           11  console.log(a);
```

Primityvus tipai

- Number - apvalinimas pagal IEEE-764 standartą

```
1 123
2 987.65
3 0.2 + 0.1 // 0.3000000000000004
```

- String - tarp " arba ` arba `` kabučių (UTF-16)

```
1 'Here I am'
2 "Štai ir aš"
3 `Mano vardas ${name}` // Mano vardas Mykolas
```

- Boolean - true arba false

- Null - tik null

- rekomenduojama naudoti taikomajame kode

- Undefined - tik undefined

- rekomenduojama nenaudoti taikomajame kode

- Symbol

Objektai ir masyvai

- Object

```
1 const obj = {  
2   name: 'Mykolas',  
3   age: 30  
4 }
```

```
1   const obj = {  
2     name: 'Mykolas',  
3     age: 30,  
4     address: {  
5       house: 10,  
6       street: 'Vilniaus g.',  
7       city: 'Vilnius'  
8     }  
9   }
```

- Array (masyvas) - specialus objekto tipas

```
1 const arr = [1, 2, 11];
```

```
1   const arr = [  
2     1,  
3     2,  
4     11,  
5     'Skaičius',  
6     true,  
7     null,  
8     { name: 'Mykolas'}  
9   ];
```

```
1 const obj = {} // new Object();  
2 obj.name = 'Mykolas';  
3 obj.age = 30;  
4 obj['address'] = {  
5   house: 10,  
6   street: 'Vilniaus g.',  
7   city: 'Vilnius'  
8 };
```

```
1 const arr = [] // new Array();  
2 arr[0] = 1;  
3 arr[1] = 2;  
4 arr[10] = 11;  
5 arr[11] = 'Skaičius';  
6 arr[12] = true;  
7 arr[13] = null;  
8 arr[14] = {  
9   name: 'Mykolas'  
10 };
```

Duomenų tipų konvertavimas

```
1 const result1 = 3 + '30'; // Number paverčiamas String
2 const result2 = 3 * '30'; // String paverčiamas Number
3 const integer = parseInt('34 asd', 10); // ne skaičiai ignoruojami
4 const float = parseFloat('1.234 asd'); // ne skaičiai ignoruojami
5 const string = 524.987.toString();
6 const boolean1 = !!'labas'; // true
7 const boolean2 = !!''; // false
8 const boolean3 = !!{}; // true
```

Ciklai - while, do, for

```
1 let a = 0;  
2 while (a < 10) {  
3     a++;  
4 }
```

```
1 for (let a = 0; a < 10; a++) {  
2     a++;  
3 }
```

```
1 let a = 0;  
2 do {  
3     a++;  
4 } while (a < 10);
```

```
1 const a = [1, 2, 3];  
2 for (let i of a) {  
3     console.log(i);  
4 }
```

```
1 const a = { a: 1, b: 2, c: 3};  
2 for (let i in a) {  
3     console.log(i);  
4 }
```

```
1 const a = { a: 1, b: 2, c: 3};  
2 for (let i in a) {  
3     if (i === 'a') {  
4         continue;  
5     }  
6     console.log(i);  
7 }
```

```
1 const a = { a: 1, b: 2, c: 3};  
2 for (let i in a) {  
3     if (i === 'a') {  
4         break;  
5     }  
6     console.log(i);  
7 }
```

Sąlygos: if, switch

```
1  const a = 0;          1  const a = 0;          1  const a = 0;
2  if (a === 0) {        2  if (a === 0) {        2  switch (a) {
3    console.log(a);    3    console.log(a);    3    case 0:
4  }                    4  } else {           4    console.log(0);
5                      5    console.log('not 0'); 5    break;
6                      6  }                         6  case 1:
7                                7  case 2:           7    console.log(1);
8                                8  case 3:           8    break;
9                                9  default:         9    console.log('2 or 3');
10                           10  console.log('2 or 3');10  break;
11                           11  default:         11  console.log('> 3');
12                           12  }               12  }
```

Tipų interpretacija sąlygoje

```
1 true //true  
2 false // false  
3 undefined // false  
4 null // false  
5 "" // false  
6 '' // false  
7 `` // false  
8 'a' // true  
9 ' ' //true  
10 'tekstas' ' // true  
11 0 // false  
12 0.0 // false  
13 1 // true  
14 234 // true  
15 1232.34 // true  
16 {} // true  
17 [] // true
```

```
1 if (result) {  
2  
3 }
```

Tik šios reikšmės yra "false" (visa kita "true"):

undefined

null

false

0

NaN // (Not a Number)

'' '' '' // (tuščias String)

Išraiškos (expressions) ir operatoriai

```
1  3 + 2 // 5
2  3 - 2 // 1
3  3 / 2 // 1.5
4  3 * 2 // 6
5  3 % 2 // 1
6  -'3' // -3
7  +'3' // 3
8  ++i // padidina vienetu ir grąžina
9  i++ // grąžina ir padidina vienetu
10 --i // sumažina vienetu ir grąžina
11 i-- // grąžina ir sumažina vienetu
12
13 2 + 3 * 4 // 14, o ne 20
```

```
1  1 === 1 //true
2  1 == '1' // true
3  1 !== 1 // false
4  1 != '1' // false
5  1 < 1 // false
6  1 <= 1 // true
7  1 > 1 // false
8  1 >= 1 // false
9  1 && 1 // true
10 1 || 1 // true
```

Išraiškos (expressions) ir operatoriai

```
1  typeof undefined // 'undefined'  
2  typeof null // 'object'  
3  typeof {} // 'object'  
4  typeof [] // 'object'  
5  typeof true // 'boolean'  
6  typeof 1 // 'number'  
7  typeof 1.23 // 'number'  
8  typeof '' // 'string'  
9  typeof Symbol() // 'symbol'  
10 typeof function() // 'function'
```

```
1  const person = {  
2      name: 'Mykolas'  
3  };  
4  console.log(person); // { name: 'Mykolas' }  
5  delete person.name;  
6  console.log(person); // {}
```

Objektinė kalba

- Enkapsuliacija (Encapsulation)
- Paveldėjimas (Inheritance)
- Virtualūs metodai (Virtual methods)

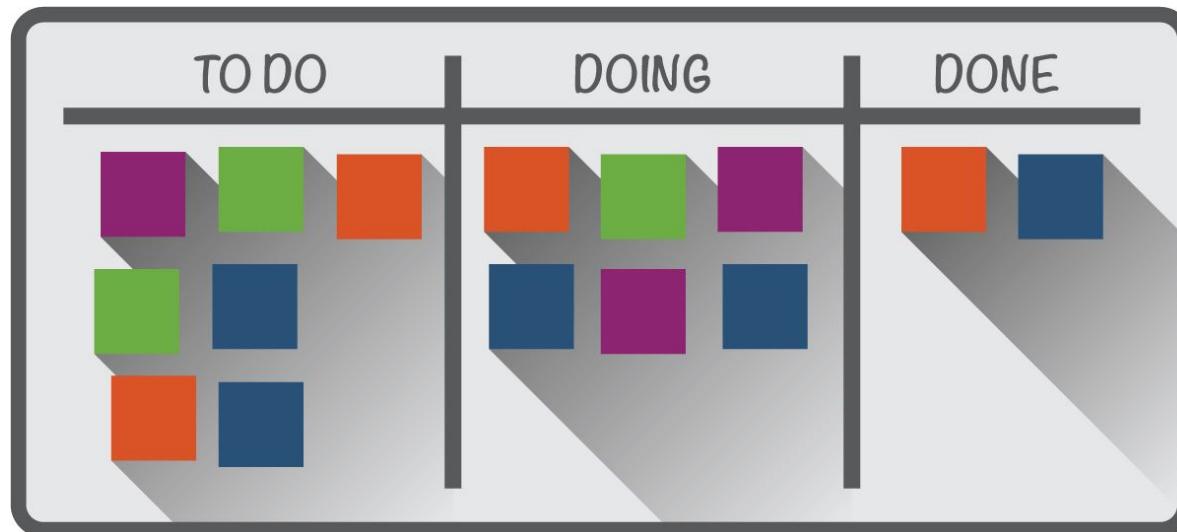
Funkcinė kalba

- Grynos funkcijos (Pure functions)
- Nikintamumas (Immutability)
- Neturinti būsenos (Stateless)
- Deklaravyti (Declarative)
- Aukštesnės paskirties funkcijos (Higher order functions)
- Rekursija (Recursion)

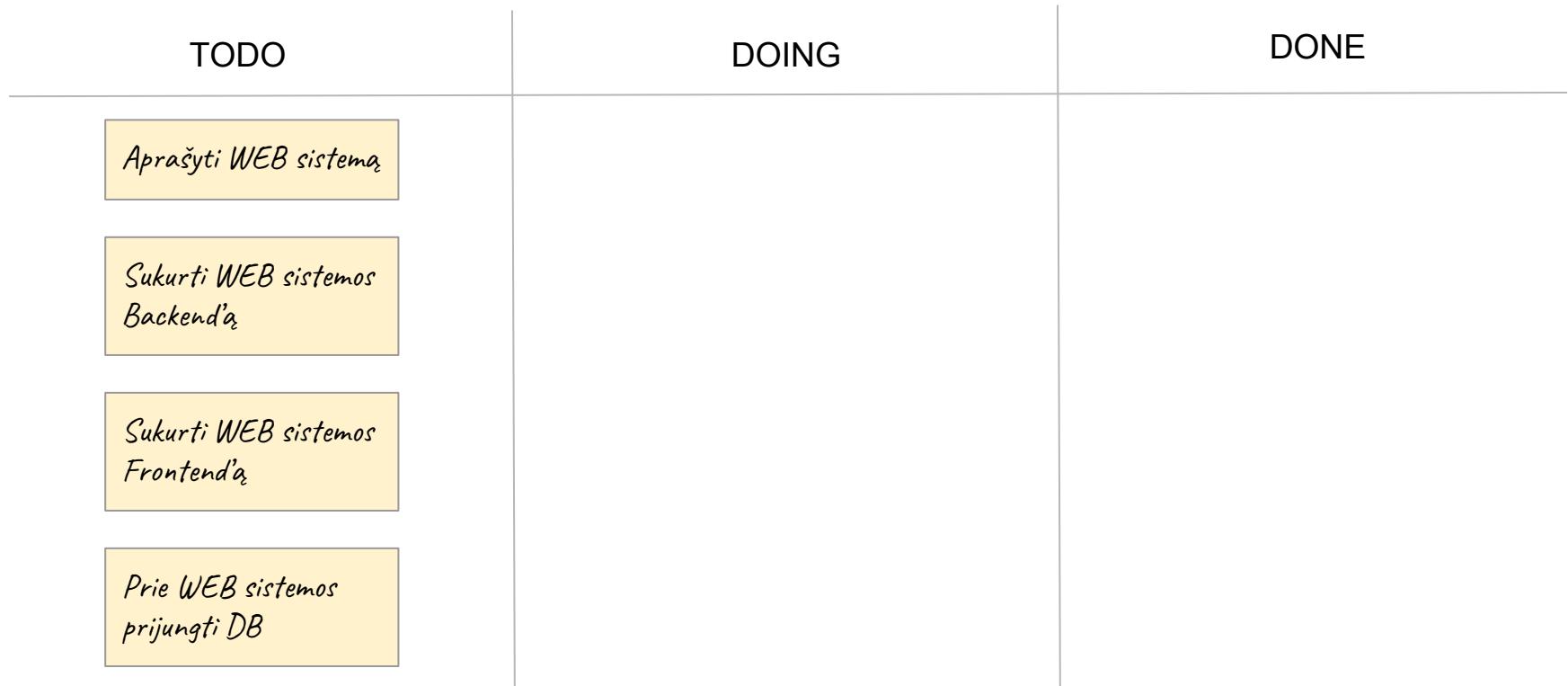


P2 - Sistemos kūrimo procesas

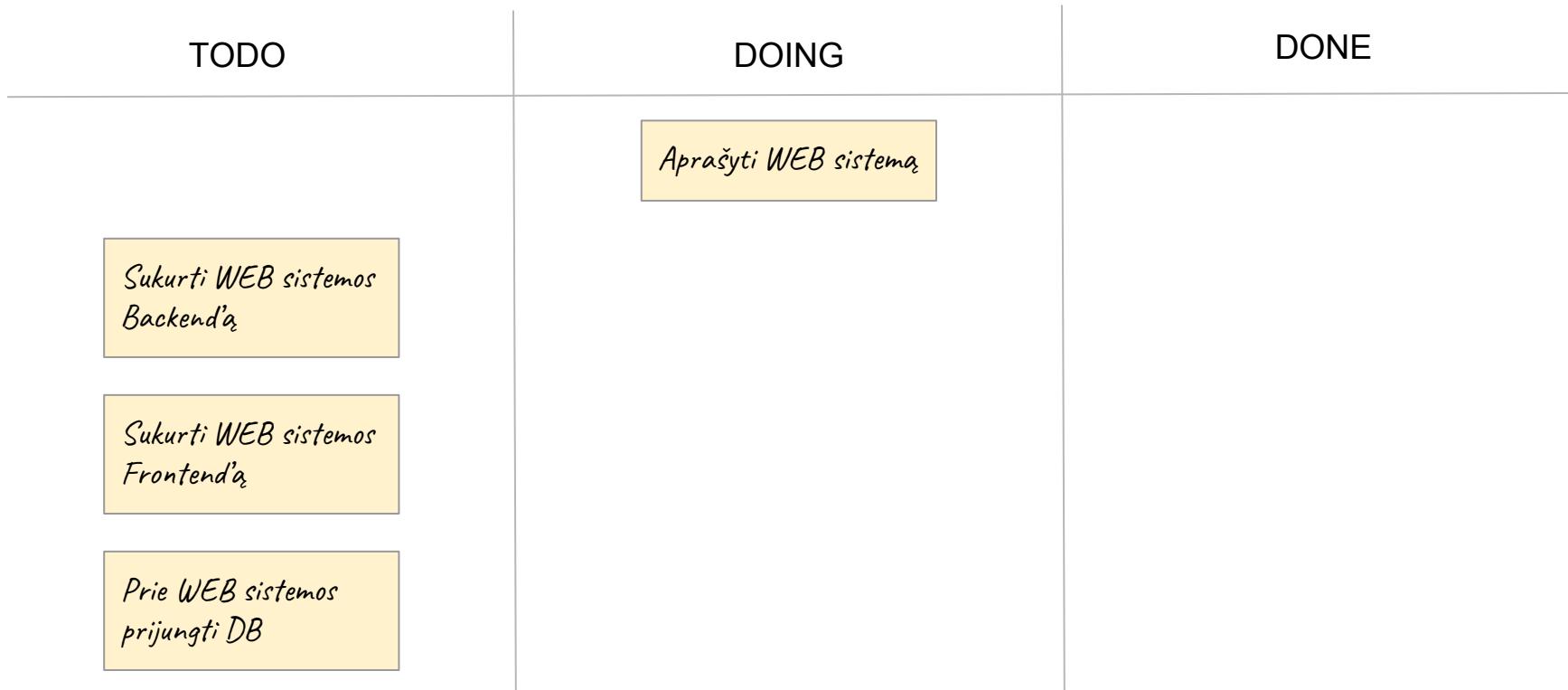
Sistemos kūrimo procesas - Agile - Kanban



‘Organizacijos’ Kanban



‘Organizacijos’ Kanban



‘Organizacijos’ Kanban



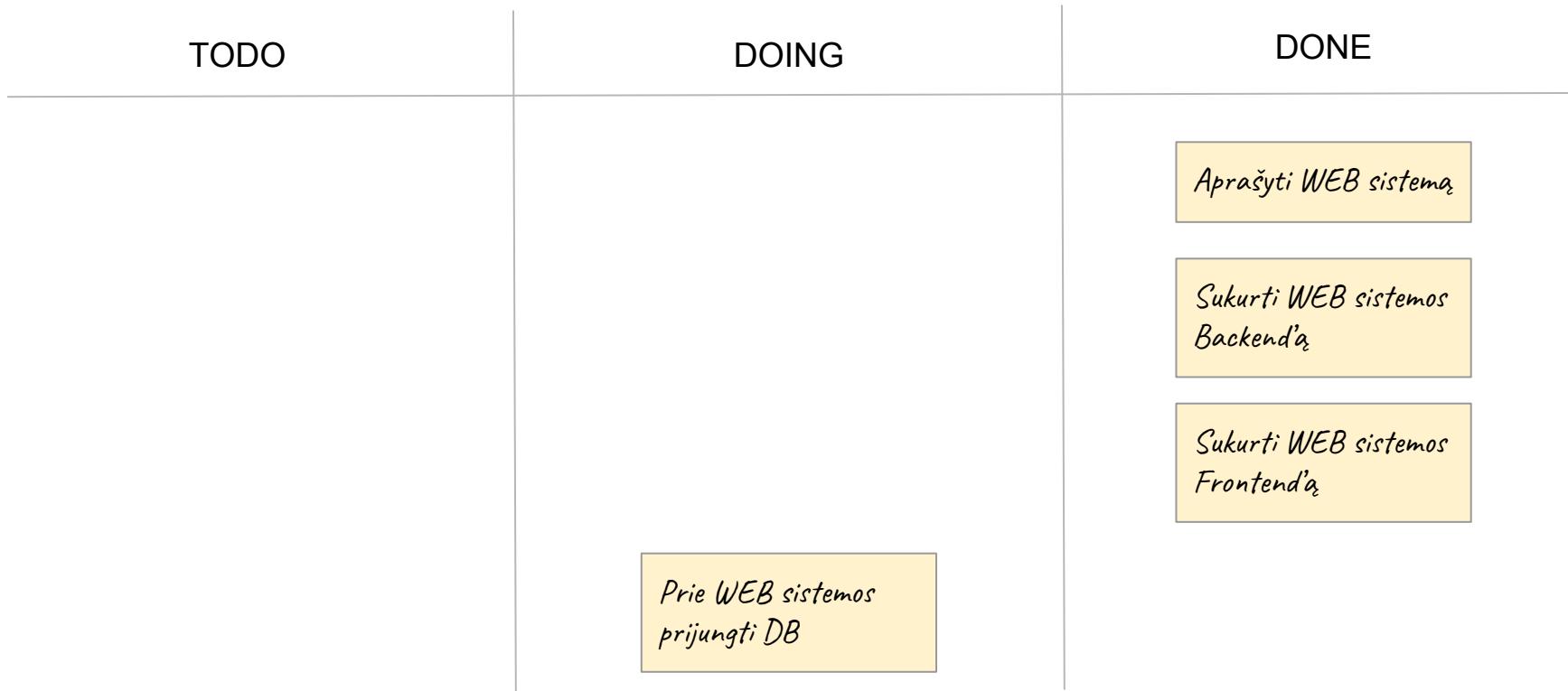
‘Organizacijos’ Kanban



‘Organizacijos’ Kanban



‘Organizacijos’ Kanban



‘Organizacijos’ Kanban

| TODO | DOING | DONE |
|--------------------|-------|---|
| Papildoma užduotis | | <p>Aprašyti WEB sistemos Backend'ą</p> <p>Sukurti WEB sistemos Frontend'ą</p> <p>Prie WEB sistemos prijungti DB</p> |

Agile proceso ir kodo valdymo paslauga - GitHub



- Kiekvienam 'organizacijos' nariui skirta:
 - Kodo rezervuarija (Code) - WEB sistemos kodas ir aprašymas (README.md)
 - Kanban lenta (board) (Projects)
 - Užduotys ir jų valdymas (Issues) - kanban lentos užduotys

Pirmaji užduotis

Sistemos aprašymas:

1. Gauti preieigą prie organizacijos repozitorijos
2. Susikurti projektą (naudoti “Kanban” šablona)
3. Susikurti pirmajį bilietą (issue):
 - a. Bilieto pavadinimas: “Describe web system”
 - b. Užduoties aprašymas <https://github.com/Mykolas-PO/web-system-template>
4. Kiekvieną pirmosios užduoties dalį aprašyti atskirame biliete (issue):
 - a. Entity requirements: “Add DB into the web system”
 - b. REST API requirements: “Build web system back-end”
 - c. UI requirements: “Build web system front-end”



T4 - Web backend'o pagrindai

Funkcijos

```
1  function calculateAgeInYears1(age, years) {  
2      const newAge = age + years;  
3      return newAge;  
4  }  
5  
6  const calculateAgeInYears2 = (age, years) => {  
7      const newAge = age + years;  
8      return newAge;  
9  }  
10  
11 let calculateAgeInYears3 = calculateAgeInYears1;  
12 calculateAgeInYears3 = calculateAgeInYears2;  
13  
14 const newAge1 = calculateAgeInYears3(15, 5); // 20  
15 const newAge2 = calculateAgeInYears3(15, 5, 'anything', true, 999); // 20  
16 const newAge3 = calculateAgeInYears3(15); // NaN  
17 const newAge4 = calculateAgeInYears3(); // NaN
```

Funkcijos (spread operator, default arguments)

```
1 const f = (...args) => {
2   console.log(args);
3 }
4
5 f('name', 35, true, {title: 'Till I Come'}); // [ 'name', 35, true, { title: 'Till I Come' } ]
```

```
1 const f = (name, age = 56) => {
2   console.log(name, age);
3 }
4
5 f('Mykolas', 23); // Mykolas 23
6 f('Mykolas'); // Mykolas 56
```

Funkcijos (destructuring)

```
1 const f1 = (audioTrack) => {
2     console.log(audioTrack.title, audioTrack.length);
3 }
4
5 const f2 = (audioTrack) => {
6     const title = audioTrack.title;
7     const length = audioTrack.length;
8     console.log(title, length);
9 }
10
11 const f3 = (audioTrack) => {
12     const {title, length} = audioTrack;
13     console.log(title, length);
14 }
15
16 const f4 = ({ title, length }) => {
17     console.log(title, length);
18 }
19
20 const autoTrack = {
21     title: 'Till I Come',
22     length: 125
23 }
24 f1(autoTrack); // Till I Come 125
25 f1({ title: 'Till I Come', length: 125 }); // Till I Come 125
```

Funkcijos (object property)

```
1 const obj1 = {  
2     name: 'Mykolas',  
3     printName() {  
4         console.log(`Your name is ${this.name}`);  
5     }  
6 }  
7  
8 obj1.printName(); // Your name is Mykolas  
9  
10 const obj2 = {  
11     name: 'Mykolas',  
12     printName: () => {  
13         console.log(`Your name is ${this.name}`);  
14     }  
15 }  
16  
17 obj2.printName(); // Your name is undefined
```

Apimtis (Scope)

```
1 // Global (module) scope
2 const a = 1;
3 let b = 2;
4
5 // Block scope
6 {
7     const c = 3;
8     console.log(c); // 3
9 }
10 console.log(c); // c is not defined
11
12 // Block scope
13 const f = () => {
14     const d = 3;
15     console.log(d); // 3
16 }
17 console.log(d); // d is not defined
```

Modulis (Module)

- Node.js kiekvieną programos failą laiko moduliu
 - Visos globalios deklaracijos (kintamieji ir funkcijos) iš tiesų yra pasiekiamos tik tame pačiame faile
 - Jei norima, kad deklaracijas (kintamuosius) pasiektų kitas failas, jas reikia eksportuoti (`module.export = ...`)
 - Jei norima deklaracijas (kintamuosius) kitame faile pasiekti, jas reikia importuoti arba 'reikalauti' (`require(...)`)

Modulis (Module)

app.js

module.js

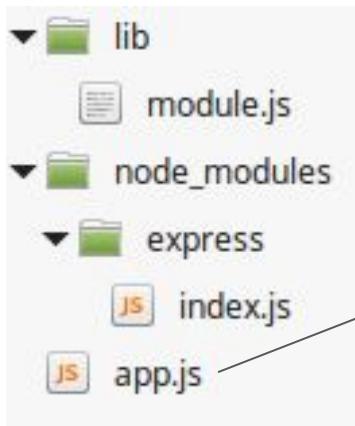
```
1 const myModule = require('./module');
2
3 const sum = myModule.calc(1, 2);
4 console.log(myModule.name, sum); // Mykolas 3
```

