

# HTML5, CSS, JavaScript

Kurso dėstytojas: Tautvydas Dulskis

# How computer software is made



step 1

open a blank document



step 2

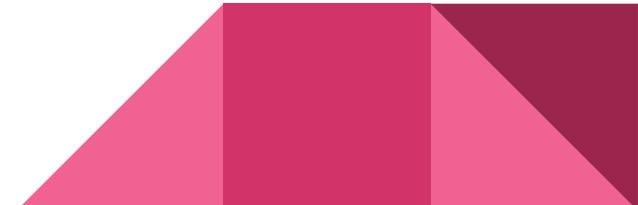
summon magical fairies



step 3

publish your software

# Kurso pabaiga



## Four Reasons to Turn On Your Webcam During Video Calls



# Tarpiniai atsiskaitymai

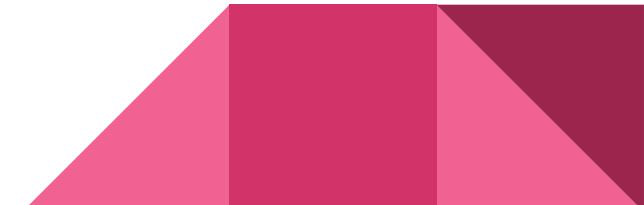
Kurso metu bus **2-3** tarpiniai atsiskaitymai.

Praleidus atsiskaitymą  
rašomas 0 balų vertinimas

# Svarbūs patarimai kursui

JS

HTML



# Pirmas kartas kai susiduri su programavimu ?



CSS

HTML

HTML

# We learn

- 📖 10 percent of what we read,
- 💡 20 percent of what we hear,
- 👀 30 percent of what we see,
- 👀💡 50 percent of what we see and hear,
- 🗣 70 percent of what we discuss,
- 🧑 80 percent of what we experience and
- 🧑🧑 95 percent of what we teach others.

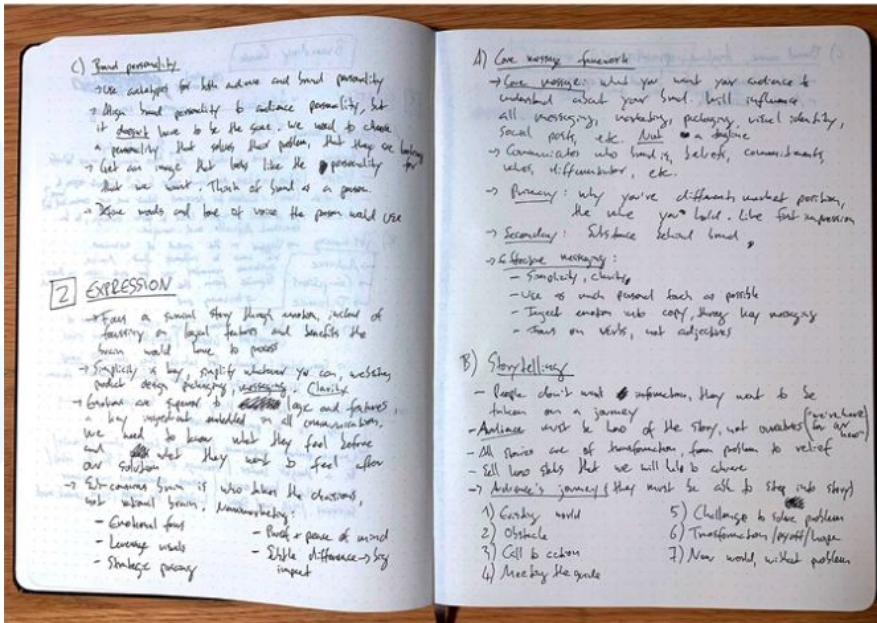
JS



HTML



# Užrašai...



# Programavimas tai...

JS

- 1% kodo rašymas
- 40% debuginimas (Problemos paieška)
- 15% kavos pertraukėlės
- 30% Klaidų Googlinimas
- 9% Spoksojimo su kolegom į ekraną
- 5% bandymas copy/paste Stack Overflow sprendimus

HTML

CSS

Tip nr.: 1

JS

# Niekada nepasiduok



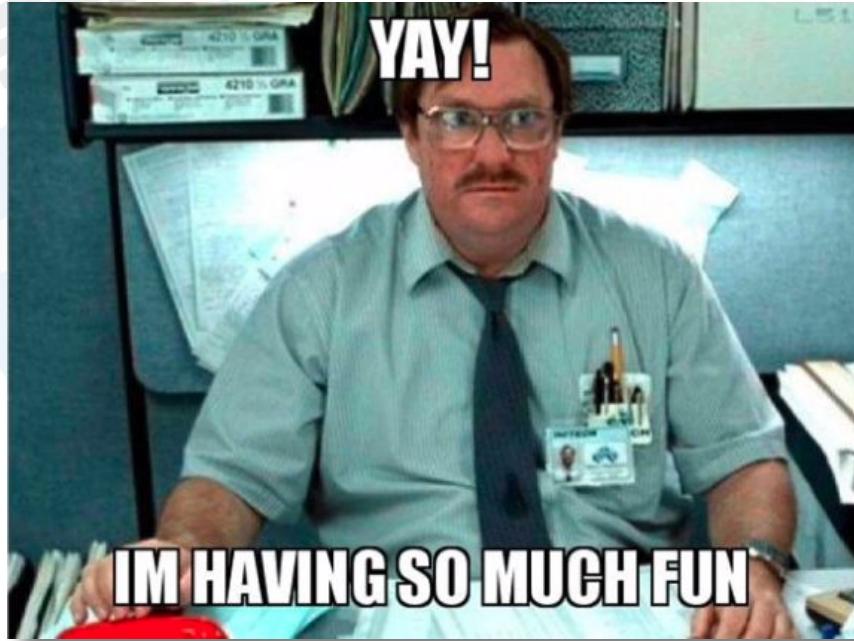
CSS

HTML

HTML

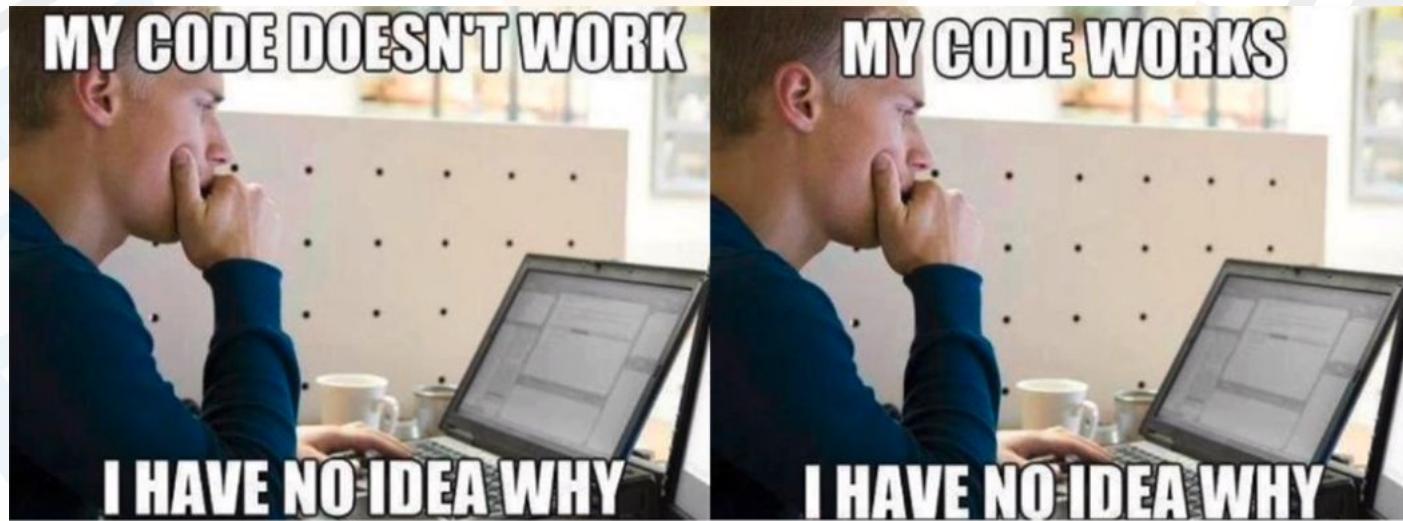


# Ir svarbiausia, tai turi būti smagu



Juk smagu matyti savo darbo rezultatus.  
Na o jei visgi pasijusi nusivylės, sustok ir  
grįžk vėliau, svarbiausia nenutrauk to ką  
jau pradėjai