```
1 const name = 'Mykolas';
2
3 const calc = (a, b) => {
4     return a + b;
5 };
6
7 module.exports = {
8     name: name,
9     calc: calc
10};
```

Modulio tipai

| Tipas | Reikšmė paduodama į require funkciją | Pavyzdys |
|-------|--|--|
| Core | Neprasideda / ./ arba ../ | <code>require('fs')</code> <code>require('http')</code> |
| File | Prasideda / ./ arba ../ | <code>require('../module')</code> <code>require('.../../file')</code> <code>require('/var/log/a')</code> |
| npm | Nėra core tipas ir neprasideda / ./ arba ../ | <code>require('express')</code> <code>require('react')</code> |

Pavyzdys



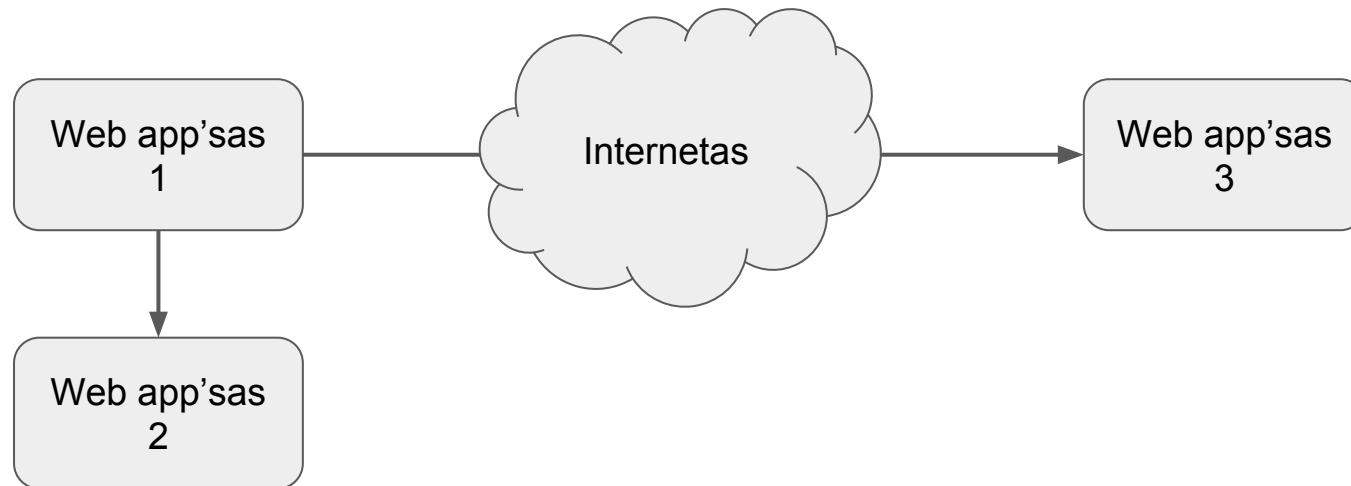
```
1 const http = require('http');
2 const express = require('express');
3 const myModule = require('./lib/module');
```

Web buvo sukurtas žmonėms

- Tiksliau: komuniuoti žmogui-su-aparatu (human-to-machine)
- HTML dokumentai yra skirti žmonėms
- HTML dokumentai nėra patogūs skaityti aparatams (machines)
- Reikalingas būdas komuniuoti apratui-su-aparatu (machine-to-machine)

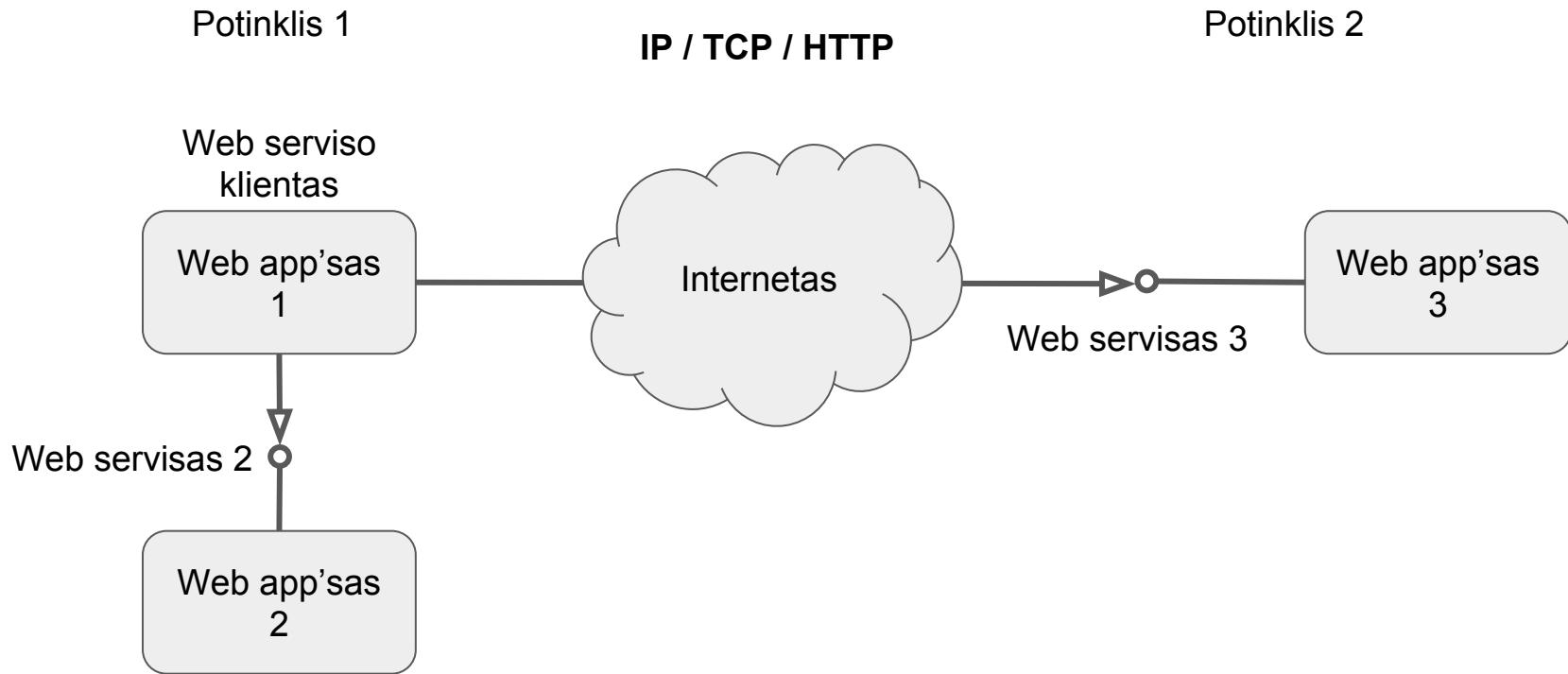
Komunikacija tarp app'sų

Potinklis 1



Potinklis 2

Web servisai (Web services)



SOAP

- Simple Object Access Protocol

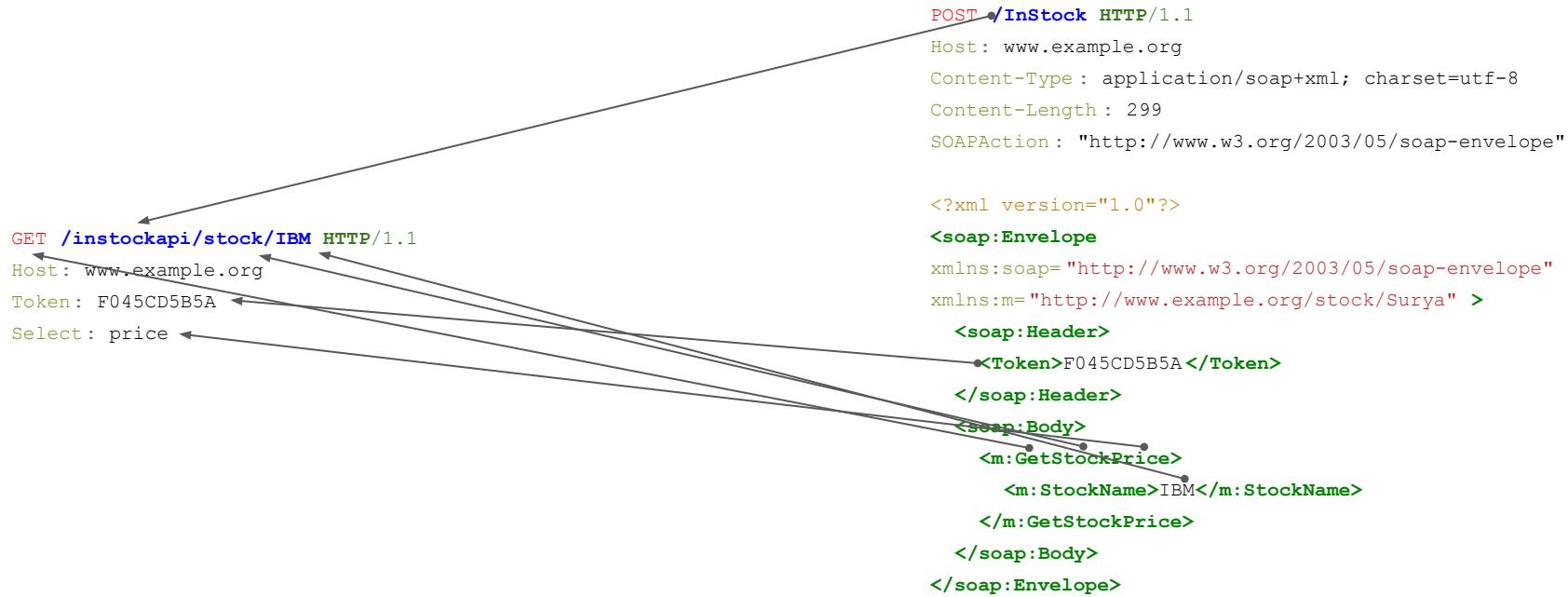
```
POST /InStock HTTP/1.1
Host: www.example.org
Content-Type: application/soap+xml; charset=utf-8
Content-Length: 299
SOAPAction: "http://www.w3.org/2003/05/soap-envelope"

<?xml version="1.0"?>
<soap:Envelope
    xmlns:soap= "http://www.w3.org/2003/05/soap-envelope"
    xmlns:m= "http://www.example.org/stock/Surya" >
    <soap:Header>
        <Token>F045CD5B5A</Token>
    </soap:Header>
    <soap:Body>
        <m:GetStockPrice>
            <m:StockName>IBM</m:StockName>
        </m:GetStockPrice>
    </soap:Body>
</soap:Envelope>
```

- Lėtas XML nagrinėjimas (parsing)
- Žinutėse daugiažodžiaujama (verbose)
 - Juk skirta aparatams, o ne žmonėms
- Praktiškai naudojamas tik POST HTTP metodas
- Praktiškai nenaudojamas URI
- Praktiškai nenaudojamos HTTP antraštės (headers)

REST

Representational state transfer (REST) arba RESTful Web services



REST

Representational state transfer (REST) arba RESTful Web services

HTTP/1.1 200 OK
Content-Type : application/json; charset=utf-8
Content-Length : 21
Timestamp : 2016-01-01T00:01:20.456

```
{  
  "price": 25400  
}
```

HTTP/1.1 200 OK
Content-Type : application/xml; charset=utf-8
Content-Length : 121
Timestamp : 2016-01-01T00:01:20.456

```
<?xml version="1.0"?>  
<StockPrice xmlns="http://www.example.org/stock/Surya"  
    <Price>25400</Price>  
</StockPrice>
```

HTTP/1.1 200 OK
Content-Type : application/xml; charset=utf-8
Content-Length : 299

```
<?xml version="1.0"?>  
<soap:Envelope  
    xmlns:soap="http://www.w3.org/2003/05/soap-envelope"  
    xmlns:m="http://www.example.org/stock/Surya" >  
    <soap:Header>  
        <Timestamp>2016-01-01T00:01:20.456 </Timestamp>  
    </soap:Header>  
    <soap:Body>  
        <m:StockPrice>  
            <m:Price>25400</m:Price>  
        </m:StockPrice>  
    </soap:Body>  
</soap:Envelope>
```

JSON

JavaScript Object Notation

```
1  {
2      "id": 123,
3      "name": "Mykolas",
4      "address": {
5          "street": "Vilniaus g.",
6          "house": 1
7      },
8      "hasCat": true,
9      "devices": [
10         {
11             "name": "iPhone 6"
12         },
13         {
14             "name": "Dell Latitude"
15         }
16     ]
17 }
```

```
1  const user = {
2      id: 123,
3      name: 'Mykolas',
4      address: {
5          street: 'Vilniaus g.',
6          house: 1
7      },
8      hasCat: true,
9      devices: [
10         {
11             name: 'iPhone 6'
12         },
13         {
14             name: 'Dell Latitude'
15         }
16     ]
17 }
```

REST (metodai ir URI)

- Skirtingas HTTP metodas - skirtingas veiksmas su resursu
- Resursas yra daiktavardis
- Organizuotas REST yra vadinamas REST API (App programming interface)
 - Pavyzdys su resursu "user" (naudotojas):
 - POST /user - naujam naudotojui sukurti
 - GET /user - gauti visus naudotojus
 - GET /user/:id - gauti naudotoją pagal jo identifikatorių
 - PUT /user/:id arba PATCH /user/:id - redaguoti egzistuojantį naudotoją
 - DELETE /user/:id - ištrinti naudotoją pagal jo identifikatorių

Express metodai ir URI

```
1 const express = require('express');
2
3 const app = express();
4
5 app.post('/user', (req, res) => {
6     // sukuriamas naujas naudotojas
7 });
8
9 app.get('/user', (req, res) => {
10    // surandami visi naudotojai
11 });
12
13 app.get('/user/:id', (req, res) => {
14    // surandamas naudotojos, kurio ID yra req.params.id
15 });
16
17 app.put('/user/:id', (req, res) => { // arba patch
18    // redaguojamas naudotojas, kurio ID yra req.params.id
19 });
20
21 app.delete('/user/:id', (req, res) => { // arba patch
22    // ištrinamas naudotojas, kurio ID yra req.params.id
23 });
24
25 app.listen(3000);
```

```
1 const http = require('http');
2
3 const server = http.createServer((req, res) => {
4     switch(req.method) {
5         case 'GET':
6             // aprodoti 2 skirtinges GET atvejus
7             break;
8         case 'POST':
9             // apdoroti POST uzklausą
10            break;
11        // ...
12    }
13 });
14
15 server.listen(3003, '127.0.0.1');
```

REST (atsakymo statusai ir kūnai (bodies))

- 2xx - viskas gerai:
 - POST /<resursas> atveju grąžinti sukurtą resursą
 - GET /<resursas> atveju grąžinti resursų sąrašą (pvz.: JSON array)
 - GET /<resursas>/:id atveju grąžinti prašomą resursą pagal jo identifikatorių
 - PUT /<resursas>/:id arba PATCH /<resursas>/id atveju grąžinti pakeistą resursą
 - DELETE /<resursas>/:id atveju grąžinti ištrintą resursą
- 3xx - resursas buvo perkeltas
- 4xx - kliento klaida, grąžinti klaidos tekštą
- 5xx - serverio klaida, grąžinti klaidos tekštą

Express statusai ir kūnai (bodies)

```
1 const express = require('express');
2 const userRepository = require('./user/userRepository');
3
4 const app = express();
5
6 app.get('/user/:id', (req, res) => {
7     userRepository.findById(req.params.id, (error, user) => {
8         if (error) {
9             console.error(error);
10            res.status(500).send({
11                message: 'Internal error. Please try again.'
12            });
13            return;
14        }
15        if (!user) { // user === null || user === undefined
16            res.status(400).send({
17                message: 'User is not found'
18            });
19            return;
20        }
21        res.status(200).send(user);
22    });
23 });
24
25 app.listen(3000);
```

REST (antraštė)

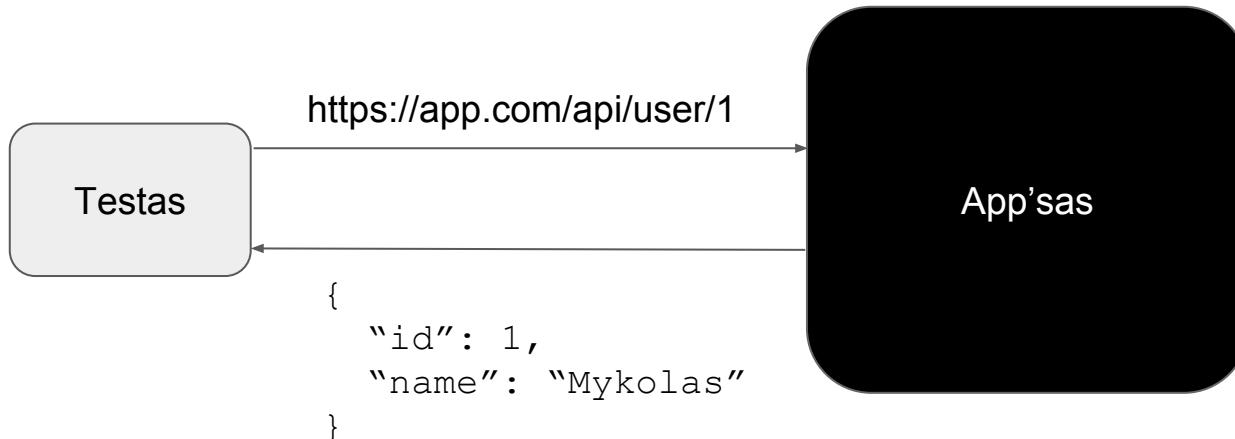
- Puslapiavimas (pagination):
 - `start: 30` grąžinti nuo 30 resurso
 - `count: 15` grąžinti 15 resursų
- Autentifikacija (authentication):
 - `token: A2FCD65BC88` pateikti prisijungimo metu išduotą žetoną (token)
- Filtravimas
 - `filter: name=Mykolas&age=26` pateikti filtro sąlygas

Express antraštēs (headers)

```
1 const express = require('express');
2 const userRepository = require('./user/userRepository');
3
4 const app = express();
5
6 app.get('/user', (req, res) => {
7     const start = req.get('start') ? parseInt(req.get('start'), 10) : 0;
8     const count = req.get('count') ? parseInt(req.get('count'), 10) : 10;
9     userRepository.findByStartAndCount(start, count, (error, users) => {
10         if (error) {
11             console.error(error);
12             res.status(500).send({
13                 message: 'Internal error. Please try again.'
14             });
15             return;
16         }
17         res.status(200).set('total', users.length).send(users);
18     });
19 });
20
21 app.listen(3000);
```

Juodos dėžės (Black-Box) testavimas

- Tai tokis testavimo metodas, kuriame app'so funkcionalumas yra tikrinamas be žinių, kaip tai veikia app'so viduje.



Testavimas su mocha ir supertest

```
const request = require('supertest')

describe('GET /user/:id', () => {
  it('returns a user', () => {
    return request(app)
      .get('/user')
      .set('Accept', 'application/json')
      .expect(200, {
        id: '1',
        name: 'John Math'
      });
  });
});
```

Dokumentacija

- Kiekvienas REST API turėtų būti dokumentuotas
 - Koks HTTP metodas
 - Koks URI
 - Kokie URI parametrai (pvz.: “:id”)
 - Kokios antraštės (pvz.: “start”, “count”)
 - Kokia užklausos kūno struktūra (pageidautinas pavyzdys, pavyzdžiai)
 - Kokie galimi atsakymo HTTP statusai (pvz.: 400, 500, 200)
 - Kokia atsakymo kūno struktūra (pageidautinas pavyzdys, pavyzdžiai)

Dokumentacija su apiDoc

```
1  /**
2   * @api {get} /login Login to Trade Manager
3   * @apiName GetLogin
4   * @apiGroup Auth
5   * @apiVersion 1.0.0
6   * @apiDescription Permissions:
7   * - any user
8   *
9   * @apiHeader {String} Authorization Base64 encoded user name and password used in HTTP Basic access authentication.
10  *
11  * @apiSuccess {String} token Token value to be used to call other REST services that require authentication.
12  *
13  * @apiSuccessExample Account details:
14  *   HTTP/1.1 200 OK
15  *   {
16  *     "token": "123e4567-e89b-12d3-a456-426655440000"
17  *   }
18  *
19  * @apiError (Client errors 400) InvalidCredentials Wrong user name or password are provided.
20  * @apiErrorExample InvalidCredentials:
21  *   HTTP/1.1 400 Bad Request
22  *   {
23  *     "error": "InvalidCredentials"
24  *   }
25  *
26  * @apiError (Server errors 500) InternalServerError Some internal error, for instance, out of memory.
27  * @apiErrorExample InternalServerError:
28  *   HTTP/1.1 500 Internal Server Error
29  *   {
30  *     "error": "InternalServerError"
31  *   }
32  *
33  */
34 expressApp.get('/api/login', (req, res) => {
35   // kodas
36 });

});
```

Dokumentacija su apiDoc

Auth - Login to Trade Manager

1.0.0 ▾

Permissions:

- any user

GET

<http://localhost:3001/api/login>

Header

| Field | Type | Description |
|---------------|--------|---|
| Authorization | String | Base64 encoded user name and password used in HTTP Basic access authentication. |

Success 200

| Field | Type | Description |
|-------|--------|---|
| token | String | Token value to be used to call other REST services that require authentication. |

Account details:

```
HTTP/1.1 200 OK
{
  "token": "123e4567-e89b-12d3-a456-426655440000"
}
```

Client errors 400

| Name | Description |
|--------------------|---|
| InvalidCredentials | Wrong user name or password are provided. |

Server errors 500

| Name | Description |
|---------------------|---|
| InternalServerError | Some internal error, for instance, out of memory. |

Dokumentacija su “joi” “kalba”

```
const Joi = require('joi');

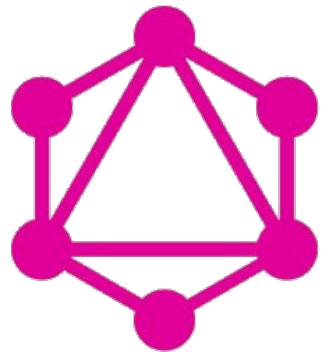
const schema = Joi.object().keys({
  username: Joi.string().alphanum().min(3).max(30).required(),
  password: Joi.string().regex(/^[a-zA-Z0-9]{3,30}$/),
  access_token: [Joi.string(), Joi.number()],
  birthyear: Joi.number().integer().min(1900).max(2013),
  email: Joi.string().email()
}).with('username', 'birthyear').without('password', 'access_token');

// Return result.
const result = Joi.validate({ username: 'abc', birthyear: 1994 }, schema);
// result.error === null -> valid
```

Versijavimas

- Sukurtas REST API yra atnaujinamas
 - Galbūt keičiasi užklausos parametrai
 - Galbūt keičiasi atsakymo struktūra
- Kiekvienam API ar servisui yra nurodoma versija
 - Vietoje `https://api.com/api/user`
 - `https://api.com/ 1/api/user`
 - `https://api.com/api/ 1/user`

GraphQL



GraphQL



T5 - Web frontend'o pagrindai

Asynchroninės operacijos (async operations)

```
1 const user = userService.getUserById(id);
```

```
1 try {
2     const user = userService.getUserById(id);
3 } catch (error) {
4     console.error(error);
5 }
```

} synchroninė

```
1 userService.getUserById(id, (error, user) => {
2     if (error) {
3         return console.error(error);
4     }
5     user;
6 });
```

} asynchroninė

Pažadai (Promises)

```
1 userService.getUserId(id, (error, user) => {
2   if (error) {
3     return console.error(error);
4   }
5   user;
6});
```

```
1 const getUserById = (id, cb) => {
2   dbCollection.find({id: new ObjectID(id)}).toArray(cb);
3};
```

```
1 getUserId(id)
2   .then(user => {
3     user;
4   })
5   .catch(error => {
6     console.error(error);
7   });
8};
```

```
1 const getUserById = id => {
2   return new Promise((resolve, reject) => {
3     dbCollection.find({ id: new ObjectID(id) }).toArray((error, users) => {
4       if (error) {
5         return reject(error);
6       }
7       resolve(users);
8     });
9   });
10};
```

Promises - Mongoose

```
1 userService.getUserId(id, (error, user) => {
2   if (error) {
3     return console.error(error);
4   }
5   user;
6});
```

```
1 const getUserId = (id, cb) => {
2   dbCollection.find({id: new ObjectID(id)}).toArray(cb);
3};
```

```
1 getUserId(id)
2   .then(user => {
3     user;
4   })
5   .catch(error => {
6     console.error(error);
7});
```

```
1 const getUserId = id => {
2   return User.findById(id);
3};
```

“Vėl aplankymo pragaras” (“callback hell”)

```
1  userRepository.findUserById(id, (error, user) => {
2      // do smth with a user
3  });

4  const getAllTransactions = (cb) => {
5      findUsers((error, users) => {
6          if (error) {
7              return cb(error);
8          }
9          findAccountsByUsers(users, (error, accounts) => {
10             if (error) {
11                 return cb(error);
12             }
13             findTransactionsByAccounts(accounts, (error, transactions) => {
14                 if (error) {
15                     return cb(error);
16                 }
17             })
18         });
19     });
20 };

21 
```



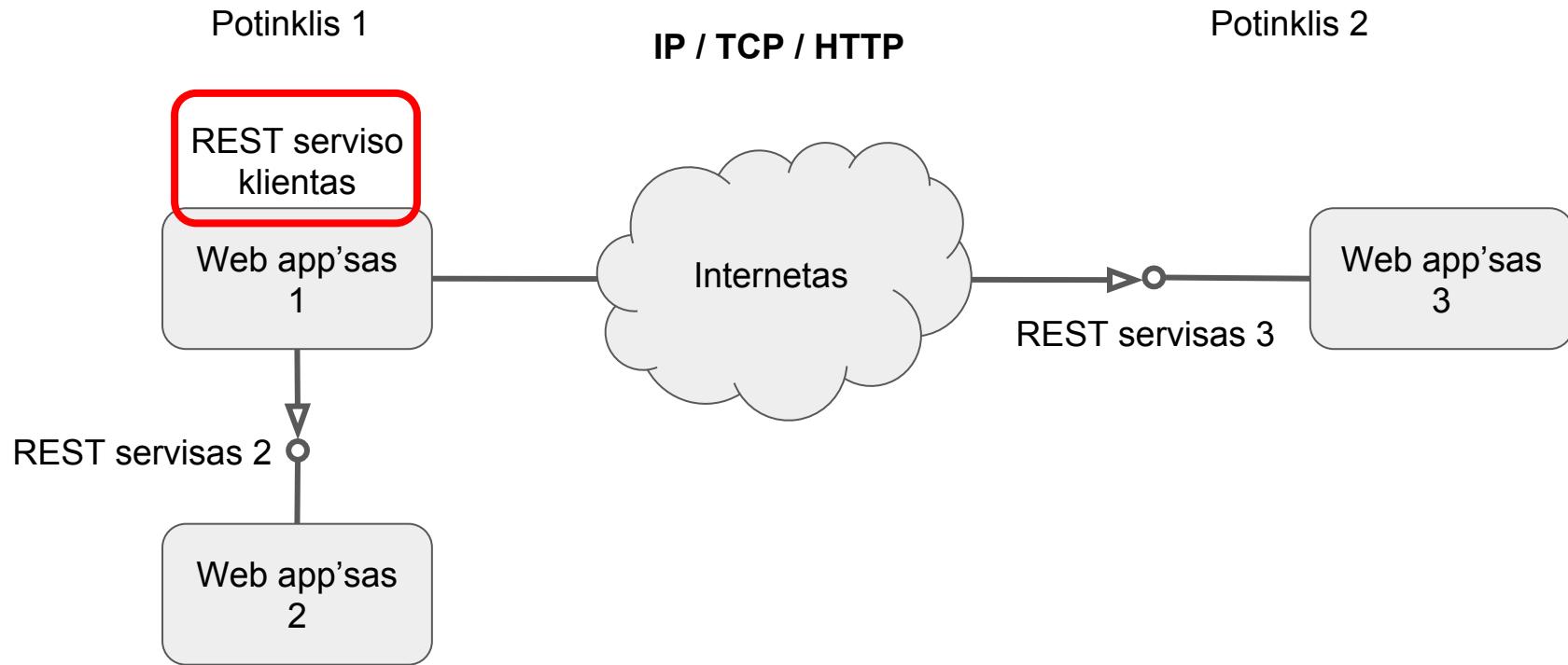
callback

Promises išsprendžia “callback hell”

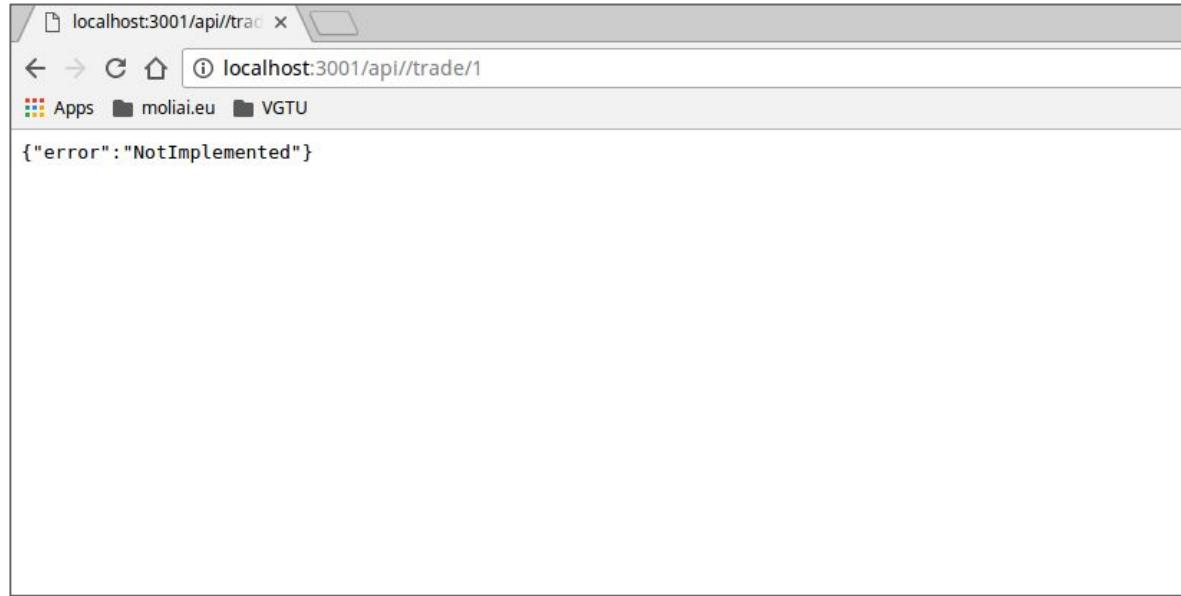
```
1 const getAllTransactions = (cb) => {
2   findUsers((error, users) => {
3     if (error) {
4       return cb(error);
5     }
6     findAccountsByUsers(users, (error, accounts) => {
7       if (error) {
8         return cb(error);
9       }
10    findTransactionsByAccounts(accounts, (error, transactions) => {
11      if (error) {
12        return cb(error);
13      }
14      cb(null, transactions);
15    })
16  })
17 });
18 };
```

```
1 const getAllTransactions = (cb) => {
2   findUsers()
3     .then(users => findAccountsByUsers(users))
4     .then(accounts => findTransactionsByAccounts(accounts))
5     .then(transactions => cb(null, transactions))
6     .catch(error=>cb(error));
7   };
8 }
```

REST servisai

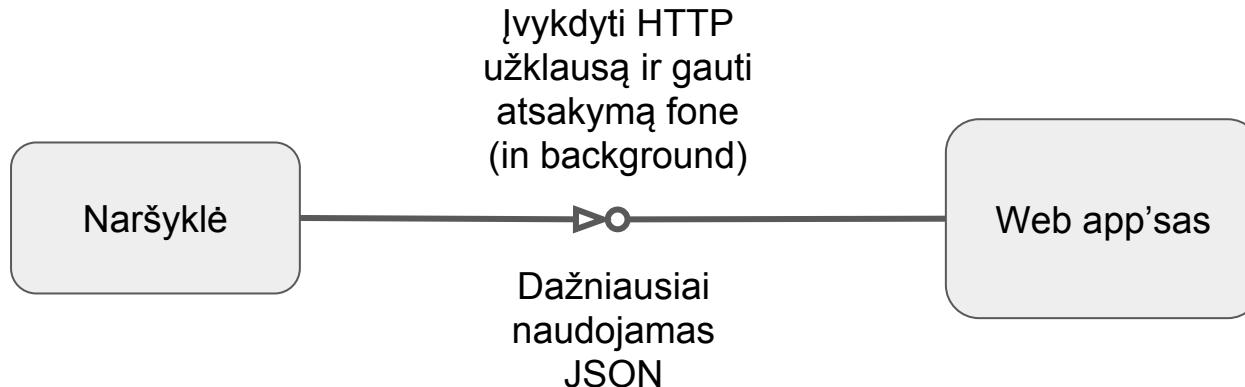


Web klientas - Chrome



Web klientas - AJAX

Asynchronous JavaScript and XML



AJAX klientas - fetch

```
1  export const getUsers = (password, done) => {
2      fetch('/api/users', { method: 'GET', headers: { pass: password } })
3          .then(response => {
4              if (response.status >= 400 && response.status < 600) {
5                  return response.json().then((body) => {
6                      throw new Error(body.error);
7                  });
8              }
9              return response.json();
10         })
11         .then(users => done(null, users))
12         .catch(error => done(error));
13     };

```

Manipuliavimas masyvu

```
1 const arr = ['a', 'b'];
2 arr[0]; // a
3 arr[1]; // b
4 arr[2]; // undefined
5 arr.length; // 2
6
7 arr[3] = 'd';
8 arr[2]; // undefined
9 arr[3]; // d
10 arr.length; // 4

1 const arr = ['b', 'c'];
2 arr.splice(0, 1, 'a', 'A');
3 arr; // ['a', 'A', 'c']
4 arr.splice(2);
5 arr; // ['a', 'A']
```

```
1 const arr = ['b', 'c'];
2 arr.push('d');
3 arr; // ['b', 'c', 'd']
4 arr.unshift('a');
5 arr; // ['a', 'b', 'c', 'd']
6 arr.pop(); // 'd'
7 arr; // ['a', 'b', 'c']
8 arr.shift(); // 'a'
9 arr; // ['b', 'c']
10 arr.concat('d', ['e', 'f']); // ['b', 'c', 'd', 'e', 'f']
11 arr; // ['b', 'c']
12 arr.slice(0, 2); // ['b', 'c']
13 arr.slice(2, 4); // ['d', 'e']
```

Masyvo rūšiavimas

```
1  const arr = ['b', 'a', 'c'];
2  arr.sort();
3  arr; // ['a', 'b', 'c']
4
5  const users = [{id: 1, name: 'Mykolas'}, {id: 2, name: 'Adolfas'}, {id: 3, name: 'Birutė'}];
6  users.sort();
7  users; // [ { id: 1, name: 'Mykolas' }, { id: 2, name: 'Adolfas' }, { id: 3, name: 'Birutė' } ]
8
9  users.sort((u1, u2) => u1.name.localeCompare(u2.name));
10 users; // [ { id: 2, name: 'Adolfas' }, { id: 3, name: 'Birutė' }, { id: 1, name: 'Mykolas' } ]
11
12 users.sort((u1, u2) => u2.name.localeCompare(u1.name));
13 users; // [ { id: 1, name: 'Mykolas' }, { id: 3, name: 'Birutė' }, { id: 2, name: 'Adolfas' } ]
14
15 users.reverse((u1, u2) => u1.name.localeCompare(u2.name));
16 users; // [ { id: 1, name: 'Mykolas' }, { id: 3, name: 'Birutė' }, { id: 2, name: 'Adolfas' } ]
```

Paieška masyve

```
1 const arr = ['a', 'b', 'c'];
2 arr.indexOf('b'); // 1
3 arr.indexOf('d'); // -1
4
5 const users = [{ id: 1, name: 'Mykolas' }, { id: 2, name: 'Sofija' }];
6 users.findIndex(user => user.id === 2); // 1
7 users.findIndex(user => user.id === 3); // -1
8
9 users.find(user => user.id === 2); // { id: 2, name: 'Sofija' }
10 users.find(user => user.id === 3); // -1
11
12 users.some(user => user.id === 2); // true
13 users.some(user => user.id === 3); // false
```

```
1 let user = null;
2 for (let u of users) {
3     if (u.id === 2) {
4         user = u;
5         break;
6     }
7 }
8 console.log(user);
```

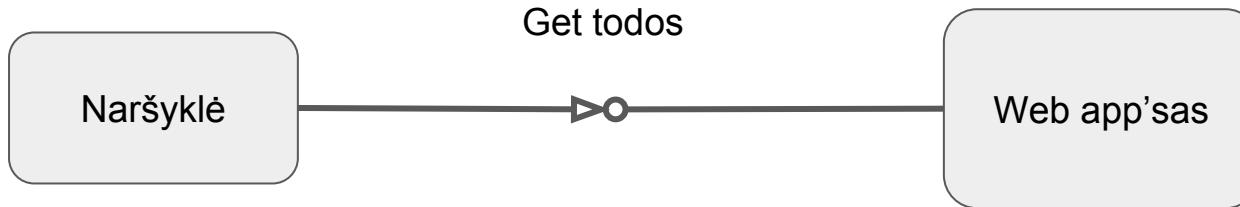
Masyvo iteravimas, persiejimas ir filtravimas

```
1 const users = [{ id: 1, name: 'Mykolas' }, { id: 2, name: 'Sofija' }];
2
3 users.forEach(user => console.log(user)); // { id: 1, name: 'Mykolas' }\n{} id: 2, name: 'Sofija' }
4
5 users.map(user => user.id); // [1, 2]
6 users.map(user => user.name); // ['Mykolas', 'Sofija']
7 users.map(user => {
8     return { desc: `${user.name} identity is ${user.id}` }
9 });
10 // [ { desc: 'Mykolas identity is 1' }, { desc: 'Sofija identity is 2' } ]
11 users.filter(user => user.id); // [{ id: 1, name: 'Mykolas' }, { id: 2, name: 'Sofija' }]
12 users.filter(user => user.id === 3); // []
13
14 users.filter(user => user.id === 2).map(user => user.id); // [2]
```

Magija su masyvu - reduce

```
1  const users = [{ id: 1, income: 1025 }, { id: 2, income: 654 }, { id: 3, income: 2015 }];
2
3  users.reduce(
4      (sum, user) => {
5          return sum + user.income;
6      },
7      0
8  ); // 3694
9
10 users.reduce(
11     ({sum, count}, user) => {
12         return {
13             sum: sum + user.income,
14             count: count + 1
15         };
16     },
17     { sum: 0, count: 0 }
18 ); // { 3694, 3 }
```

Manipuliavimas duomenimis frontend'e



```
1 const notDoneTitles = todos  
2   .filter(todo => !todo.isDone)  
3   .map(todo => todo.title);
```

Komponentai (components)

The screenshot displays the homepage of the 'Verslo žinios' website, featuring several distinct components:

- Header:** A navigation bar at the top with categories: Verslo aplinka, Rinkos, Finansai, Prekyba, Vadyba, Rinkodara, Technologijos, Daugiau, PREMIUM, user icon, search icon, and mobile icon.
- Hero Area:** A large image of a man in a suit speaking, with the text "JAV investuotojai apie Lietuvą: gera šalis, bet talentų mažėja" and a "PREMIUM" button.
- News List:** A vertical list of news items with timestamps:
 - 15:15 Vanile tampa auksu: rekordinės kainos dar šoks į viršų
 - 15:05 JAV investuotojai apie Lietuvą: gera šalis, bet talentų mažėja **PREMIUM**
 - 15:02 „BaltCap“ didina investicijas į Estijos statybininkę
 - 14:31 Prancūzijos mokykloje – šūviai
 - 14:12 Deklaruojantiesiems pajamas – paskutinis kartas be GPM lubų
 - 13:34 Į TVF atsiuntė sprogstantį laišką
- Call-to-Action:** A large blue banner on the right side with the text "Metų įmonės teisininkas/-ė" and "RINKIMAI".
- Footer Categories:** A row of four categories: VADYBA, RINKOS, RINKODARA, and PREKYBA, each with a small image and a brief description.
- Footer News:** A row of three news items with timestamps and small images:
 - Darbo kodeksas: 10 nuostatai, kurias nutarta pakeisti
 - Deklaruojantiesiems pajamas – paskutinis kartas be GPM lubų
 - „Password 2017“: sėkmės laurai – ne didžiausiems, o besikeičiantiems
- Footer Call-to-Action:** A red button at the bottom right with the text "KVIEČIAME DALYVAUTI".

DOM

Document Object Model (document)

The screenshot shows a news website layout. At the top is a header bar with the logo 'verslo žinios' and navigation links: Verslo aplinka, Rinkos, Finansai, Prekyba, Vadyba, Rinkodara, Technologijos, Dauglau, PREMIUM, user icon, search icon, and mobile icon. Below the header is a main content area with a large image of a man speaking. To the right of the image is a sidebar with news snippets. The main content area has a blue border and contains sections for breaking news, latest news, and top news. At the bottom is a sidebar with categories: VADYBA, RINKOS, RINKODARA, and PREKYBA, each with a small image and text. A red box highlights the 'PREMIUM' link in the header, and a green box highlights the main content area.

```
1 <body>
2   <div class="header">
3     <div class="logo">
4       
5     </div>
6     <div class="menu">...</div>
7     <div class="options">...</div>
8   </div>
9   <div class="content">
10    <div class="breakingNews">...</div>
11    <div class="latestNews">...</div>
12    <div class="topNews">...</div>
13  </div>
14  <div style="rightSide">
15    <div class="advertising">
16      <a href="...">
17        
18      </a>
19    </div>
20  </div>
21 </body>
```

HTML - index.html

Hypertext Markup Language

```
1  <!DOCTYPE html>
2  <html>
3
4  <head>
5      <title>App title</title>
6  </head>
7
8  <body>
9      <div id="app"></div>
10     <script src="/app/dist/bundle.js"></script>
11  </body>
12
13 </html>
```

HTML 5

- <input />
- <button />
- <div />
-
- <a />
-
- <video />
- <audio />



Express.js statiniai resursai

```
▸ lab3
  ▷ frontend
    ▷ dist
      bundle.js
      bundle.js.map
  ▷ lib
    .gitignore
    index.html
    .gitignore
    index.js
  README.md
```

```
1  const express = require('express');
2
3  const app = express();
4
5  app.use('/app/dist/bundle.js', express.static('./frontend/dist/bundle.js'));
6  app.use('/app/index.html', express.static('./frontend/index.html'));
7
8  app.get('/', (req, res) => {
9    res.redirect('./app/index.html');
10 });
11
12 app.listen(3004);
```

DOM manipuliavimas su JavaScript

```
<!DOCTYPE html>
<html>
<head>
</head>
<body>

<p>If you click on me, I will disappear.</p>
<p>Click me away!</p>
<p>Click me too!</p>
|
</body>
</html>
```

```
for (p of document.getElementsByTagName('p')) {
    p.onclick = event => {
        event.target.style.display = 'none';
    };
}
```

DOM manipuliavimas su jQuery

- Biblioteka, kuri leidžia paprastai manipoliuoti DOMu: visais jo objektais, jų savybėmis bei funkcijomis

```
<!DOCTYPE html>
<html>
<head>
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.1.1/jquery.min.js"></script>
</head>
<body>

<p>If you click on me, I will disappear.</p>
<p>Click me away!</p>
<p>Click me too!</p>

</body>
</html>

for (p of document.getElementsByTagName('p')) {
  p.onclick = event => {
    event.target.style.display = 'none';
  };
}

$(document).ready(() => {
  $("p").click(() => {
    $(this).hide();
  });
});
```

Pavyzdys

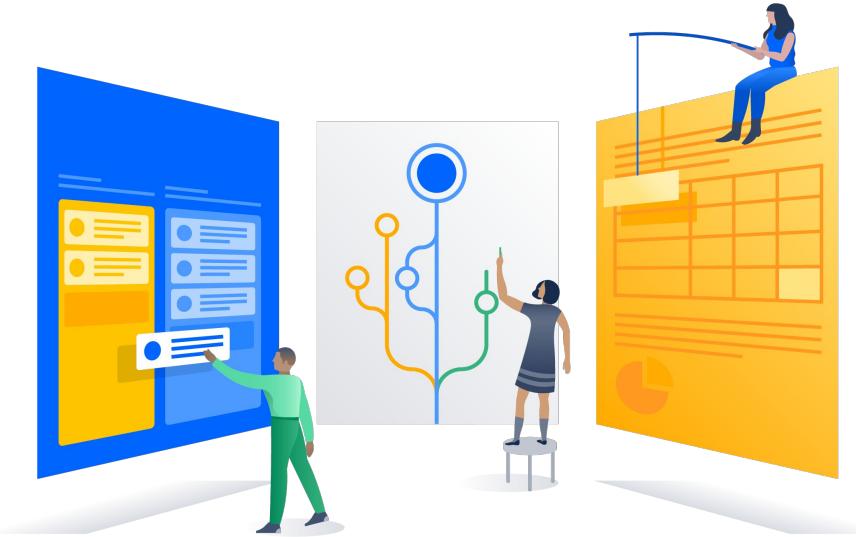
```
1  <!DOCTYPE html>
2  <html>
3
4  <head>
5      <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.1.1/jquery.min.js"></script>
6  </head>
7
8  <body>
9      <p>Users</p>
10     <ul id="userList"></ul>
11     <p id="errorMessage"></p>
12     <script src="app.js"></script>
13 </body>
14
15 </html>