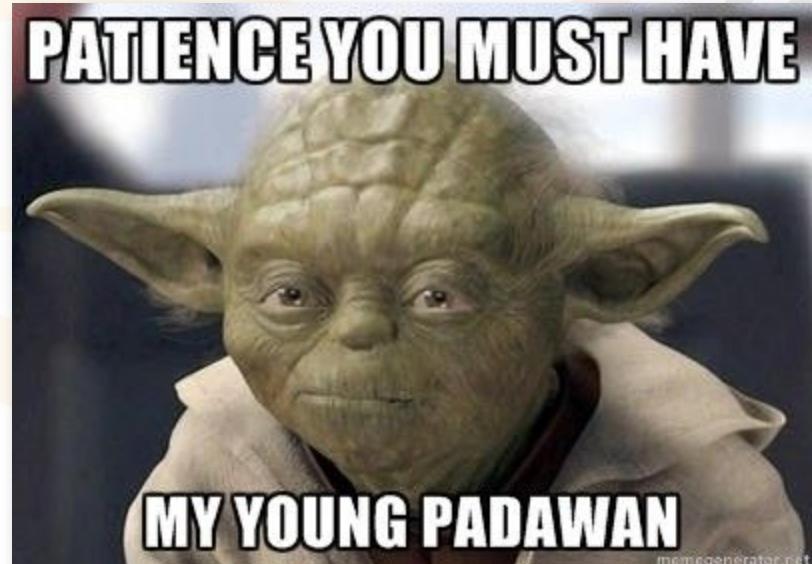
# Pradžioje nesivargink besiaiškindamas KODĖL



Nestresuok dėl efektyvaus, greito ar švaraus kodo. Kol mokomės norime kad jis tiesiog veiktu

# Neskubėk mokytis

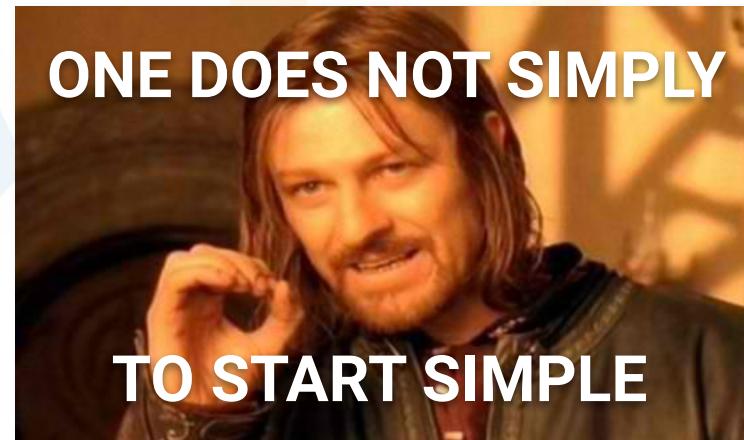
- Suprasti pagrindus yra svarbiau už greitį
- Kokybiskas mokymasis užima laiko
- Praktika ir projekto patirtis yra nepakeiciama
- Perkraunama informacija gali būti sunku įsisavinti
- Nepamiršk eksperimentuoti
- Pasitikėk savimi ir savo progresu



Tip nr.: 2

JS

## Pradėk nuo paprastų dalykų



Tip nr.: 3



# Kaip suteikti informaciją programuotojui ?

<https://fb.watch/aC9VjgS-XG/>

Jei klausiate PHP programuotojo, reikia nurodyti kokias bibliotekos ar programinę įrangą naudojate, kokius duomenis turite ir kokius rezultatus tikitės gauti.