→ 1 const getUsers = () => {
2     return fetch('/api/user', { method: 'GET' }); // [{ id: 1, name: 'Mykolas' }, { id: 2, name: 'Marius' }]
3 };
4
5 getUsers()
6     .then(users => {
7         users.forEach(user => {
8             $('#userList').append(`<li>${user.name}</li>`);
9         });
10    })
11    .catch(error => {
12        $('#errorMessage').text(error.message)
13    });

```

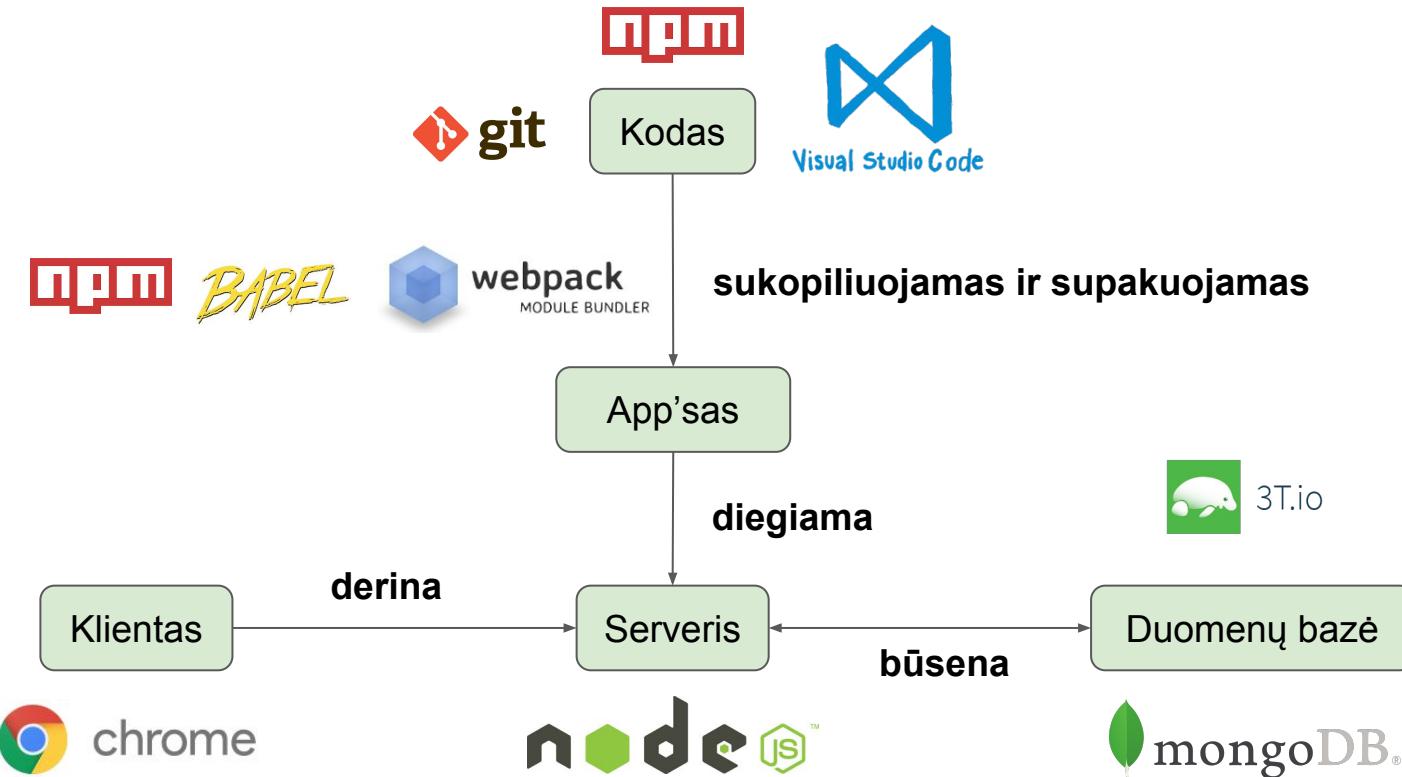
Veiksmai frontend'e

- Galimybė imti duomenis iš backend'o nadojantis AJAX klientu (pvz.: fetch)
- Galimybė apdoroti gautus iš backend'o duomenis naudojantis deklaratyviu programavimu (pvz.: naudojant masyvo (array) metodus)
- Galimybė panaudoti (apdorotus) duomenis DOM modeliui manipuliuoti (pvz.: su jQuery, React)

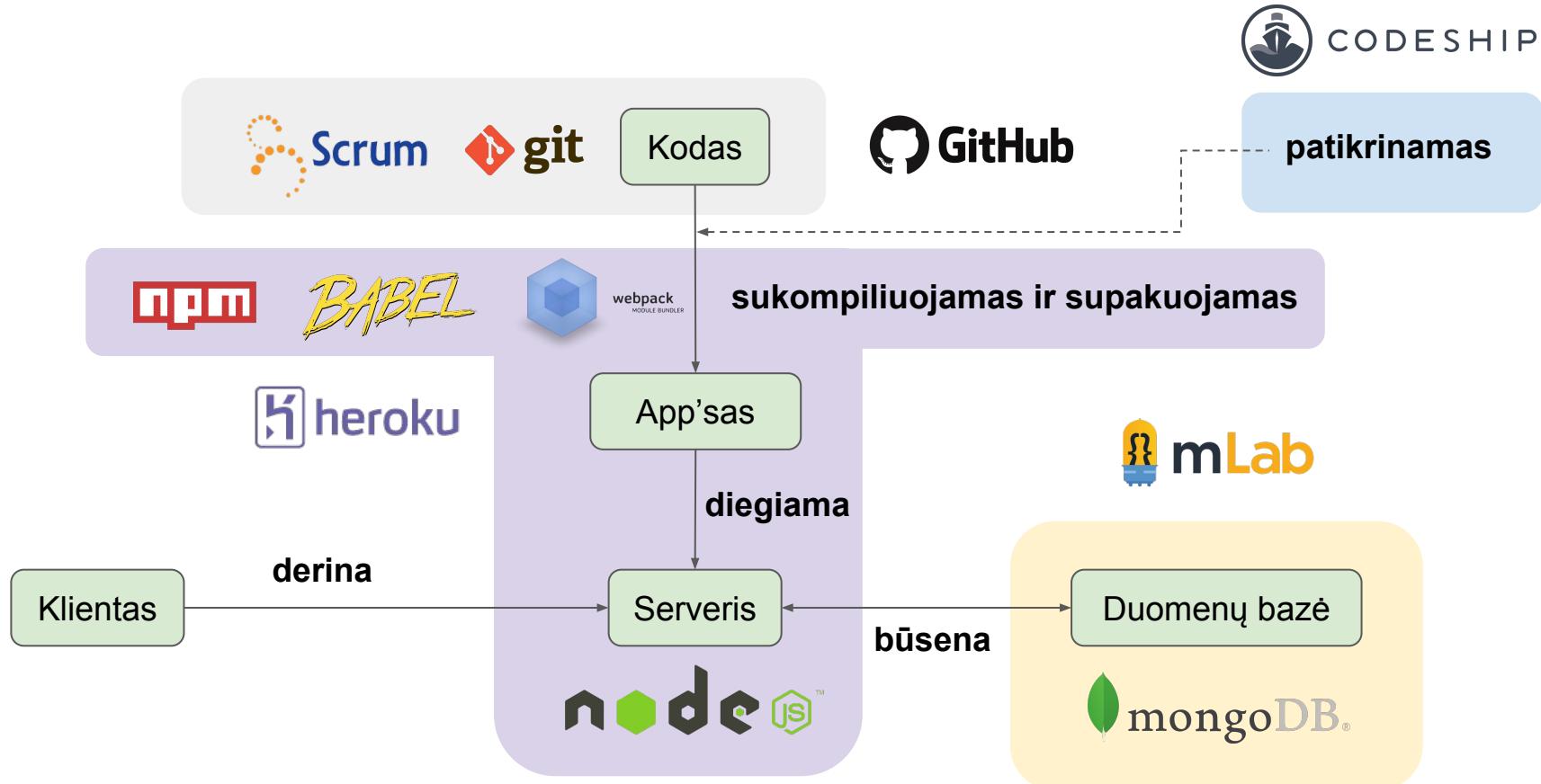


P3 - Sistemos kūrimo įrankiai

Sistemos kūrimo įrankiai programuotojo kompiuteryje



Sistemos kūrimo paslaugos debesyje



Kodo valdymo įrankis - Git

- <https://git-scm.com/downloads>
- Dažniausiai naudojamos komandos:
 - Lokalios repozitorijos būsena:
 - git status
 - Išsaugoti (commit) pakeitimus lokalioje repozitorijoje:
 - git commit -a
 - Išsaugojimų istorija:
 - git log
 - Išsaugoti (push) pakeitimus nutolusioje (remote) repozitorijoje (pvz. GitHub):
 - git push
 - Gauti (pull) pakeitimus iš nutolusios (remote) repozitorijos (pvz. GitHub):
 - git pull
 - Atšaukti visus lokalios repozitorijos išsaugojimus po paskutinio gavimo:
 - git reset --hard HEAD

PJ kūrimo valdymo įrankis ir modulių paslauga - npm

- Node.js dalis: <https://nodejs.org/en/download/>
- Modulių repozitorija
 - <https://www.npmjs.com>
- Node.js projekto valdymo įrankis
 - Priklausomybių (modulių) diegimas
 - npm install <modulis>@<versija> [--save] [--save-dev]
 - Priklausomybių (modulių) išdiegimas
 - npm uninstall <modulis>
 - App'so skriptų (scripts) leidimas
 - npm run <skriptas>
 - App'so leidimas
 - npm start
 - Testų leidimas
 - npm test

Kodo redaktorius - Visual Studio Code

- Atviro kodo nemokamas išeities kodo redaktorius
 - <https://code.visualstudio.com/download>
- Alternatyvos
 - Atom
 - Sublime Text
 - Vim / Emacs
 - WebStorm
 - NetBeans
 - Eclipse

MongoDB valdymo įrankis - 3T.io

- Heroku paslaugos app'so nustatymuose (Settings) prie kintamųjų (Config Variables) yra išsaugotas prisijungimo prie DB URL pavadinimu 'MONGODB_URI':
 - `mongodb://<naudotojas>:<slaptažodis>@<hostas>:<prievadas>/<duomenų bazė>`
- 3T.io nustatymai susieti su URL:
 - Reikia naudoti mygtuką 'From URI...'
- Prisijungimo pavadinimas: komandos pavadinimas
- Įrankis leidžia valdyti duomenų bazės rinkinius (collections) bei indeksus (indexes)

Web app'so klientas - Chrome

- <https://www.google.com/chrome/browser/desktop/>
- Chrome DevTools
 - JavaScript kodo sintaksės klaidų sprendimas
 - Web aplikacijos būsenos klaidų sprendimas
 - Naudotojo sąsajos (UI) klaidų sprendimas
 - REST klaidų sprendimas
- React Developer Tools
 - Komponentų medžio (Component tree) derinimas
 - Komponento ypatybių (Component properties) derinimas
 - Komponento būsenos (Component state) detinimas

App'so išleidimo paslauga - Heroku

Overview Resources Deploy Metrics Activity Access Settings

- Kiekvienai 'organizacijos' komandai sukuriamas app'sas
 - Suteikiama prieiga komandos nariams
- App'sas susideda iš:
 - Būsenos ir įvykių suvestinės (log) (Overview) (Activity)
 - Resursų (Resources) - mūsų atveju: Virtualios Mašinos (VM), MongoDB ir Codeship
 - App'so išleidimo (Deploy) - mūsų atveju pagal GitHub paslaugos repositorijos kodą
 - App'so metrikos (Metrics) - mūsų atveju nepasiekiamos, nes naudojamės nemokamai
 - Komandos narių (Access)
 - Nustatymų (Settings) - mūsų atveju čia galima rasti MongoDB URL ir Codeship ID

Nuolatinės integracijos (CI) paslauga - Codeship

- Įkelti kaip Heroku paslaugos app'so resursą
- Susieti su GitHub paslauga
- Pasirinkti standartinius Node.js app'so nustatymus
 - App'so kodas bus jdiegtas: `npm install`
 - App'so kodas bus ištestuotas: `npm test`
- Heroku paslaugos app'so nustatymuose (Settings) reikia aktyvuoti diegimą tik po sėkmingos app'so kodo verifikacijos
- Pasirinktinai:
 - Integracija su Slack - pranešti apie nepavykusias app'so kodo verifikacijas

MongoDB paslauga - mLab

- Įkelti kaip Heroku paslaugos app'so resursą
- Heroku paslaugos app'so nustatymuose (Settings) prie kintamųjų (Config Variables) yra išsaugotas prisijungimo prie DB URL pavadinimu 'MONGODB_URI', pvz.:
 - `mongodb://naudotojas:slaptažodis@ds111.mlab.com:12345/heroku_12abcde1a`
- Gautą kintamajį galima gauti Node.js app'se:
 - `process.env.MONGODB_URI`
 - `MongoClient.connect(process.env.MONGODB_URI, (error, db) => {})`

Rezultatas

- Darbo vietoje turėtų būti šie įrankiai:
 - Visual Studio Code (arba kitas) - kodo redaktorius (patikrinti ar pasileidžia)
 - **git** - kodo valdymas (patikrinti: `git --version`)
 - **Node.js / npm** - web app'so kūrimo valdymas ir leidimas (serve)
 - Patikrinti NPM: `npm --version`
 - Patikrinti Node.js: `node --version`
 - **Chrome** (su React Developer Tools įskiepiu) - web app'so klientas ir derintojas (debugger)
 - **MongoDB** - duomenų bazė (patikrinti ar pasileidžia)
 - **3T.io** - duomenų bazės vizuali naudotojo sąsaja (GUI) (patikrinti ar pasileidžia)
- Turėtų būti pasiekiamos šios paslaugos:
 - **GitHub** - kodo repozitorija bei PĮ kūrimo proceso įrankiai
- Pasirinktinai:
 - **Heroku** (su mLab resursu) - web'apso išleidimas (deployment): serveris + DB
 - **Codeship** - kodo verifikavimas / kokybės tikrinimas
 - **mLab** - Mongo DB debesyje



T6 - Modernus web sistemas frontend'as

DOM

Document Object Model (document)

The screenshot shows a news website layout. At the top is a red-bordered header bar containing the logo 'verslo žinios', a navigation menu with links like 'Verslo aplinka', 'Rinkos', 'Finansai', etc., and a 'PREMIUM' section with user icons. Below the header is a large image of a man in a suit. To the right of the image is a column of news snippets. The main content area has a blue background with the word 'TEISĖS' in large red letters. At the bottom of this area is a red button with the text 'Kviečiame dalyvauti'. The footer contains links for 'VADYBA', 'RINKOS', 'RINKODARA', and 'PREKYBA', along with some text and small images.

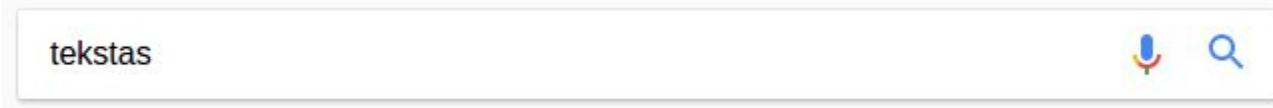
```
1 <body>
2   <div class="header">
3     <div class="logo">
4       
5     </div>
6     <div class="menu">...</div>
7     <div class="options">...</div>
8   </div>
9   <div class="content">
10    <div class="breakingNews">...</div>
11    <div class="latestNews">...</div>
12    <div class="topNews">...</div>
13  </div>
14  <div style="rightSide">
15    <div class="advertising">
16      <a href="...">
17        
18      </a>
19    </div>
20  </div>
21 </body>
```

HTML komponentai kaip objektai

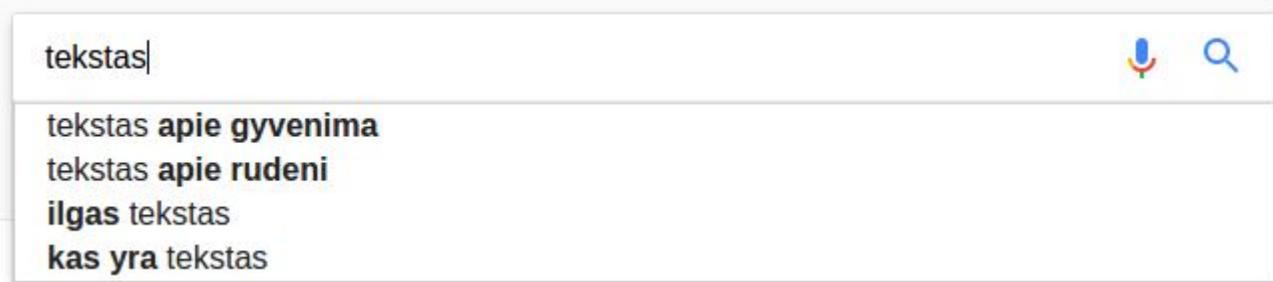
- <input />
 - <button />
 - <div />
 -
 - <a />
 -
 - <video />
 - <audio />
- 

Objektinis programavimas

- Duomenų enkapsuliacija: būsena ir veiksmai yra vieno objekto dalis
 - Pvz.: teksto įvedimo laukas turi būseną - įvestas tekstas ir veiksmus - pakeisti tekštą



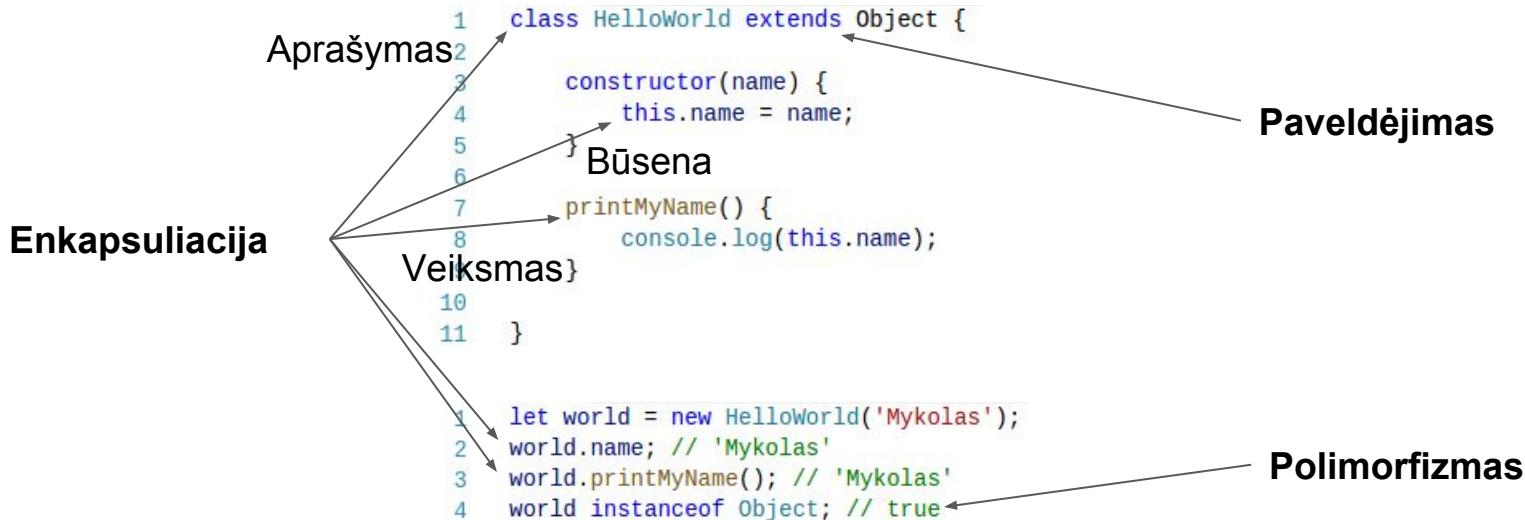
- Paveldėjimas ir polimorfizmas: objektas gali paveldėti būseną ir veiksmus iš kito objekto, paveldėjės objektas taip pat yra ir paveldimas objektas
 - Pvz.: teksto įvedimo laukas su pasiūlymais - paveldi įvestą tekštą ir leidžia pakeisti tekštą ir jis vis tiek naudojamas kaip teksto įvedimo laukas



Klasė prieš Objektą (class vs object)

- Klasė - bendrinis dalykas
 - Aprašymas kokią būseną ir veiksmus objektas turi
 - Pvz.: teksto įvedimo laukas (Text field) su įvestu tekstu ir galimybe jį pakeisti
- Objektas - specifinis dalykas
 - Konkreti klasės instancija (instance)
 - Pvz.: konkretus teksto įvedimo laukas naudotojo vardui įvesti su identifikatoriumi (ID) 'login'
 - Pvz.: konkretus teksto įvedimo laukas naudotojo slaptažodžiui įvesti su identifikatoriumi (ID) 'pass'

JavaScript klasė ir objektas



ES5 objekto konstruktorius prieš ES6 klasę

JavaScript klasė yra tiesiog funkcija, kuri sukonstruoja objektą

```
1  typeof HelloWorld // function
```

```
1  class HelloWorld extends Object {          1  function HelloWorld(name) {  
2    constructor(name) {                      2    this.name = name;  
3      this.name = name;                     3  }  
4    }                                         4  HelloWorld.prototype.printMyName = function () {  
5    printMyName() {                         5    console.log(this.name);  
6      console.log(this.name);               6  };  
7    }                                         7  var __extends = (this && this.__extends) || function (d, b) {  
8    }                                         8    for (var p in b) if (b.hasOwnProperty(p)) d[p] = b[p];  
9    }                                         9    function __() { this.constructor = d; }  
10   }                                         10   d.prototype = b === null ? Object.create(b) : (__.prototype = b.prototype, new __());  
11 }                                         11  
12                                         12   __extends(HelloWorld, __super);  
13                                         13  
14                                         14  
15                                         15
```

JavaScript klasė prieš Java klasę

```
1  public class HelloWorld extends Object {  
2  
3      private String name;  
4  
5      public HelloWorld(String name) {  
6          this.name = name;  
7      }  
8  
9      public void printMyName() {  
10         System.out.println(name);  
11     }  
12 }  
13 }
```

```
1  class HelloWorld extends Object {  
2  
3      constructor(name) {  
4          this.name = name;  
5      }  
6  
7      printMyName() {  
8          console.log(this.name);  
9      }  
10 }  
11 }
```

Nėra priegos modifikatorių - viskas laisvai prieinama

Kintamujų, funkcijų (metodų) argumentų ir rezultatų tipas nėra statinis - jis gali būti bet koks

“this” JavaScript klasėje

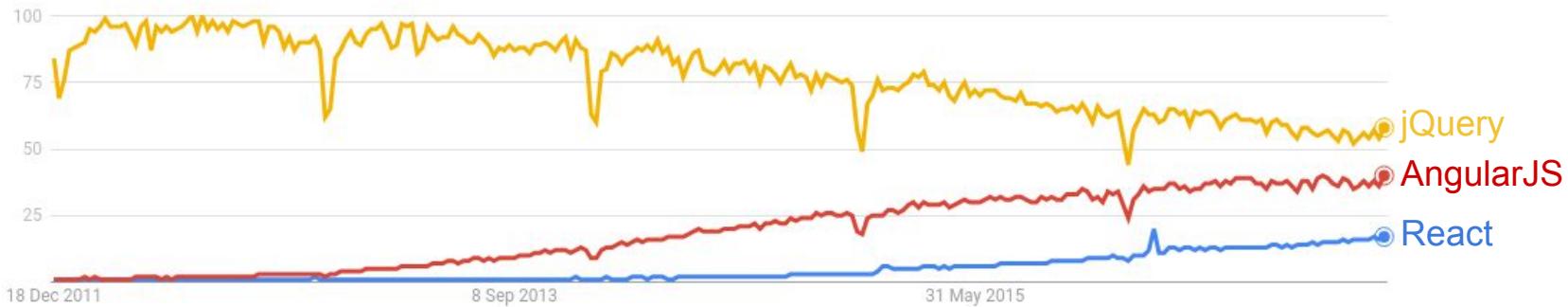
objekto savybė
(property)

objekto metodas
(method)

```
1  class WordsTextField extends TextField {  
2  
3      constructor() {  
4          this.words = [];  
5      }  
6  
7      setText(text) {  
8          const words = text.split(' ');  
9          if (words.length === 1) {  
10              this.registerWord(word);  
11              return;  
12          }  
13          words.forEach(word => {  
14              this.registerWord(word);  
15          }); // words.forEach(this.registerWord);  
16      }  
17  
18      registerWord(word) {  
19          this.words.push(word);  
20      }  
21  }
```



jQuery, AngularJS ir React



React frontend app'sas

```
lab3
  ▾ frontend
    ▾ dist
      bundle.js
      bundle.js.map
  ▾ lib
    .gitignore
    index.html
    .gitignore
    index.js
    README.md
```

```
1  <!DOCTYPE html>
2  <html>
3
4  <head>
5    <title>App title</title>
6  </head>
7
8  <body>
9    <div id="app"></div>
10   <script src="/app/dist/bundle.js"></script>
11
12
13 </html>
```

React frontend app'sas

```
lib
index.jsx
PingComponent.jsx
pingInfrastructureService.js
```

```
1 import ReactDOM from 'react-dom';
2 import React from 'react';
3 import Ping from "./PingComponent";
4
5 ReactDOM.render(
6   <Ping/>,
7   document.getElementById("app")
8 );
```

React frontend app'sas

```
lib
  index.jsx
  PingComponent.jsx
  pingInfrastructureService.js
```

```
1 import React from 'react';
2
3 export default class PingComponent extends React.Component {
4   render() {
5     return <div>
6       <p>This is a ping component</p>
7     </div>
8   }
9 }
```

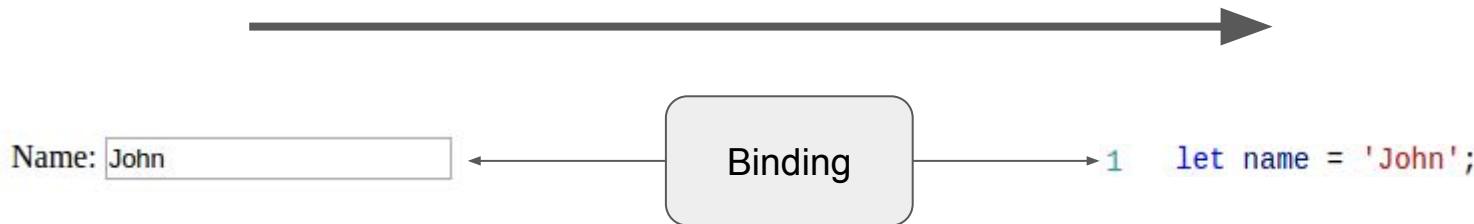
Duomenų surišimas (Data binding)

Bendra technika kuri tarpusavyje susieja duomenų šaltinius (data sources) iš jų tiekėjo (provider) ir jų naudotojo (consumer) pusės ir tuos šaltinius sinchronizuoją



Vieno kelio surišimas (one way binding)

Sinchronizuojama tik į vieną puse - iš tiekėjo į naudotoją, arba atvirkščiai

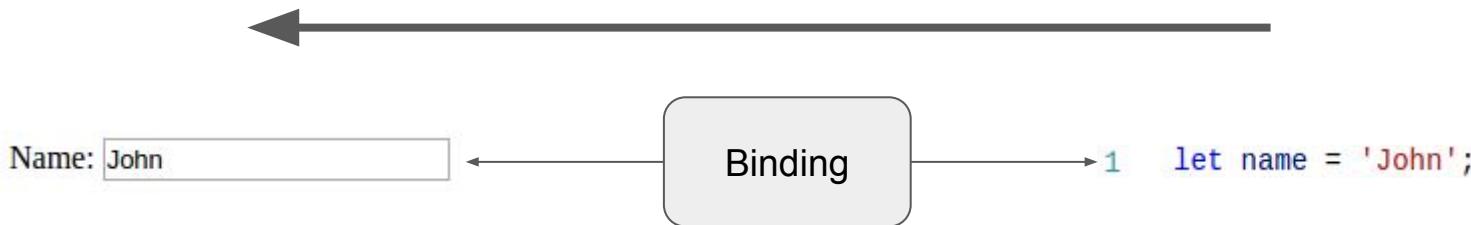


Jei pakeičiame teksto lauko reikšmę: pakeitimas **sinchronizuojamas** į kintamajį "name"

Jei pakeičiame kintamojo "name" reikšmę: pakeitimas **NEsynchronizuojamas** į teksto lauko reikšmę

Vienos pusės surišimas (one way binding)

Sinchronizuojama tik į vieną puse - iš tiekėjo į naudotoją, arba atvirkščiai

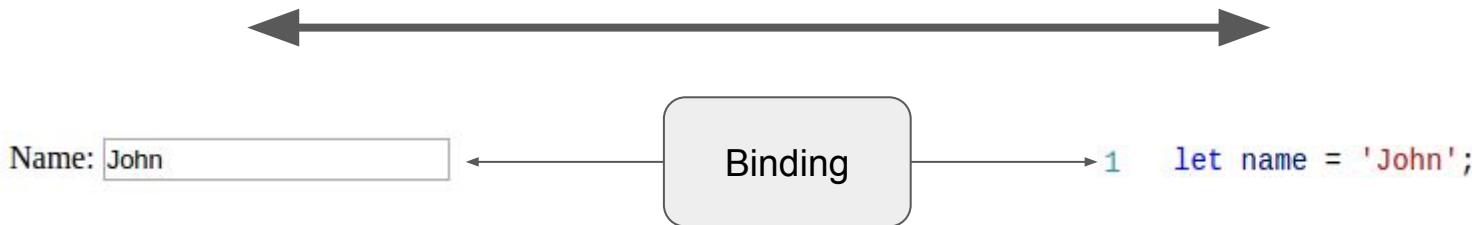


Jei pakeičiame kintamojo "name" reikšmę: pakeitimas **sinchronizuojamas** į teksto lauko reikšmę

Jei pakeičiame teksto lauko reikšmę: pakeitimas **NEsynchronizuojamas** į kintamąjį "name"

Dviejų pusių surišimas (two way binding)

Sinchronizuojama tik į vieną puse - iš tiekėjo į naudotoją, arba atvirkščiai



Jei pakeičiame kintamojo “name” reikšmę: pakeitimas **synchronizuojamas** į teksto lauko reikšmę

Jei pakeičiame teksto lauko reikšmę: pakeitimas **synchronizuojamas** į kintamajį “name”

AngularJS

Vizualinė
dalis

Elgsenos
dalis

```
<div ng-app="myApp" ng-controller="myCtrl">  
  {{name}}  
</div>  
  
<script>  
var app = angular.module('myApp', []);  
app.controller('myCtrl', function($scope) {  
  $scope.name = "John Doe";  
});  
</script>
```

one way binding

UI komponentas

AngularJS

Vizualinė
dalis

```
<div ng-app="myApp" ng-controller="myCtrl">  
    Name: <input ng-model="name">  
</div>
```

Elgsenos
dalis

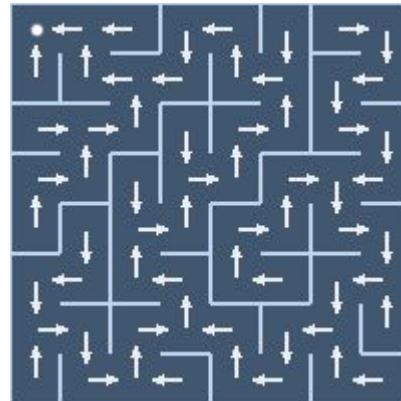
```
<script>  
var app = angular.module('myApp', []);  
app.controller('myCtrl', function($scope) {  
    $scope.name = "John Doe";  
});  
</script>
```

two way binding

UI komponentas

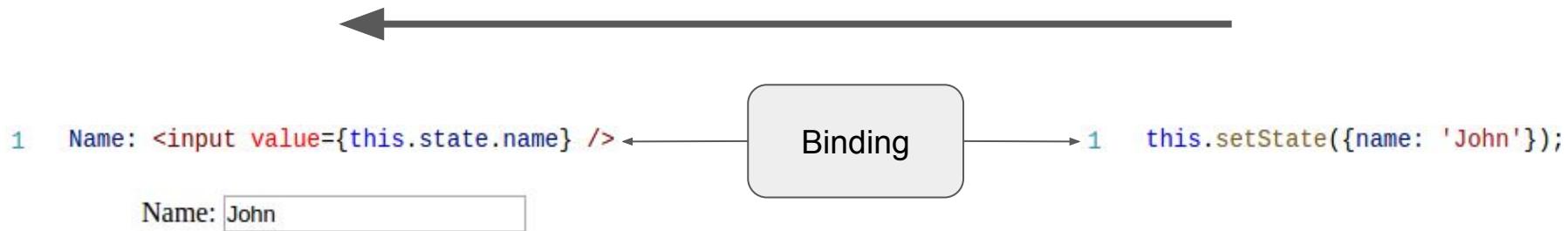
Dviejų pusiu surišimo (two way binding) problema

- Sunku valdyti ryšius tarp kelių dviejų pusiu surišimų
 - Sunku taisyti klaidas, kuomet kažkas nesynchronizuojama
 - Sunku taisyti klaidas, kuomet kažkas synchronizuojasi amžinai ("užsiciklina")



React surišimas (data binding)

Sinchronizuojama tik į vieną pusę:
iš komponento būsenos (duomenų) į vizualią dalį (vaizdą)



React surišimas (data binding)

```
1 import React from 'react';
2
3 export default class PingComponent extends React.Component {
4
5     constructor(props) {
6         super(props);
7         this.state = {
8             name: 'John'
9         };
10    }
11
12    render() {
13        return <div>
14            Name:<input value={this.state.name}>/>
15        </div>
16    }
17 }
```

React surišimas (data binding)

```
1 import React from 'react';
2
3 export default class PingComponent extends React.Component {
4
5     constructor(props) {
6         super(props);
7         this.state = {
8             name: 'John'
9         };
10    }
11
12    render() {
13        return <div>
14            Name:<input value={this.state.name} onChange={event => this.setState({name: event.target.value})}>
15        </div>
16    }
17 }
```



JSX - JavaScript išraiškos (expressions)

```
1 import React from 'react';
2
3 export default class PingComponent extends React.Component {
4
5     constructor(props) {
6         super(props);
7         this.state = {
8             name: 'John'
9         };
10    }
11
12    item() {
13        if (this.state.name !== 'John') {
14            return null;
15        }
16        return <p>Message for John</p>
17    }
18
19    render() {
20        return <div>
21            Name:<input value={this.state.name} onChange={event => this.setState({name: event.target.value})}>
22            {this.item()}
23        </div>
24    }
25 }
```

JSX - JavaScript išraiškos (expressions)

```
1 import React from 'react';
2
3 export default class PingComponent extends React.Component {
4
5     constructor(props) {
6         super(props);
7         this.state = {
8             name: 'John'
9         };
10    }
11
12    item() {
13        return <p>Message for John</p>
14    }
15
16    render() {
17        return <div>
18            Name:<input value={this.state.name} onChange={event => this.setState({name: event.target.value})} />
19            {this.state.name === 'John' ? this.item() : null}
20        </div>
21    }
22 }
```

JSX - įdėti komponentai (nested components)

```
1  export default class User extends React.Component {
2    render() {
3      return <h1>{this.props.name}</h1>
4    }
5  }
6
7  export default class Address extends React.Component {
8    render() {
9      return <p>{this.props.address}</p>
10   }
11 }
12
13 export default class Parent extends React.Component {
14
15   constructor(props) {
16     super(props);
17     this.state = {
18       name: 'Mykolas'
19       address: 'Vilnius'
20     };
21   }
22
23   render() {
24     return <div>
25       <User name={this.state.name} />
26       <Address address={this.state.address} />
27     </div>
28   }
29 }
```

JSX - Masyvo (array) panaudojimas

```
1 import React from 'react';
2
3 export default class PingComponent extends React.Component {
4
5     constructor(props) {
6         super(props);
7         this.state = {
8             relatives: [{ id: 1, name: 'Mike' }, { id: 2, name: 'Sofia' }, { id: 3, name: 'Dan' }]
9         };
10    }
11
12    render() {
13        return <div>
14            <ul>
15                {this.state.relatives.map(relative => {
16                    return <li key={relative.id}>{relative.name}</li>
17                })}
18            </ul>
19        </div>
20    }
21 }
```

DOM manipuliacijos su jQuery

```
<!DOCTYPE html>
<html>
<head>
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.1.1/jquery.min.js"></script>
</head>
<body>

<p>If you click on me, I will disappear.</p>
<p>Click me away!</p>
<p>Click me too!</p>

</body>
</html>
```

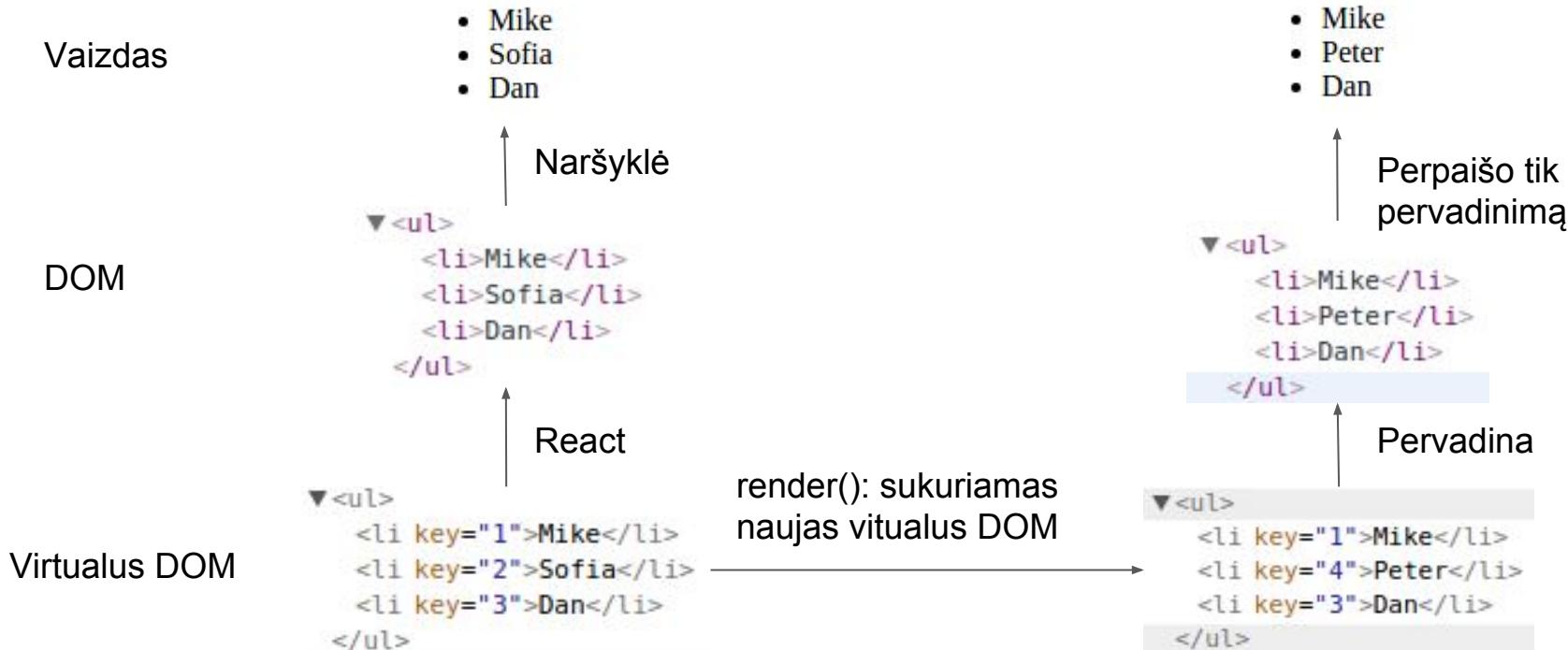
```
$(document).ready(() => {
    $("p").click(() => {
        $(this).hide();
    });
});
```

1. Ant paragrafo paspaudžiamas pelēs mygtukas
2. Surandamas norimas paragrafas
3. Nurodoma surastam paragrafui pasislėpti

Virtualus DOM

- DOM pakeitimai yra labai lėti, todėl dažnai reikia galvoti kaip padaryti kuo mažiau pakeitimų DOM'e, kad pasiektume tą patį rezultatą greičiau.
- React karkasas DOM atnaujina už mus per taip vadinamą "virtualų DOM":
 - Nors mums atrodo, kad kode manipuliuojame DOM'u, tačiau tai darome su "virtualiu DOM" - tikro DOM atspindys sudarytas iš React elementų
 - React karkasas automatiškai sinchronizuja pakeitimus tarp "virtualaus DOM" ir tikrojo DOM efektyviausiu būdu
 - Pvz.: jei keičiame sąrašą išimdami iš jo senus elementus ir juos pakeisdami naujais - React panaudoja trinamus DOM elementus jiems suteikdamas naujų elementų reikšmes, t.y. stengiasi trinti ir kurti kuo mažiau naujų elementų

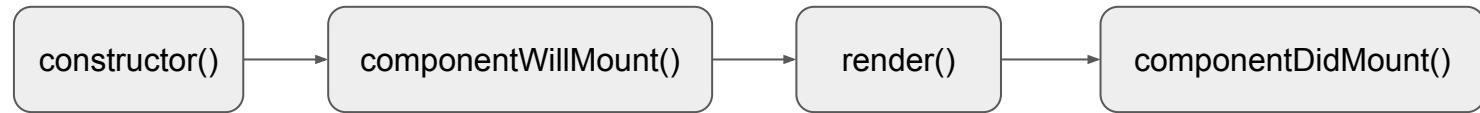
Virtualus DOM



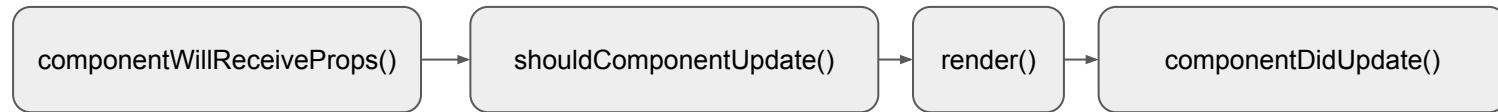
React komponento gyvavimo ciklas (lifecycle)

```
22  
23     render() {  
24         return <div>  
25             <User name={this.state.name} />  
26             <Address address={this.state.address} />  
27     </div>  
28 }
```

Prijungti (mount):



Atnaujinti (update):



Atjungti (unmount):



Komponento atnaujinamas

- Komponento atnaujinimas - render() metodo iškvietimas
 - Sukuriama nauja virtualaus DOM dalis
 - Virtualus DOM naudojamas DOM pakeisti
 - Naršyklė perpaišo pakeistą DOM vietą
- Atnaujinimas įvyksta kai:
 - paduodamas bet vienas komponento "props"
 - pakeičiamas komponento "state"

Single Page App'sas

- React app'sas yra atidaromas vieną kartą ir jį naudojant nereikia atidaryti vis kitų web puslapių:
 - duomenys parodomi greičiau (naujo puslapio atidarymas yra lėtas)
 - veikia greičiau, nes:
 - kolas taip pat paleidžiamas tik vieną kartą
 - būsenos nereikia išsaugoti ir vėl skaityti pereinant į kitą web puslapį

React Developer Tools

The screenshot shows a browser developer tools interface with the React tab selected. The left pane displays the application's UI, which includes a header bar with the title "Trading Manager" and a main content area with the heading "Welcome to Trading Manager". The right pane shows the React component tree for this page. The tree starts with a `<MuiThemeProvider muiTheme={...}>` component, followed by a `<Router history={getCurrentLocation: getCurrentLoc...}>`, and so on. The component tree highlights the `<AppBar title="Trading Manager" onLeftIconBu...>` component, indicating it is currently selected or being inspected.

TV ▾ 1152 × 864 100% ▾ :

Elements Console Sources Network Timeline Profiles React » ⌂ 1 | ⌂ X

(\$r in the console)

Trace React Updates Use Regular Expressions

AppBar

Trade Manager

Welcome to Trading Manager

Trading Manager is a tool for Administrators of Forex Broker Web System

Login

Press 'login' button on the top right corner of the page to provide Admin

Manage cache

Choose 'Cache' from the menu on the top left corner of a page to review

Manage trades

Choose 'Trades' from the menu on the top left corner of a page to view

Manage transactions

```
<MuiThemeProvider muiTheme={...}>
  <Router history={getCurrentLocation: getCurrentLoc...}>
    <RouterContext router={getCurrentLocation: getCurrentLoc...}>
      <AppComponent location={pathname: "/app", search: ""}>
        <div>
          <AppBar title="Trading Manager" onLeftIconBu...>
            <LoginComponent ref="loginComponent" location={...}>
              <Drawer open=false docked=false onRequestChang...>
                <div style={padding: "15px", fontFamily: "Ro...>
                  <WelcomeComponent location={pathname: "/app...>
                    </div>
                  </div>
                </AppComponent>
              </RouterContext>
            </Router>
          </MuiThemeProvider>
        <DialogInline title="Login" actions={[...], [...]} modal={...}>
      </AppComponent>
    </RouterContext>
  </Router>
</MuiThemeProvider>
```



T7 - Web sistemos duomenų bazė

Reliacinė duomenų bazė (RDB)

- Apibrėžta reliaciniu modeliu 1970 metais
- Reliacinis modelis paremtas matematika: aibų (set) ir logikos (logics) teorija
- Duomenų bazės normalizacija (normalization)
 - Duomenų kartojimui eliminuoti
 - 1, 2, 3, Boyce-Codd, ...



Structured Query Language (SQL)

- Data Definition Language (DDL)

- CREATE TABLE users (name varchar(255), age int)
- ALTER TABLE users ADD address varchar(1024)
- ALTER TABLE users DROP COLUMN name
- DROP TABLE users

- Data Manipulation Language (DML)

- SELECT name, age FROM users WHERE age = 33
- INSERT INTO users(name, age) VALUES ('Bob', 32)
- UPDATE users SET name = 'Sue' WHERE age = 24
- DELETE FROM users WHERE name = 'Bob'

- Data Control Language (DCL)

- GRANT SELECT, UPDATE ON users TO some_user, another_user
- REVOKE SELECT, UPDATE ON users FROM some_user, another_user

Duomenų bazės tranzakcija (Transaction)

- ACID
 - **Atomicity** - įvykdžius tranzakciją arba atliekami VISI pakeitimai arba NEI VIENAS
 - **Consistency** - kiekviena tranzakcija DB perveda iš vienos validžios būsenos į kitą
 - **Isolation** - vykdant tranzakciją atrodo, kad vykdoma tik ji viena
 - **Durability** - įvykdžius tranzakciją ji pasilieka įvykdyta, nepaisant dingusios elektros, PĮ užstrigimo ir t.t.

Reliacinės duomenų bazės tranzakcija (Transaction)

```
START TRANSACTION;  
UPDATE account SET balance=balance-10 WHERE id=1;  
UPDATE account SET balance=balance+10 WHERE id=2;  
COMMIT;
```

- ACID
 - **Atomicity** - apima visą DB
 - **Consistency** - YRA
 - **Isolation** - YRA
 - **Durability** - YRA

Reliacinės duomenų bazės schema

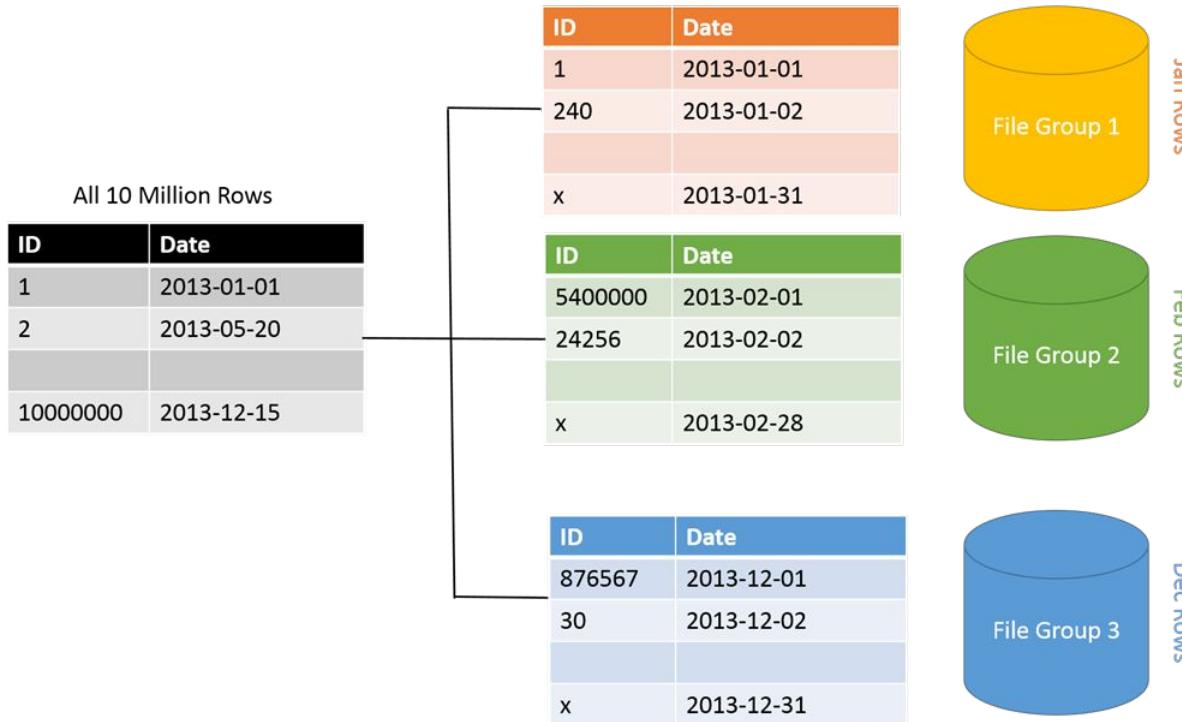


Duomenų bazės indeksas (index)

| users |
|--------------------------|
| <u>name</u> varchar(255) |
| age int |

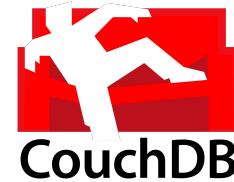
- Users lentelė turi
 - indeksuotą "name" stulpelį
 - neindeksuotą "age" stulpelį
- SELECT * FROM users WHERE name = 'Ann'
 - naudojamas "name" stulpelio indeksas - $O(\log n)$
 - jei $n = 1\ 000\ 000$, tuomet max sudėtingumas $\log n \approx 20$
- SELECT * FROM users WHERE age = 32
 - nenaudojamas indeksas - $O(n)$
 - jei $n = 1\ 000\ 000$, tuomet max sudėtingumas 1 000 000

Reliacinės DB padalinimas (partitioning)



NoSQL - 21 amžiaus mada

- Reiškia:
 - non SQL
 - non relational
 - not only SQL
- Kitokia duomenų schema (arba jos nebuvismas):
 - Stulpeliai (column) - ne eilutemės kaip iprasta - patogu dirbti su stulpelio duomenimis
 - Dokumentai (document) - medžio (tree) struktūra - patogu dirbti su hierarchiniais duomenimis
 - Raktas-reikšmė (key-value) - hash tipo duomenys (panašu į JavaScript objektą)
 - Grafai (graph) - duomenys: briaunos ir viršūnės (facebook'as)
- Dažniausia naudojama:
 - Dideliems duomenų kiekiams saugoti (lengva plėsti (scale))
 - Su realaus laiko sistemomis (greita ištraukti duomenis)





Redis

- Raktas-reikšmė (key-value) struktūra
 - Value gali būti tekstas (string), sąrašas (list), aibė (set),...
- Laikinosios atminties (in-memory) duomenų bazė
 - Galima kopijuoti duomenis į diską
- Leidžiama vienoje gijoje
- Naudojama kaip DB, cache arba žinučių brokeris (message broker)
- ACID
 - **Atomicity** - YRA (bet brangu, nes gija užsiblokuoja)
 - **Consistency** - YRA
 - **Isolation** - YRA (nes viena gija)
 - **Durability** - Gali būti (tačiau tuomet veikia lėtai)
- Indeksas = key



Redis komandos

- SET user1 Tomas
- SET user2 Ann
- GET user1
 - Tomas
- DEL user1

| key | value |
|------------|--------------|
| user1 | Tomas |
| user2 | Ann |



MongoDB

- Saugomi dokumentai
 - JSON (BSON)
- Skriptai rašomi JavaScript kalba
- ACID
 - **Atomicity** - VIENO DOKUMENTO APIMTYJE
 - **Consistency** - VIENO DOKUMENTO APIMTYJE
 - **Isolation** - VIENO DOKUMENTO APIMTYJE
 - **Durability** - YRA (with appropriate write concern)



MongoDB terminai

| SQL | MongoDB |
|-------------------|---------------------|
| database (schema) | database |
| table | collection |
| index | index |
| row | document |
| column | field |
| joining | linking & embedding |
| partition | shard |



MongoDB duomenų bazė (database)

- use DATABASE_NAME



MongoDB rinkinys (collection)

- db.createCollection('users')



MongoDB indeksas (index)

- db.users.ensureIndex({ "name":1 })



MongoDB dokumentas ir laukai (document and fields)

```
db.users.insert({  
    _id: ObjectId('7df78ad8902c'),  
    name: 'Ann',  
    description: 'A system user',  
    createdDate: 1231547566  
})
```



MongoDB susiejimas ir įdėjimas (linking and embedding)

```
{  
  _id: ObjectId('7df78ad8902c'),  
  name: 'Ann',  
  description: 'A system user',  
  createdDate: 1231547566,  
  accounts: [  
    { id: 1, balance: 123 },  
    { id: 2, balance: 654 }  
  ]  
}  
  
{  
  _id: ObjectId('7df78ad8902c'),  
  name: 'Ann',  
  description: 'A system user',  
  createdDate: 1231547566,  
  accounts: [  
    { id: ObjectId('7df78ad8902d') },  
    { id: ObjectId('7df78ad8902e') }  
  ]  
}
```

SQL prieš MongoDB

| | |
|--|--|
| CREATE TABLE users (name VARCHAR(128), age NUMBER) | db.createCollection("users") |
| INSERT INTO users VALUES ('Bob', 32) | db.users.insert({name: "Bob", age: 32}) |
| SELECT * FROM users | db.users.find() |
| SELECT name, age FROM users | db.users.find({}, {name: 1, age: 1, _id:0}) |
| SELECT name, age FROM users WHERE age = 33 | db.users.find({age: 33}, {name: 1, age: 1, _id:0}) |
| SELECT * FROM users WHERE age > 33 | db.users.find({age: {\$gt: 33}}) |
| SELECT * FROM users WHERE age <= 33 | db.users.find({age: {\$lte: 33}}) |
| SELECT * FROM users WHERE age > 33 AND age < 40 | db.users.find({age: {\$gt: 33, \$lt: 40}}) |
| SELECT * FROM users WHERE age = 33 OR name = 'Bob' | db.users.find({\$or:[{age:33}, {name:"Bob"}] }) |
| SELECT * FROM users WHERE age = 33 ORDER BY name ASC | db.users.find({age: 33}).sort({name: 1}) |
| SELECT * FROM users WHERE name LIKE '%Joe%' | db.users.find({name: /Joe/}) |
| SELECT * FROM users LIMIT 10 SKIP 20 | db.users.find().skip(20).limit(10) |

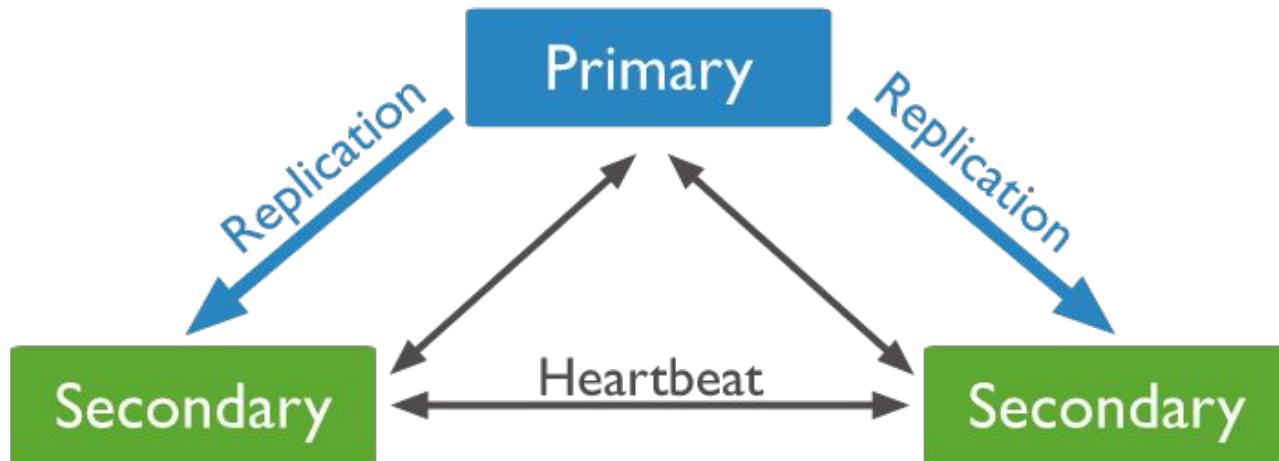


MongoDB - paskirstyta DB

- Paskirstyta DB - duomenys laikomi daugiau nei viename mazge (node)
 - Replikavimas (Replication) - duomenys kopijuojami tarp mazgų
 - Padalinimas (Sharding) - skirtinti duomenų rėžiai skirtinguose mazguose

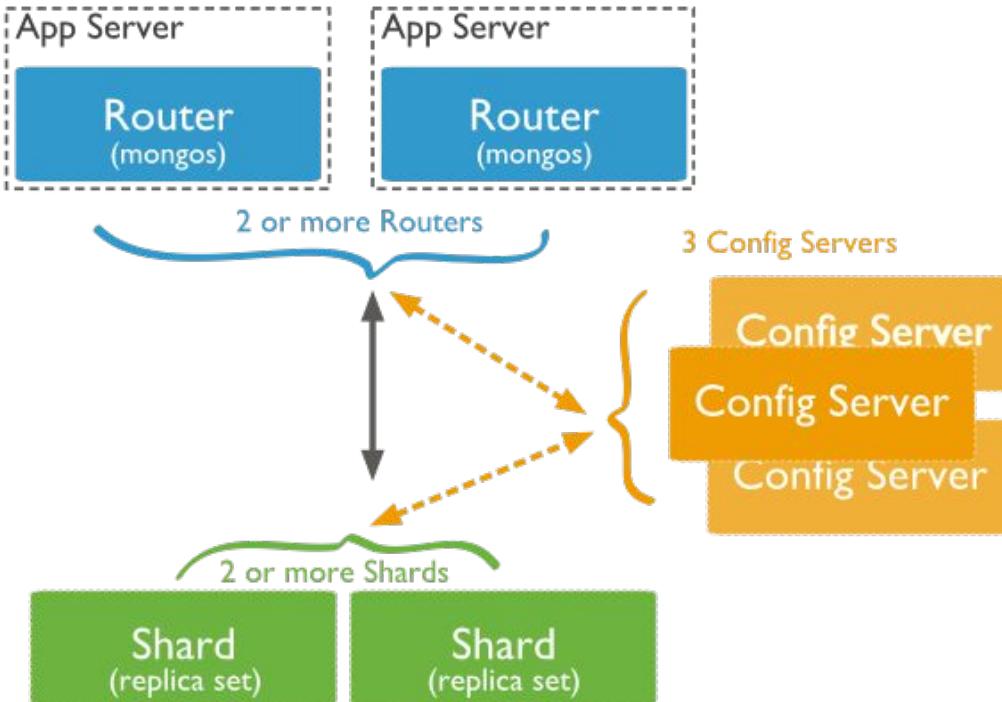


MongoDB - Replikavimas (Replication)





MongoDB - Padalinimas (Sharding)



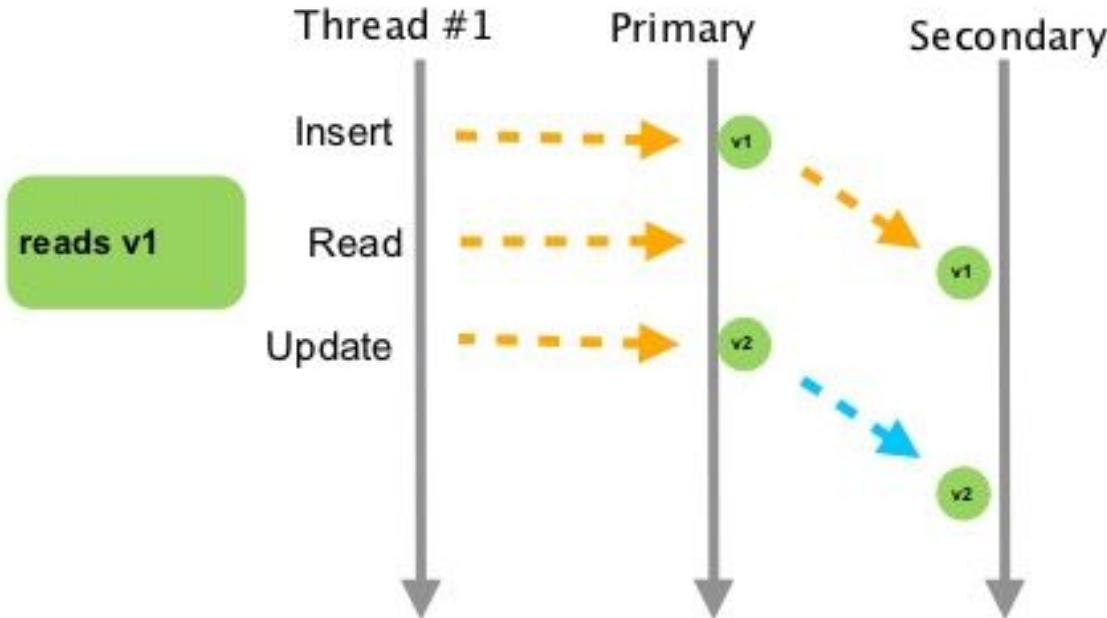


MongoDB - galiausia darna (eventual consistency)

- Galiausia darna (eventual consistency) - po pakeitimo galiausiai duomenys visuose mazguose bus darnūs (tokie kokie ir turi būti po įvykusio pakeitimo)
- Galima darant pakeitimą nurodyti kuomet pakeitimas laikomas pakankamai įvykdytas ir apie tai informuojama pakeitimą daranti pusė:
 - Pakeitimui įsigaliojus viename mazge
 - Pakeitimui įsigaliojus bent dviejuose mazguose
 - Pakeitimui įsigaliojus pusėje mazgų
 - Pakeitimui įsigaliojus visuose mazguose
 - ...



MongoDB - galiausia darna (eventual consistency)



Mongoose