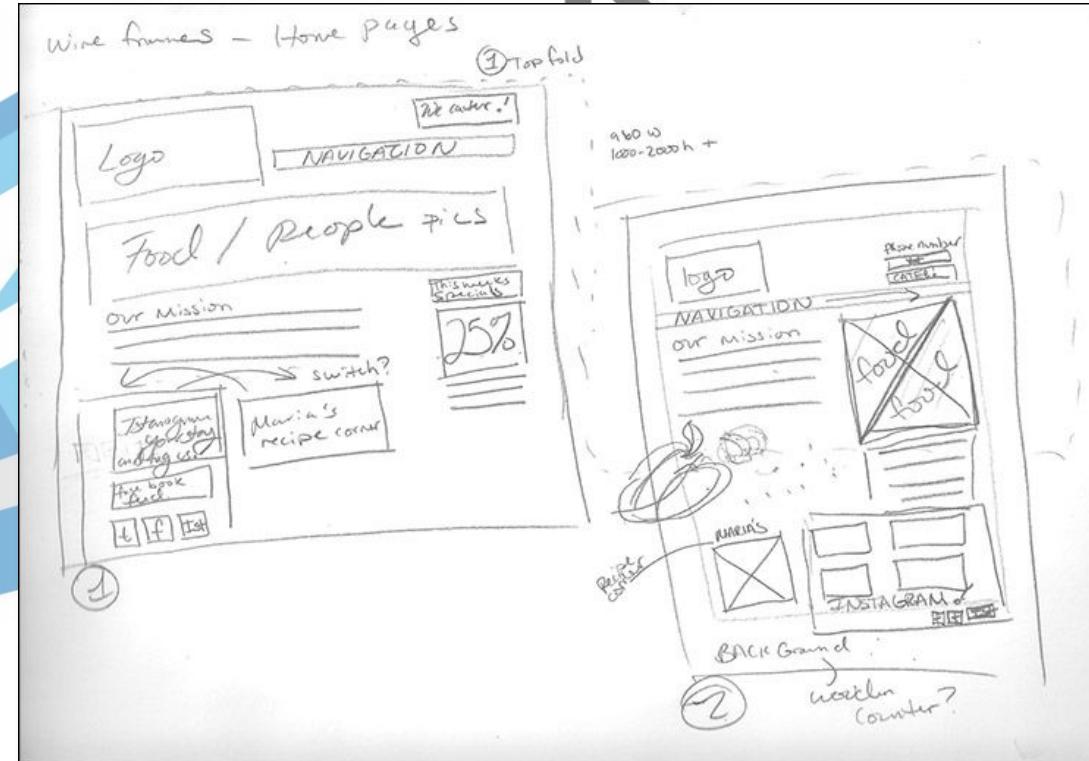
## Pavyzdžiui:

*"Bandau su PHP programavimo kalbą ir "MySQL" duomenų baze sukurti užklausą, kuri ištrauktu visus vartotojus, turinčius "premium" planą, bet mano užklausoje gaunu klaidą "Unknown column 'plan' in 'where clause'. Galėtumėte man padėti išspręsti šią problemą ir patikslinti mano užklausą?"*

Taip pat reikia pateikti kodą, kurio jūs naudojate, ir kokius klaidos pranešimus gavote. Tai padės PHP programuotojui greičiau ir tiksliau suprasti jūsų problemą ir suteikti jums reikiama pagalbą.

Tip nr.: 5

# Prototipavimas



Tip nr.: 6

Pradėkite dabar

JS

# 9 Dalykai

## kuriuos verta žinoti pradedant programuoti

CSS

HTML



JS

# TUTORIALAI

Nepakliūk į spastus

Kas perdaug tas nesveika

HTML



JS

# Nepraleisk pagrindų

Dažnai norime matyti rezultataj čia ir dabar.

Bet tai tik apsunkina mokymosi procesą.

# Nesistenk visko įsiminti

Taip tik apkrausi savo galvą dažnai besikeičiančiais dalykais.



# Susirask mentorij

Tai žmogus kuris tave nukreips tinkama linkme.  
*(Tai ne mokytojas)*

# Skaityk dokumentacijas

Nepasikliauk tutorialais, dažniausiai jie negali atsakyti į tau rūpimus klausimus.