```
1 var mongoose = require('mongoose');
2 mongoose.connect('mongodb://localhost/test');
3
4 var Cat = mongoose.model('Cat', { name: String });
5
6 var kitty = new Cat({ name: 'Zildjian' });
7 kitty.save((error) => {
8     if (error) {
9         return console.log(error);
10    }
11    console.log('meow');
12});
```

Mongoose schema ir modelis

```
1  var kittySchema = mongoose.Schema({
2      name: String,
3      address: String,
4      isSmall: Boolean,
5      age: Number
6  });
7
8  var Kitten = mongoose.model('Kitten', kittySchema);
9
10 var fluffy = new Kitten({
11     name: 'fluffy',
12     address: 'Vilnius',
13     isSmall: false,
14     age: 9
15});
```

Mongoose metodai

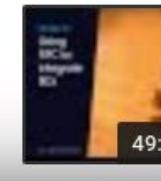
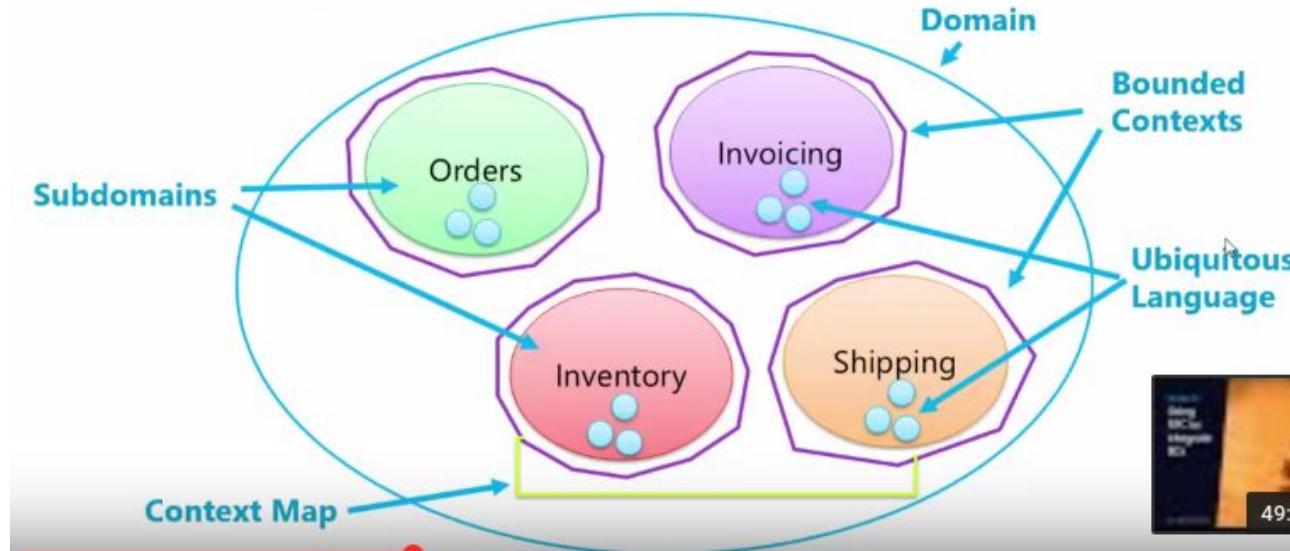
```
1  fluffy.save((error, fluffy) => {
2      if (error) {
3          return console.error(error);
4      }
5      // naudojame fluffy
6  });
7
8  Kitten.find((error, kittens) => {
9      if (error) {
10          return console.error(error);
11      }
12      console.log(kittens);
13  });
14
15 Kitten.find({ name: 'fluffy' }, (error, kitten) => {
16     if (error) {
17         return console.error(error);
18     }
19     console.log(kitten);
20 });
```



P4 - Sistemos projektavimo metodas

Sistemos projektavimo metodas - DDD

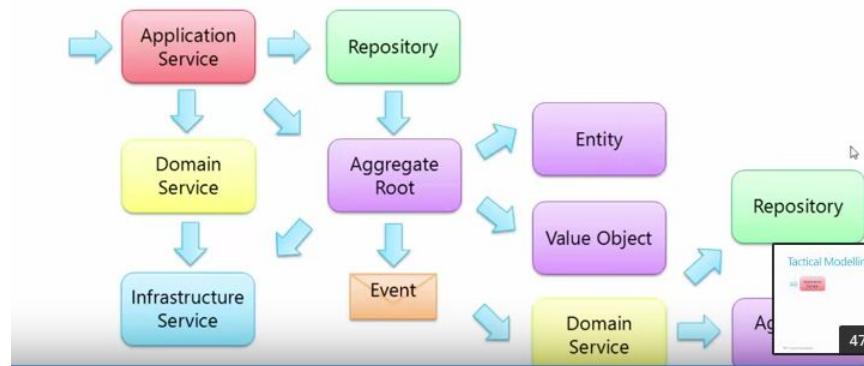
- Pj simuliuoja (in-sync) verslo sritj (domain)
- Reikalingas srities ekspertas (domain expert)



Sistemos projektavimo metodas - DDD

- Tactical Domain (“LEGO kaladėlės”)
 - Entities
 - Value Objects
 - Services
 - Domain Services
 - Application Services
 - Infrastructure Services
 - Aggregates (root, entity, entity ID)
 - Repositories - išsaugo visą agregatą
 - Domain Events

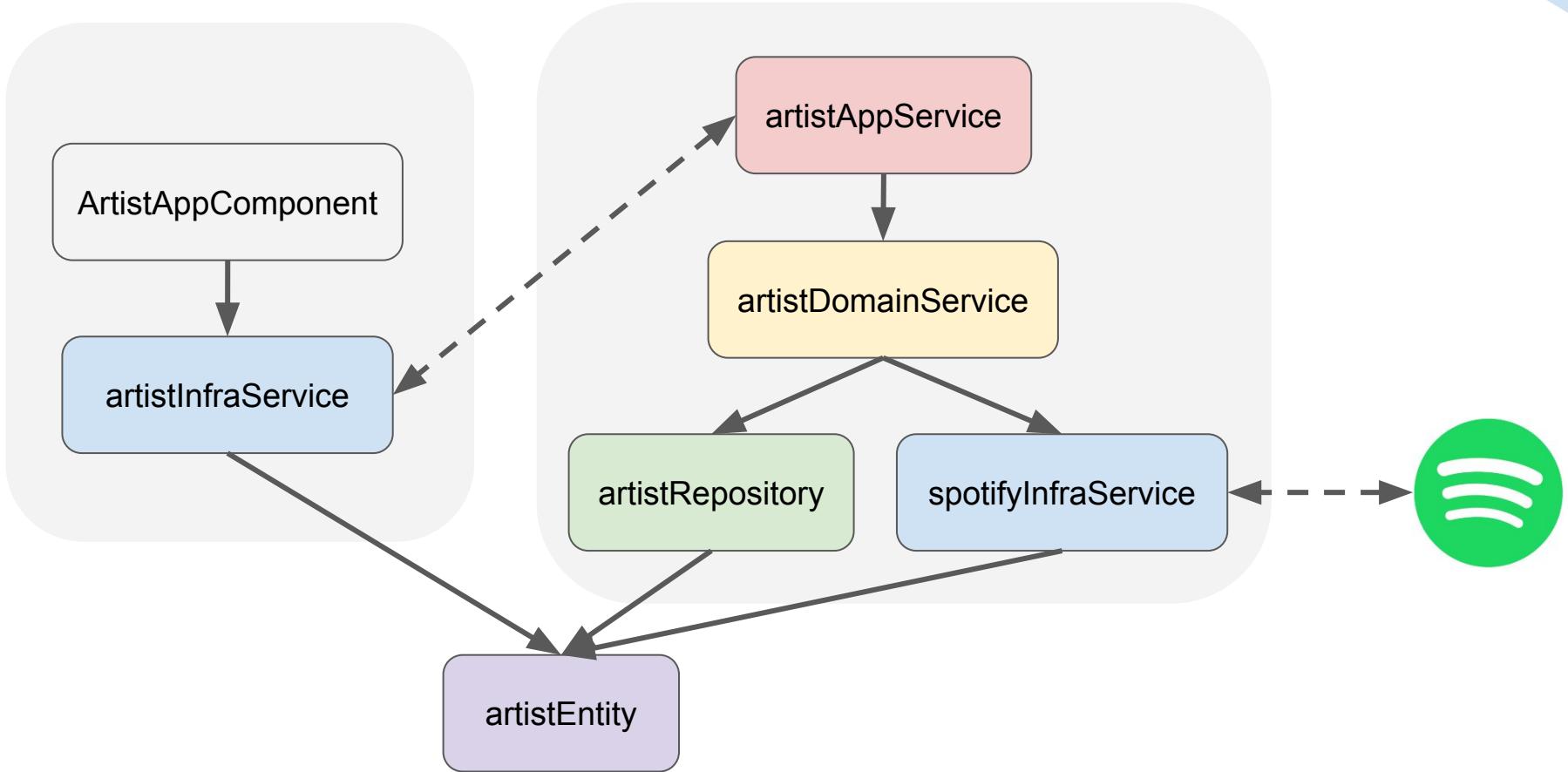
Tactical Modelling Summary



Kodo struktūros pavyzdys - 'Artist'

- **backend** // backend'o kodo direktorija
 - **index.js** // startinis backendo kodas
 - **artistAppService.js** // pateikia 'Artist' REST API
 - **artistDomainService.js** // apima 'Artist' srities logika
 - **artistRepository.js** // manipuliuoja 'Artist' esybe
 - **spotifyInfraService.js** // pasiekia 'Spotify' viešą paslaugą
- **frontend** // frontend'o kodo direktorija
 - **src**
 - **index.html** // pagrindinis HTML puslapis
 - **index.js** // startinis frontendo kodas
 - **ArtistAppComponent.jsx** // Šakninis React komponentas
 - **ArtistComponent.jsx** // 'Artist' esybės React komponentas
 - **artistInfraService.js** // pasiekia 'Artist' REST API
- **lib** // bendras kodas
 - **artistEntity.js** // 'Artist' esybė
- **.gitignore** // failai neįtraukiami į Git repozitorija
- **package-lock.json** // modulio priklausomybių raktas
- **package.json** // modulio nustatymai
- **README.md** // modulio dokumentacija
- **webpack.config.js** // WebPack nustatymai (įskaitant Babel)

Kodo struktūros pavyzdys - 'Artist'



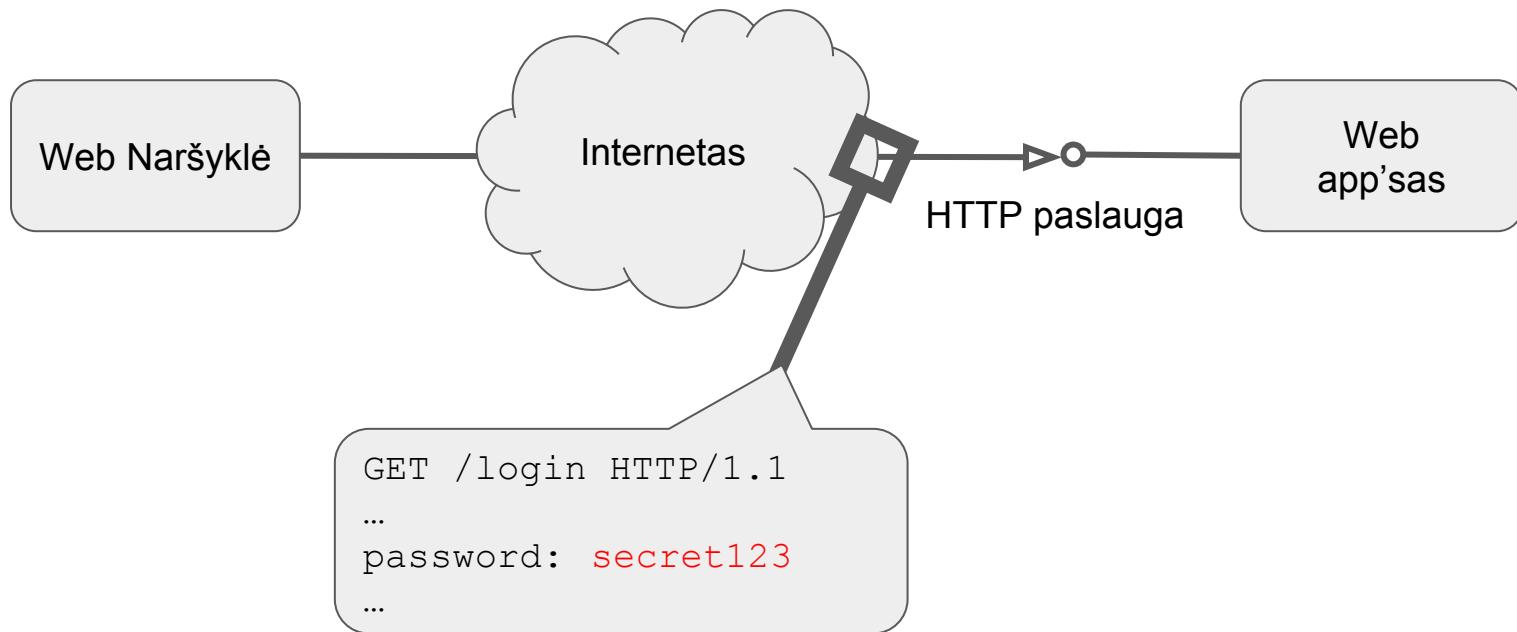


T8 - Web sistemos saugumas

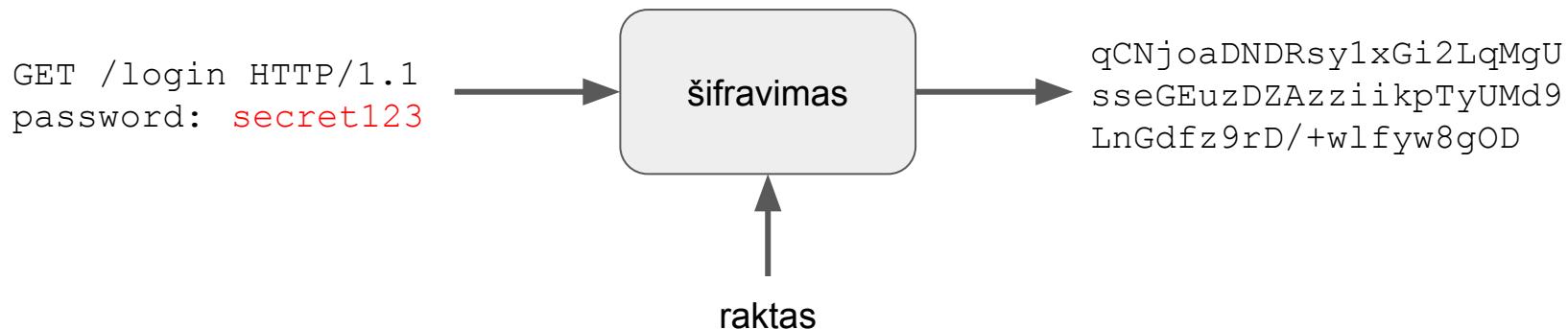


1988 metais “The Morris Worm”, pirmasis kompiuterinis virusas pasaulyje sukėlės masinius tinklo sutrikimus.

HTTP nėra saugus

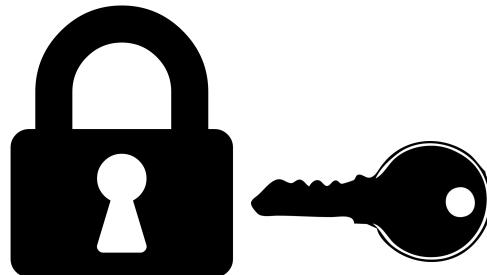


Būtina šifruoti HTTP siunčiamus duomenis



Simetrinė kriptavimo sistema

- Šifruojama ir iššifruojama naudojant tą patį raktą



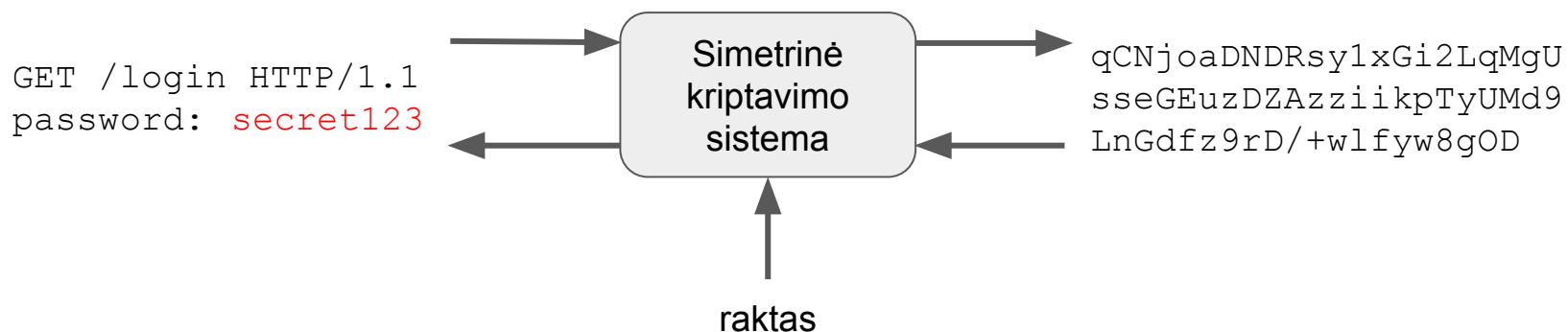
užrakinama



atrakinama

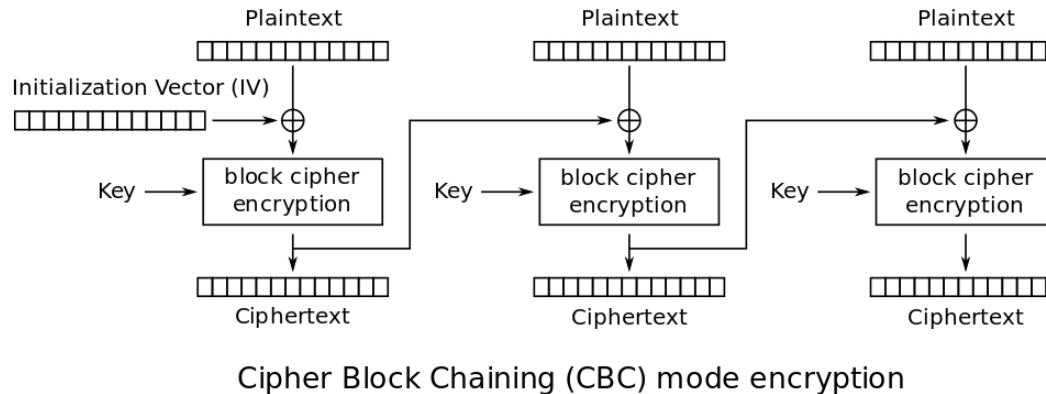
Simetrinė kriptavimo sistema

- Šifruojama ir iššifruojama naudojant tą patį raktą



AES - simetrinė kriptavimo sistema

- Raktas yra 128, 192 arba 256 bitų ilgio
- Šifruojama 128, 192 arba 256 bitų ilgio blokais
 - Duomenys suskaitomi į atitinkamo dydžio blokus
- 10 iki 14 šifravimo iteracijų (priklasomai nuo raktų ilgio)

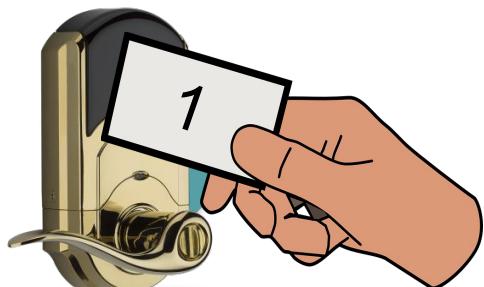


Simetrinės kriptavimo sistemos neužtenka

- Galima saugiai perdavinėti šifruotus duomenis
- Iššifruoti duomenims reikalingas raktas
- Raktą reikia saugiai perduoti gavėjui
- Reikalingas būdas saugiai perduoti raktą

Asimetrinė kriptavimo sistema

- Šifruojama vienu raktu, o iššifruojama kitu raktu



užrakinama



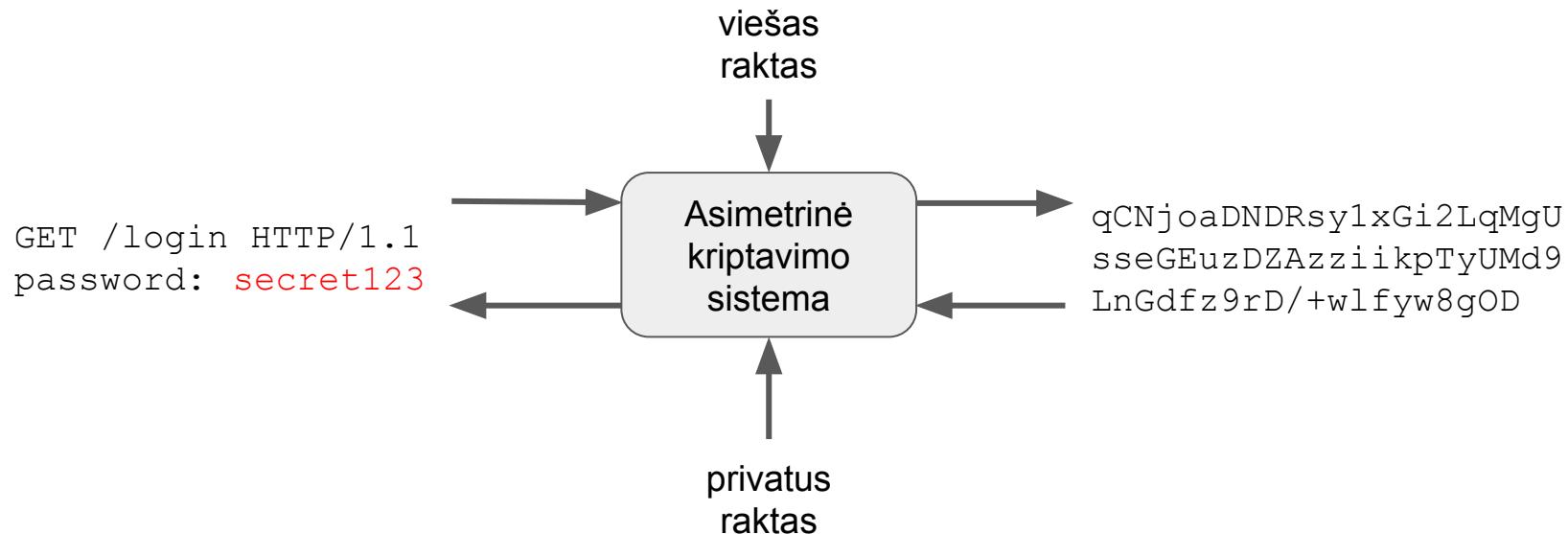
atrakinama



atrakinama

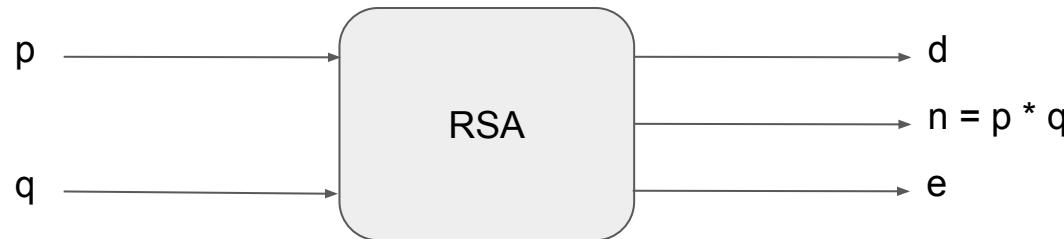
Viešojo raktų kriptavimo sistema

- Šifruojama vienu raktu, o iššifruojama kitu raktu



RSA - viešojo rakto kriptavimo sistema

- Paremta milžiniškais pirminiais skaičiais ir dalybos liekanomis



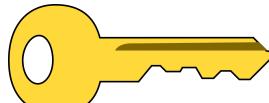
$$c = s^e \bmod n$$

$$s = c^d \bmod n$$

Viešas, privatus raktai ir sertifikatas

- Išduoda sertifikavimo centras (Certification Authority (CA))

VIEŠA



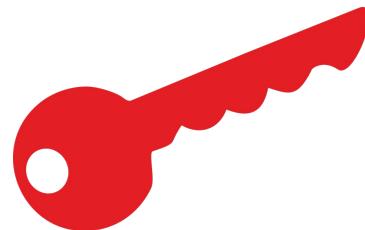
viešas raktas



sertifikatas

sertifikato
dalis

išduoda ir
pasirašo



privatus raktas

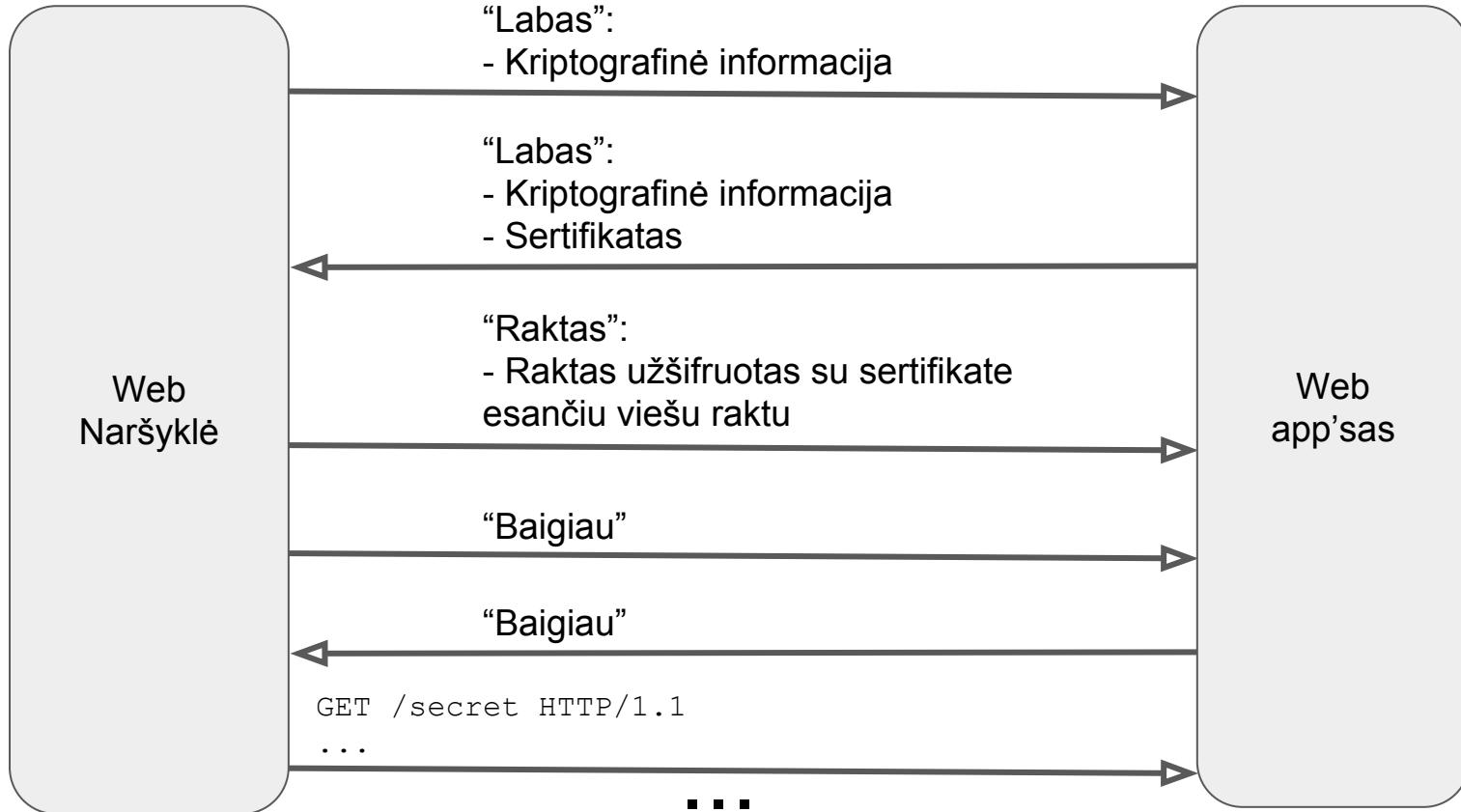
PRIVATU

išduoda

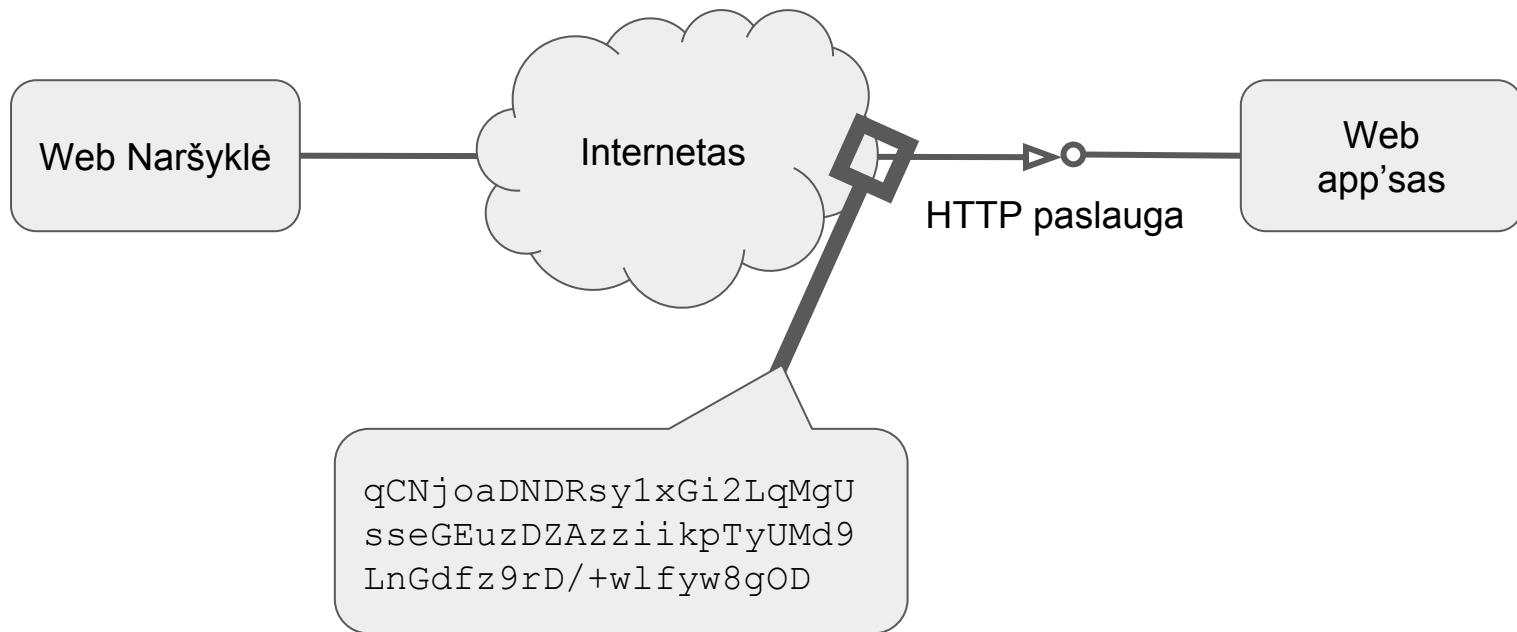
HTTPS

- HTTP virš SSL, TLS, arba tiesiog Secure
 - Kaip ir FTPS, Secure mongo, WSS (Web Socket Secure)
- SSL - Secure Socket Layer
 - yra nenaudojamas dėl saugumo problemų!
- TLS - Transport Layer Security
 - dabartinė versija 1.2
 - ruošiama versija 1.3

TLS 1.2



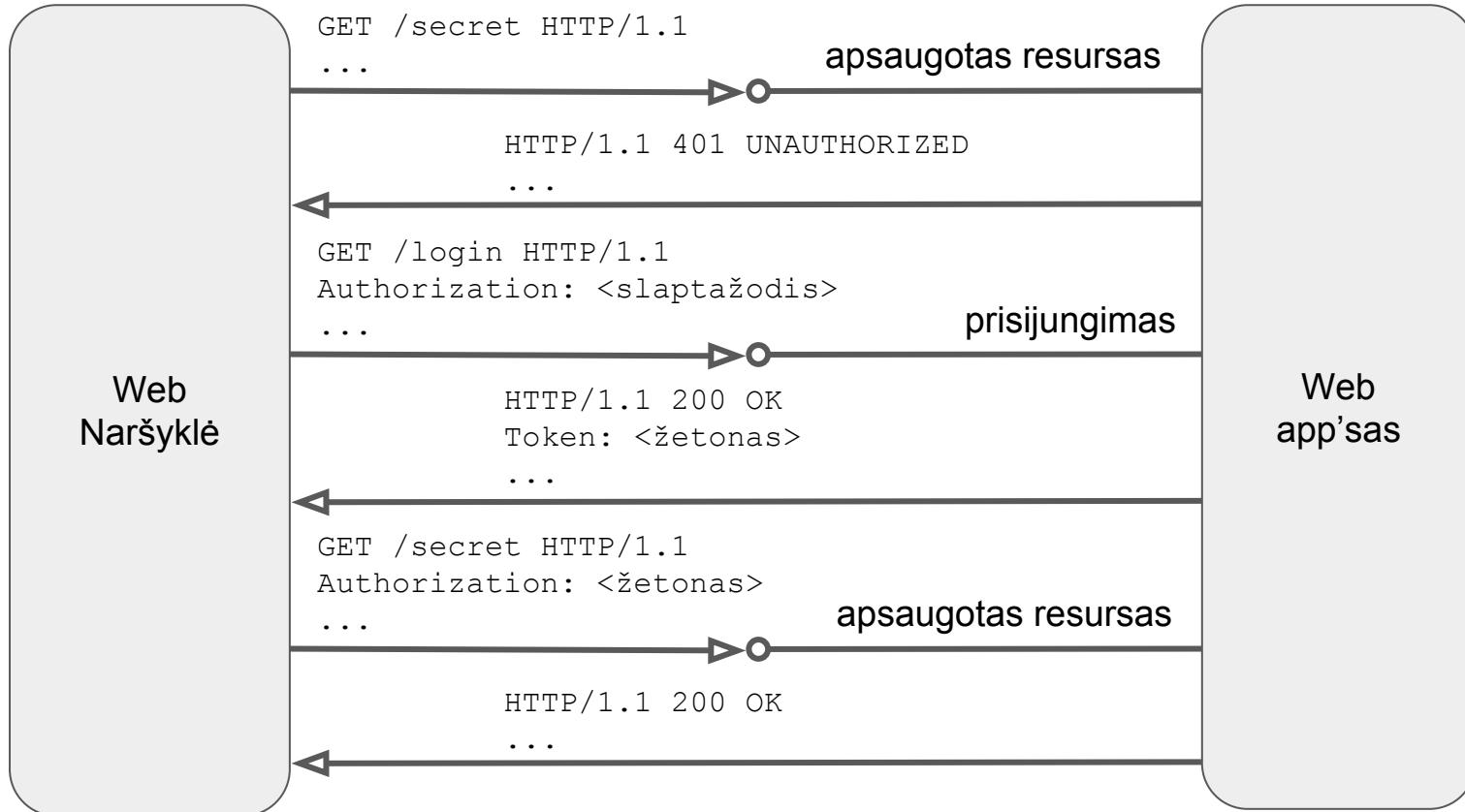
HTTPS yra saugus



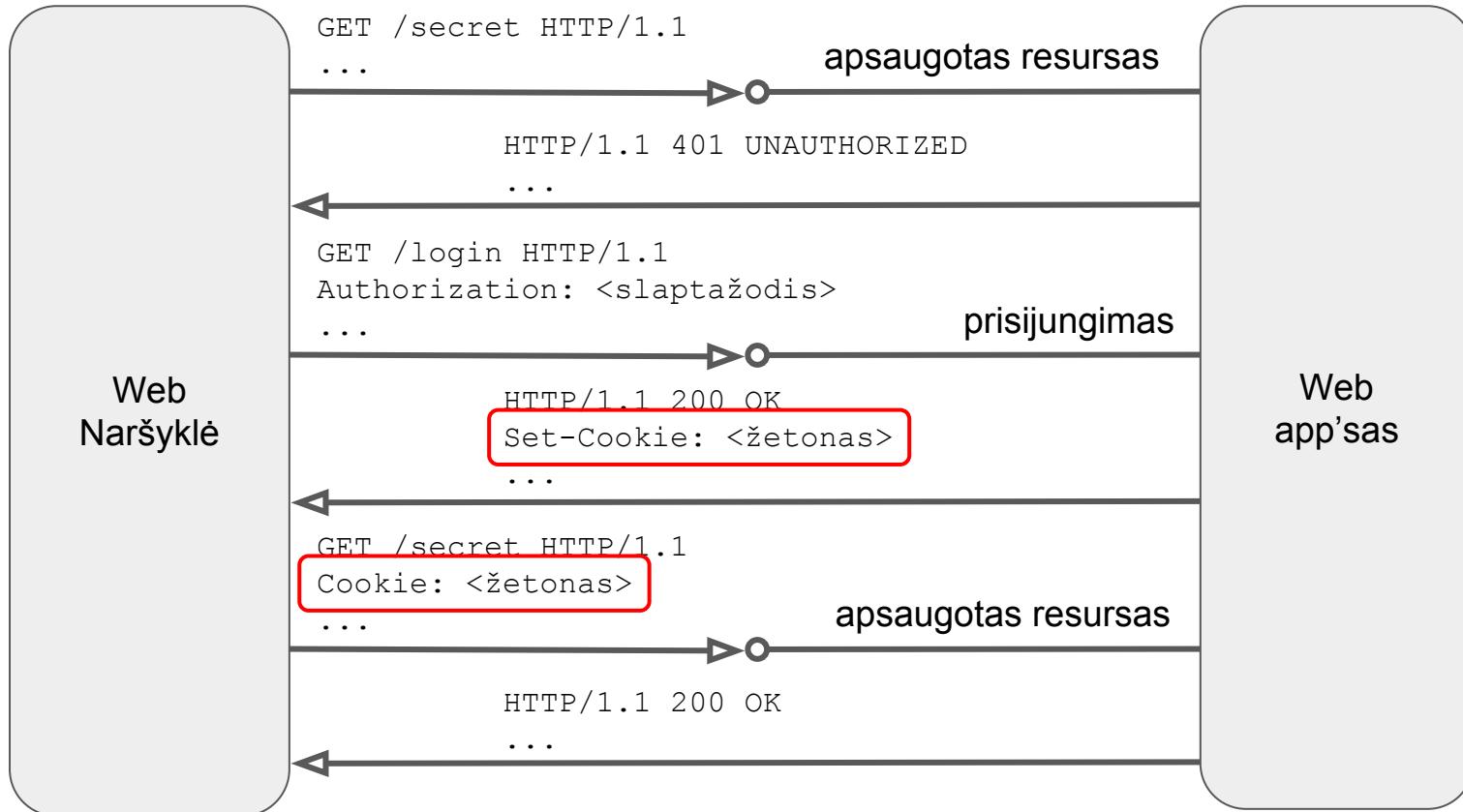
HTTPS derinimas su Express

```
1 const https = require('https');
2 const express = require('express');
3
4 const app = express();
5
6 const privateKey = fs.readFileSync(`./privatekey.pem`);
7 const certificate = fs.readFileSync(`./certificate.pem`);
8
9 const server = https.createServer({ key: privateKey, cert: certificate }, app);
10
11 server.listen(3001);
```

Web autorizacija



Web autorizacija su sausainiukais (cookies)

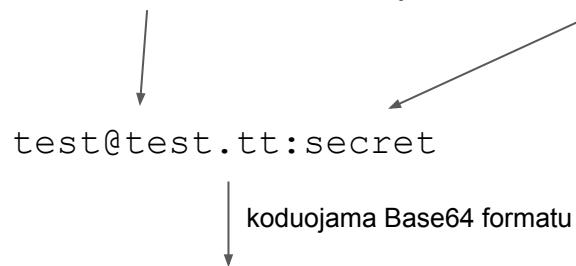


Sausainiukai (cookies)

- Turi formatą: raktas=reikšmė (key=value)
- Reikšmė užkoduojama Base64 formatu
 - Pvz.: Set-Cookie: token=Rg3vHJZnehYLjVg7qi3bZjzg
- Naršyklė sausainiukus (cookies) išsaugo atmintyje
- Sausainiukus naršyklė automatiškai įkelia į užklausos antraštę
 - Jei užklausa vykdoma į tą patį domeną (domain) ir/arba kelią (path)
 - Pvz.: jei sausainiukas buvo gautas iš example.com domeno, tuomet jei vykdoma užklausa į test.tt domeną - sausainiukas nebus įkeltas į antraštę, jei į example.com - bus
- Gali turėti galiojimo laiką (expiration time), po kurio yra automatiškai pašalinami iš naršyklės

HTTP Basic autorizacija

Naudotojas: "test@test.tt", slaptažodis: "secret"



GET /login HTTP/1.1

...

Authorization: Basic dGVzdEB0ZXN0LnR0OnNlY3JldA==

...

HTTP Basic autorizacija su Passport

```
1  const passport = require('passport');
2  const { BasicStrategy } = require('passport-http');
3
4  passport.use(new BasicStrategy((userid, password, done) => {
5      User.findOne(userId, (error, user) => {
6          if (error) {
7              return done(error);
8          }
9          if (!user) {
10              return done(null, false);
11          }
12          if (!user.verifyPassword(password)) {
13              return done(null, false);
14          }
15          return done(null, user);
16      });
17  }));
18
19  app.get('/login', passport.authenticate('basic', { session: false }), (req, res) => { {
20      // sukuriamas ir grąžinamas žetonas (token)
21  });
22  ...
```

JWT autorizacija

Encoded PASTE A TOKEN HERE

```
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJpZ  
CI6IjEyMyIsImVtYWlsIjoidGVzdEB0ZXN0LnR0In0  
.KXxuENUHeA-  
aDsceymtX3cDW8J0IXIL3DZOHTAlt5Ec
```

Decoded EDIT THE PAYLOAD AND SECRET (ONLY HS256 SUPPORTED)

HEADER: ALGORITHM & TOKEN TYPE

```
{  
  "alg": "HS256",  
  "typ": "JWT"  
}
```

PAYOUT: DATA

```
{  
  "id": "123",  
  "email": "test@test.tt"  
}
```

VERIFY SIGNATURE

```
HMACSHA256(  
  base64UrlEncode(header) + "." +  
  base64UrlEncode(payload),  
  secret  
)  secret base64 encoded
```

Pasirašoma:
HEADER ir PAYLOAD

HTTP/1.1 200 OK

Token: eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJp...

...

GET /secret HTTP/1.1

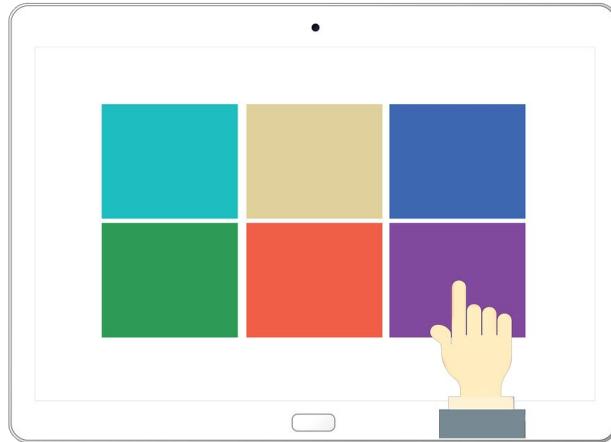
...

Authorization: JWT eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJp...

...

JWT autorizacija su Passport

```
1 const passport = require('passport');
2 const { Strategy, ExtractJwt } = require('passport-jwt').Strategy;
3
4 passport.use(new Strategy({ jwtFromRequest: ExtractJwt.fromAuthHeader(), secretOrKey: 'secret' }, (payload, done) => {
5     User.findOne(payload.id, (error, user) => {
6         if (error) {
7             return done(error);
8         }
9         if (!user) {
10             return done(null, false);
11         }
12         return done(null, user);
13     });
14 }));
15
16 app.post('/secret', passport.authenticate('jwt', { session: false }), (req, res) => {
17     // grąžinamas apsaugotas resursas
18 });
```

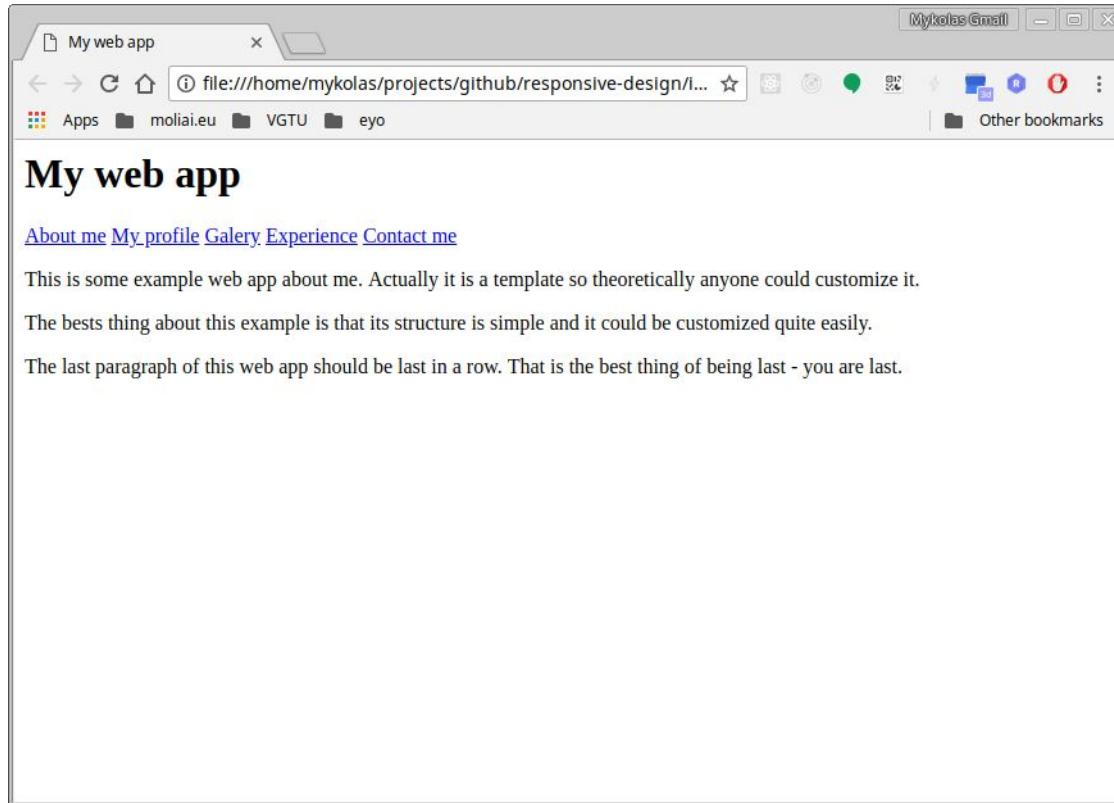


T9 - Web sistemos naudotojo patirtis

Cascading Style Sheets (CSS)

```
1  <html>
2    <head>
3      <title>My web app</title>
4    </head>
5    <body>
6      <div id="header">
7        <h1>My web app</h1>
8      </div>
9      <div id="menu">
10        <a href="#">About me</a>
11        <a href="#">My profile</a>
12        <a href="#">Galery</a>
13        <a href="#">Experience</a>
14        <a href="#">Contact me</a>
15      </div>
16      <div id="content">
17        <p>This is some example web app about me. Actually it is a template so theoretically anyone could customize it.</p>
18        <p>The bests thing about this example is that its structure is simple and it could be customized quite easily.</p>
19        <p>The last paragraph of this web app should be last in a row. That is the best thing of being last - you are last.</p>
20      </div>
21    </body>
22  </html>
```

Cascading Style Sheets (CSS)

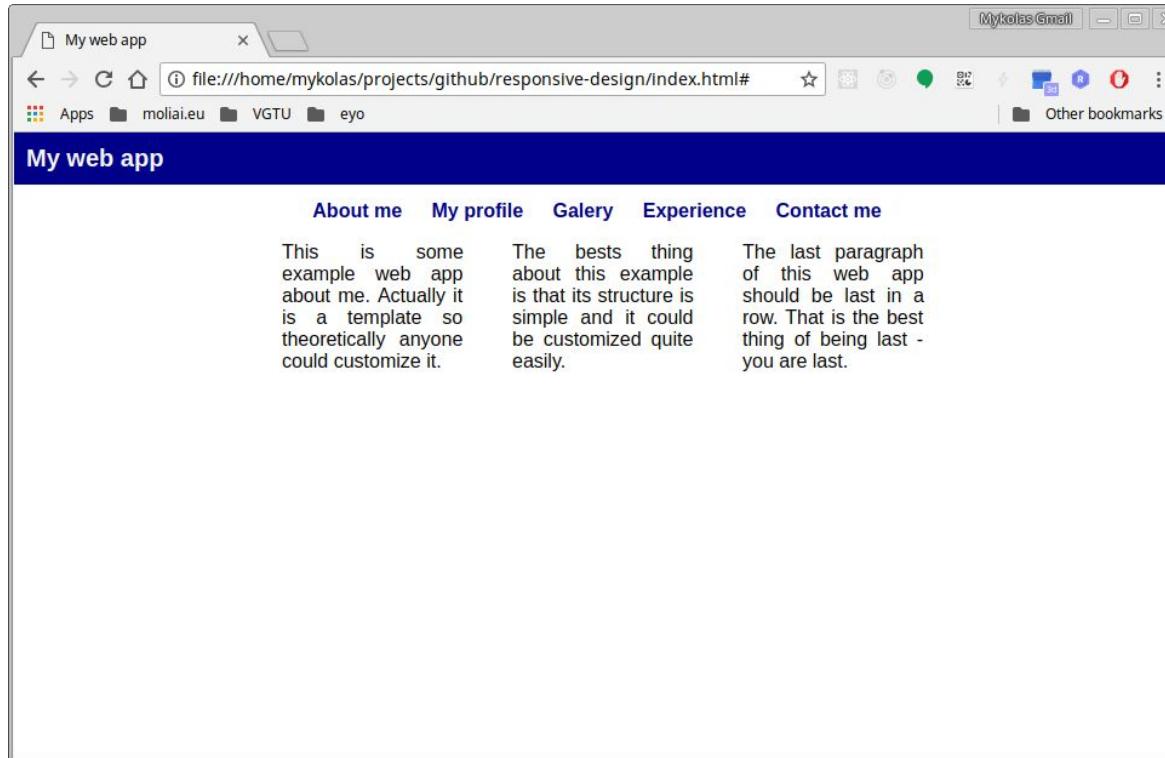


Cascading Style Sheets (CSS)

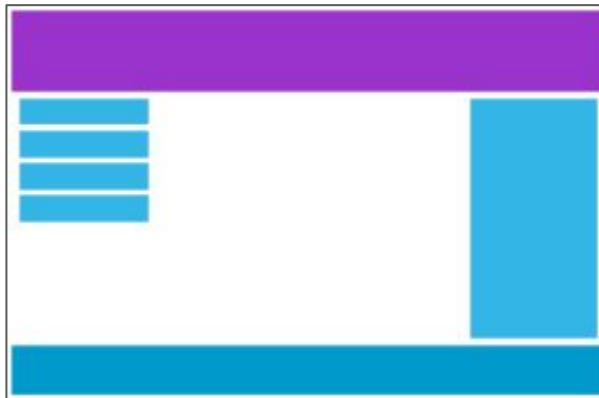
```
1 <html>
2   <head>
3     <title>My web app</title>
4     <link rel="stylesheet" type="text/css" href="style.css">
5   </head>
6   <body>
7     <div id="header">
8       <h1>My web app</h1>
9     </div>
10    <div id="menu">
11      <a href="#">About me</a>
12      <a href="#">My profile</a>
13      <a href="#">Galery</a>
14      <a href="#">Experience</a>
15      <a href="#">Contact me</a>
16    </div>
17    <div id="content">
18      <p>This is some example web app about me. Actually it is a template s
19      <p>The bests thing about this example is that its structure is simple
20      <p>The last paragraph of this web app should be last in a row. That i
21    </div>
22  </body>
23 </html>
```

```
1 body {
2   margin: 0;
3   font-family: Arial, Helvetica, sans-serif;
4 }
5 #header {
6   background-color: darkblue;
7 }
8 #header h1 {
9   color: whitesmoke;
10  padding: 10px;
11  font-size: 20px;
12 }
13 #menu {
14   width: 600px;
15   margin-left:auto;
16   margin-right:auto;
17   text-align: center;
18 }
19 #menu a {
20   padding: 10px;
21   font-weight: bold;
22   font-size: 16px;
23   color: darkblue;
24   text-decoration: none;
25 }
26 #content {
27   width: 600px;
28   margin-left:auto;
29   margin-right:auto;
30 }
31 #content p {
32   float: left;
33   width: 25%;
34   padding-left: 40px;
35   text-align: justify;
36 }
```

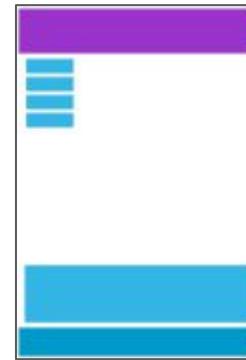
Cascading Style Sheets (CSS)



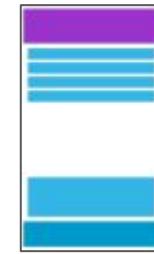
Prisitaikantis dizainas (Responsive web design)



Kompiuteris



Planšetė



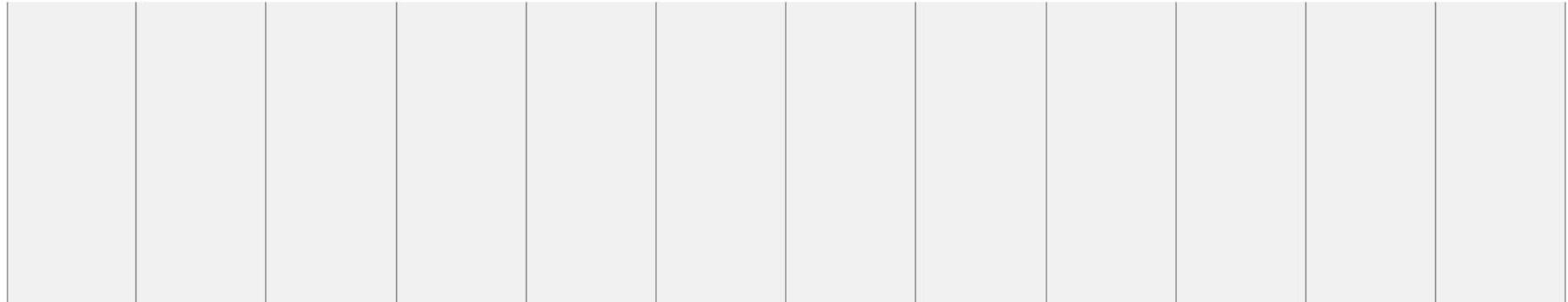
Telefonas

Prisitaikantis dizainas (Responsive web design)

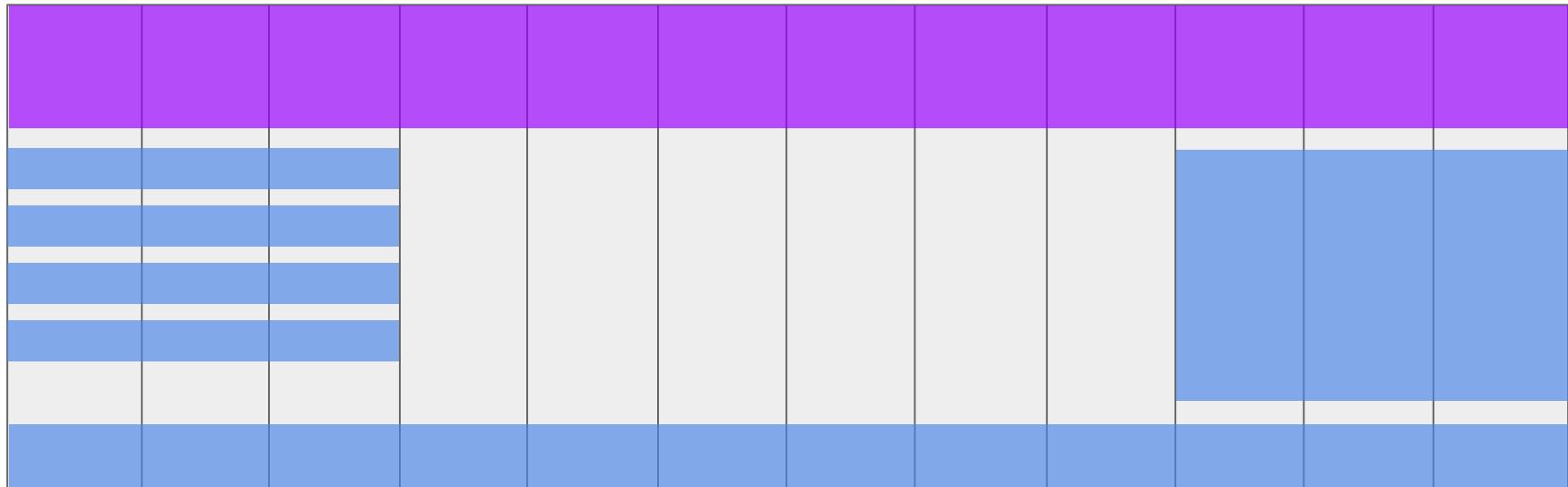
```
<meta name="viewport" content="width=device-width, initial-scale=1.0">
```



Prisitaikantis dizainas (Responsive web design)

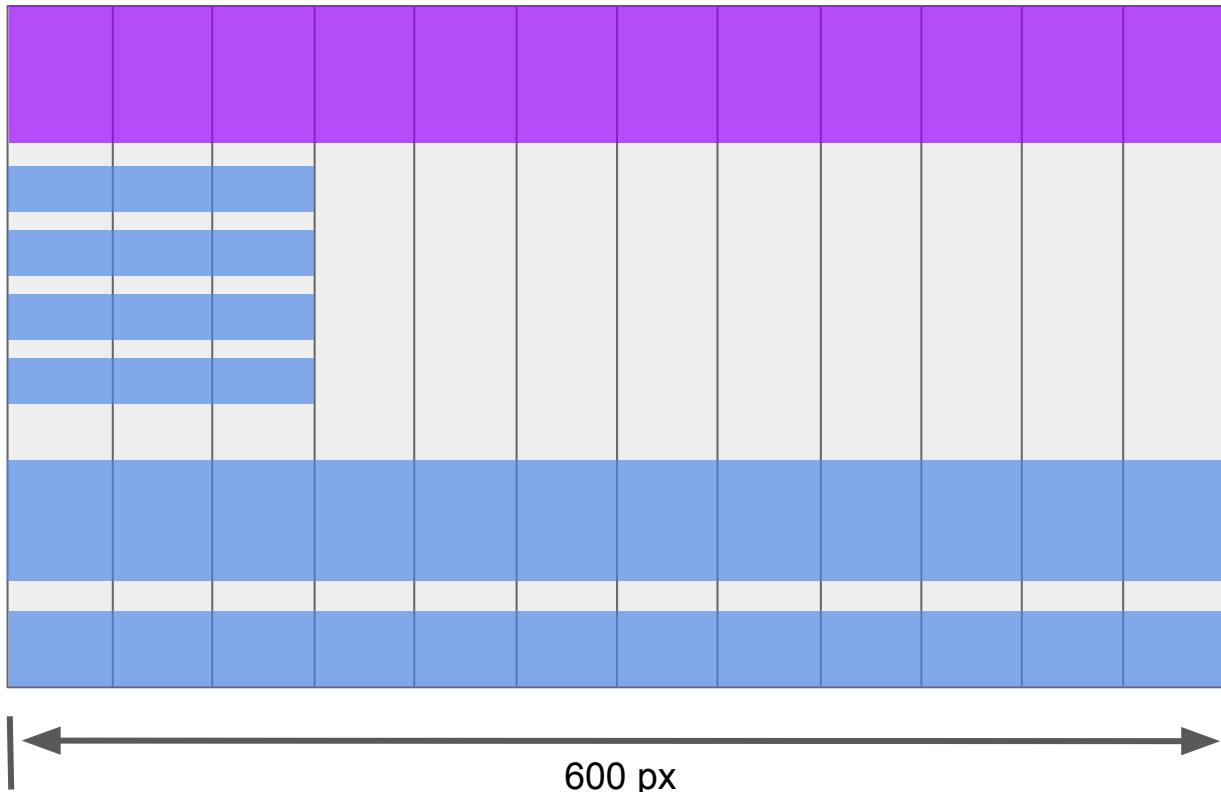


Prisitaikantis dizainas (Responsive web design)

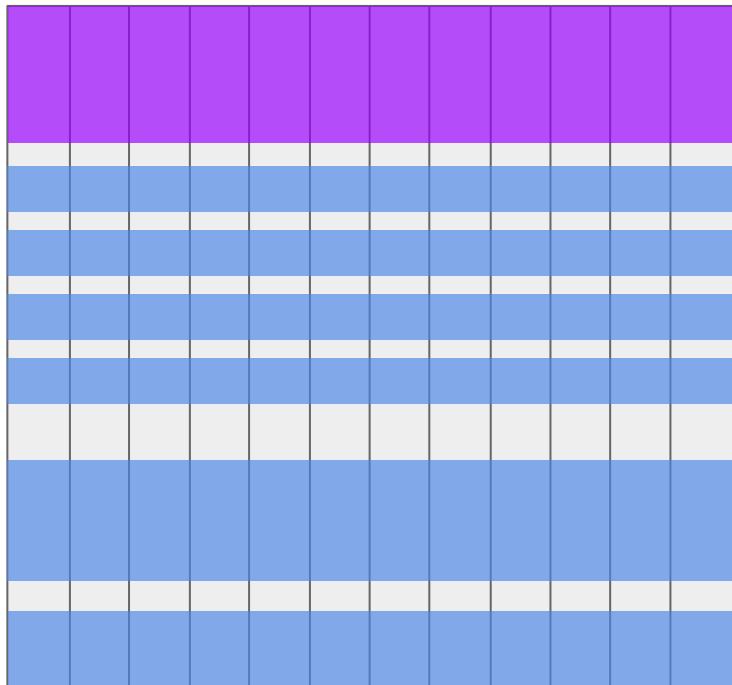


900 px

Prisitaikantis dizainas (Responsive web design)



Prisitaikantis dizainas (Responsive web design)



300 px

Prisitaikantis dizainas (Responsive web design)

```
<div class="row">
  <div class="col-3">...</div> <!-- 25% -->
  <div class="col-9">...</div> <!-- 75% -->
</div>
```

```
.col-1 {width: 8.33%;}
.col-2 {width: 16.66%;}
.col-3 {width: 25%;}
.col-4 {width: 33.33%;}
.col-5 {width: 41.66%;}
.col-6 {width: 50%;}
.col-7 {width: 58.33%;}
.col-8 {width: 66.66%;}
.col-9 {width: 75%;}
.col-10 {width: 83.33%;}
.col-11 {width: 91.66%;}
.col-12 {width: 100%;}
```

Prisitaikantis dizainas (Responsive web design)

```
/* Extra small devices (phones, 600px and down) */
@media only screen and (max-width: 600px) {...}

/* Small devices (portrait tablets and large phones, 600px and up) */
@media only screen and (min-width: 600px) {...}

/* Medium devices (landscape tablets, 768px and up) */
@media only screen and (min-width: 768px) {...}

/* Large devices (laptops/desktops, 992px and up) */
@media only screen and (min-width: 992px) {...}

/* Extra large devices (large laptops and desktops, 1200px and up) */
@media only screen and (min-width: 1200px) {...}
```

Bootstrap

The screenshot shows a web browser window with a dark-themed header bar. The address bar displays a secure connection to <https://getbootstrap.com/docs/4.1/examples/jumbotron/>. The main content area features a large, light-colored callout (jumbotron) with the text "Hello, world!". Below it, a paragraph of descriptive text and a blue "Learn more »" button are visible. The page is divided into three columns, each containing a heading and a paragraph of text, with a "View details »" button at the bottom of each section. The footer contains a copyright notice.

B Jumbotron Template for x

Secure | <https://getbootstrap.com/docs/4.1/examples/jumbotron/>

Apps moliai.eu VGTU eyo

Mykolas Gmail

Navbar Home Link Disabled Dropdown Search

Search

Hello, world!

This is a template for a simple marketing or informational website. It includes a large callout called a jumbotron and three supporting pieces of content. Use it as a starting point to create something more unique.

[Learn more »](#)

Heading

Donec id elit non mi porta gravida at eget metus. Fusce dapibus, tellus ac cursus commodo, tortor mauris condimentum nibh, ut fermentum massa justo sit amet risus. Etiam porta sem malesuada magna mollis euismod. Donec sed odio dui.

[View details »](#)

Heading

Donec id elit non mi porta gravida at eget metus. Fusce dapibus, tellus ac cursus commodo, tortor mauris condimentum nibh, ut fermentum massa justo sit amet risus. Etiam porta sem malesuada magna mollis euismod. Donec sed odio dui.

[View details »](#)

Heading

Donec sed odio dui. Cras justo odio, dapibus ac facilisis in, egestas eget quam. Vestibulum id ligula porta felis euismod semper. Fusce dapibus, tellus ac cursus commodo, tortor mauris condimentum nibh, ut fermentum massa justo sit amet risus.

[View details »](#)

© Company 2017-2018

React Bootstrap

The screenshot shows a web browser window displaying the React-Bootstrap documentation for buttons. The URL in the address bar is <https://react-bootstrap.github.io/components/buttons/>. The browser interface includes a toolbar with icons for back, forward, search, and other functions, and a sidebar with bookmarks for 'Apps', 'molai.eu', 'VGTU', and 'eyo'. The main content area has a dark header with the 'React-Bootstrap' logo and navigation links for 'Documentation' and 'GitHub'. Below the header, a large white section features the word 'Components' in a large, bold, sans-serif font. To the left of the main content, a sidebar menu lists various components: 'Getting started', 'Layout', 'Components' (which is expanded to show 'Alerts', 'Badge', 'Breadcrumb', 'Buttons' - this item is highlighted with a purple vertical bar, 'Button Group', 'Carousel', 'Dropdowns', 'Forms', 'Glyphicons', 'Images', 'Jumbotron', and 'Label'). The 'Buttons' section contains sub-links for 'Default', 'Primary', 'Success', 'Info', 'Warning', 'Danger', and 'Link'. The main content area also includes a 'EXAMPLE' section with a code snippet and a preview of the button styles.

Buttons Button

Options

Use any of the available button style types to quickly create a styled button. Just modify the `bsStyle` prop.

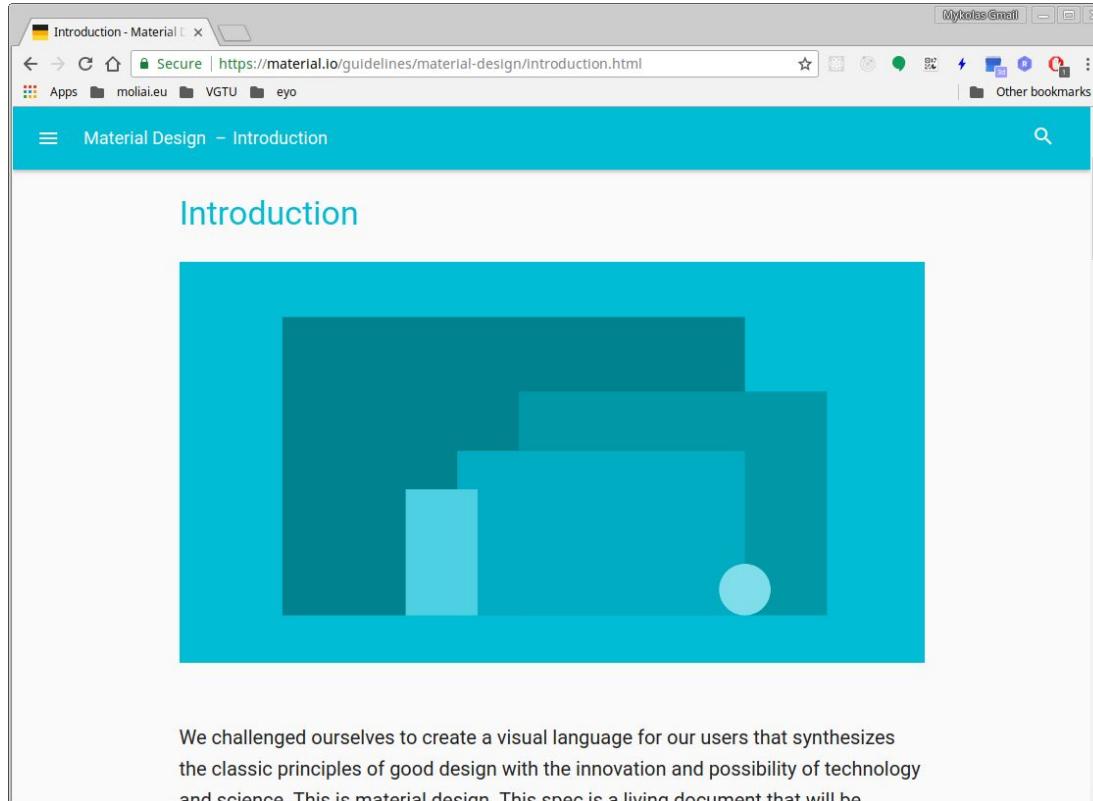
EXAMPLE

Default Primary Success Info Warning Danger Link

```
<ButtonToolbar>
  /* Standard button */
  <Button>Default</Button>

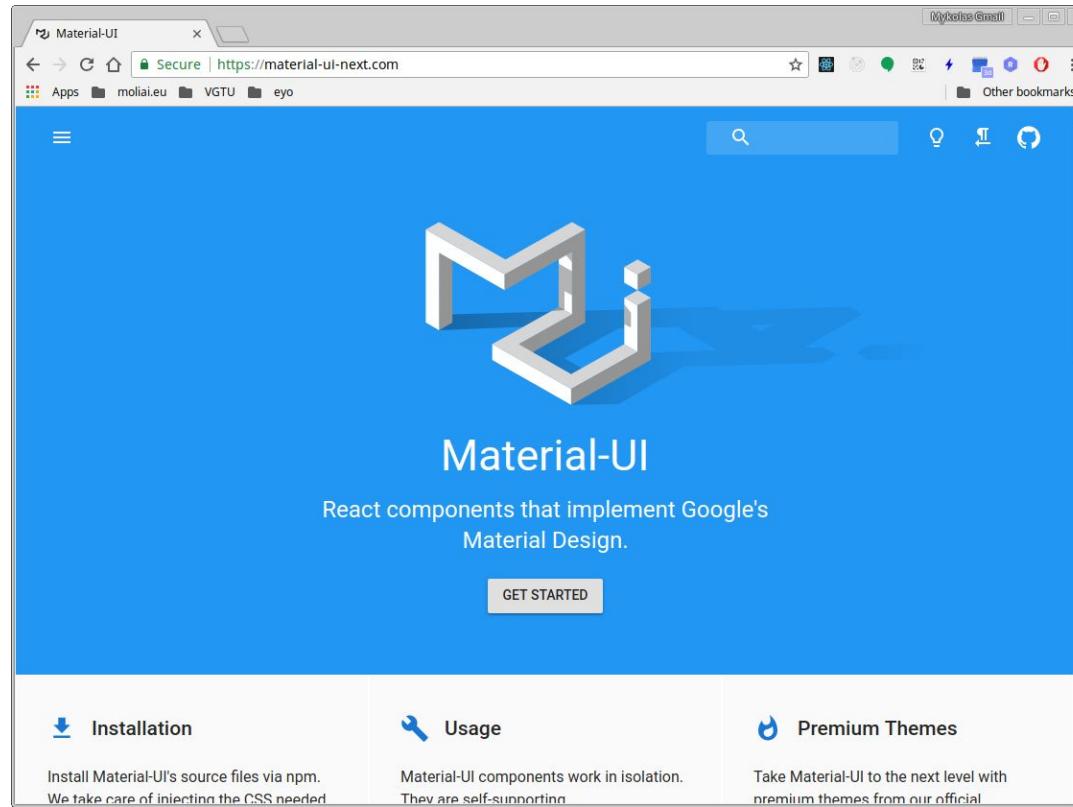
  /* Provides extra visual weight and identifies the primary action in a set of
   * buttons.
  <Button bsStyle="primary">Primary</Button>
```

Material Design



A screenshot of a web browser window displaying the 'Introduction' page of the Material Design guidelines. The browser's address bar shows the URL <https://material.io/guidelines/material-design/introduction.html>. The page itself has a teal header with the title 'Material Design – Introduction'. Below the header is a large, stylized graphic of overlapping rectangular shapes in various shades of teal and blue. At the bottom of the page, there is a block of text that reads: 'We challenged ourselves to create a visual language for our users that synthesizes the classic principles of good design with the innovation and possibility of technology and science. This is material design. This spec is a living document that will be...'.

Material UI



The screenshot shows a web browser window displaying the official Material-UI website at <https://material-ui-next.com>. The page has a blue background. At the top center is a large, metallic 3D-style 'M' logo. Below it, the text 'Material-UI' is displayed in a white sans-serif font. Underneath that, a subtitle reads 'React components that implement Google's Material Design.' A prominent 'GET STARTED' button is centered below the subtitle. At the bottom of the page, there are three sections: 'Installation' with a download icon, 'Usage' with a key icon, and 'Premium Themes' with a flame icon. Each section contains a brief description and a link.

Secure | https://material-ui-next.com

☰

M

Material-UI

React components that implement Google's Material Design.

GET STARTED

Installation

Usage

Premium Themes

Install Material-UI's source files via npm.
We take care of injecting the CSS needed

Material-UI components work in isolation.
They are self-supporting

Take Material-UI to the next level with
premium themes from our official



T10 - Konsultacija