# Programavimas turi patikti

Nemégiamas darbas tik našta



JS

# Su laiku taps lengviau

Sukurės produktą/aplikaciją/funkcija po kelių metų tau gali prieikti jo ir vėl,  
tačiau tai nereiškia, kad tu tai kopijuosis. Greičiausiai tu tai modifikuos ir prisitaikysi su jau naujom žiniom



JS

# Kantrybė

Tai **ne maratonas ir ne sprintas.**

**Nelygink savęs** su kuo nors **kitu.**

Gal kas kitas pasirinko jam tinkamą kelią, bet tas kelias tau netinka.

# Socialiniai tinklai

Naujos pažintys ir galimybė greičiau rasti tai ko ieškai

# Truputis realybės

JS

CSS

HTML

# Iš realaus gyvenimo

JS



Bug after Bug



Overrating



**Developer: Makes a simple, intuitive UI**



CSS

HTML

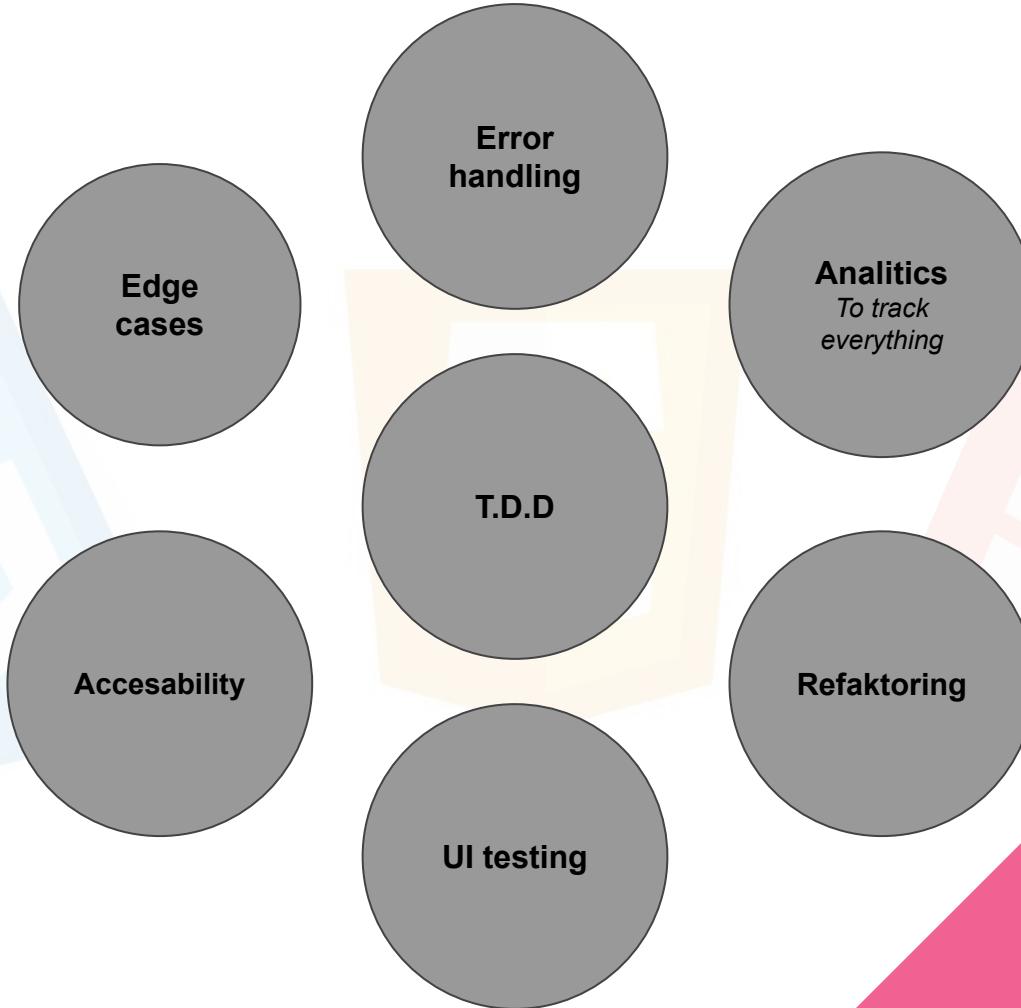
HTML



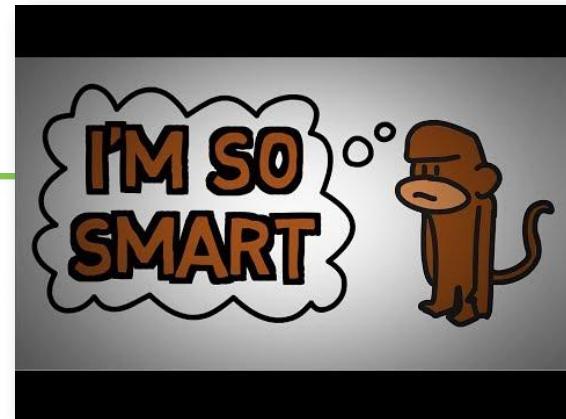
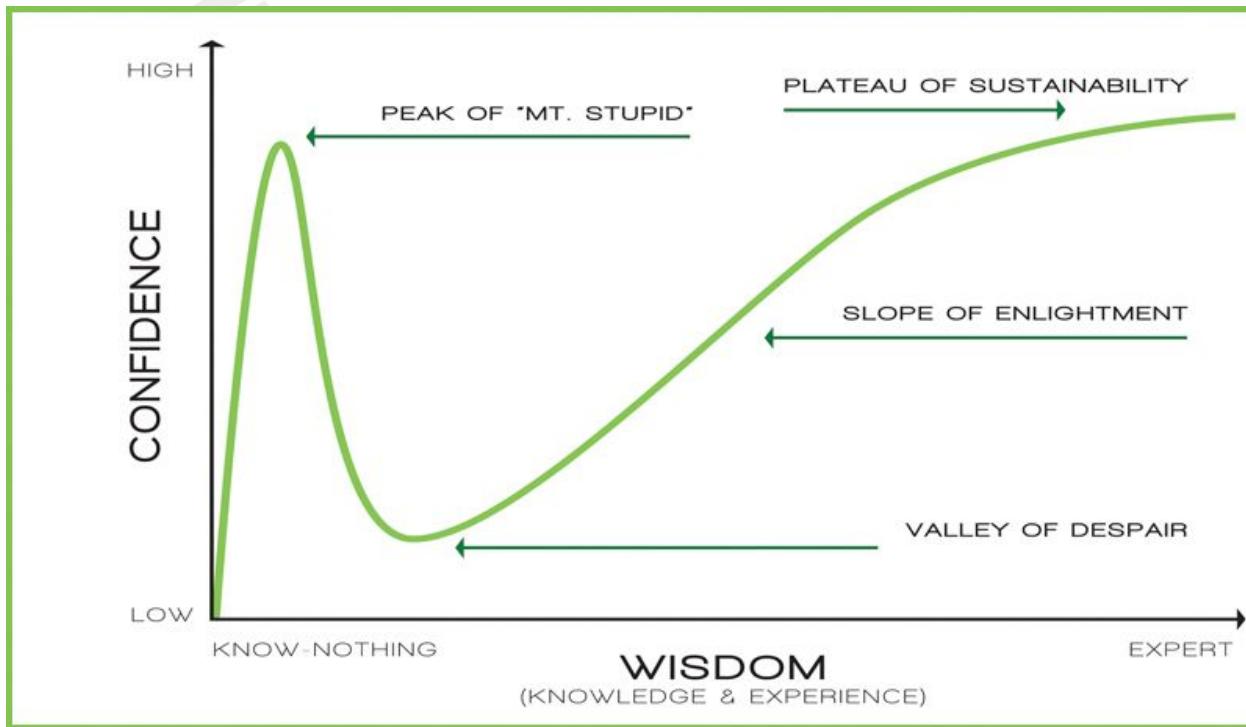
# 90 / 90 taisyklė

**Manai kad padarei 90% darbo? Liko dar 90% darbo :)**

Kiti 90% susideda iš ...



# Daningo Kriugeroio efektas



<https://www.youtube.com/watch?v=GJz66wm95-M>

# Kaip paspartinti pasitikėjimą ???



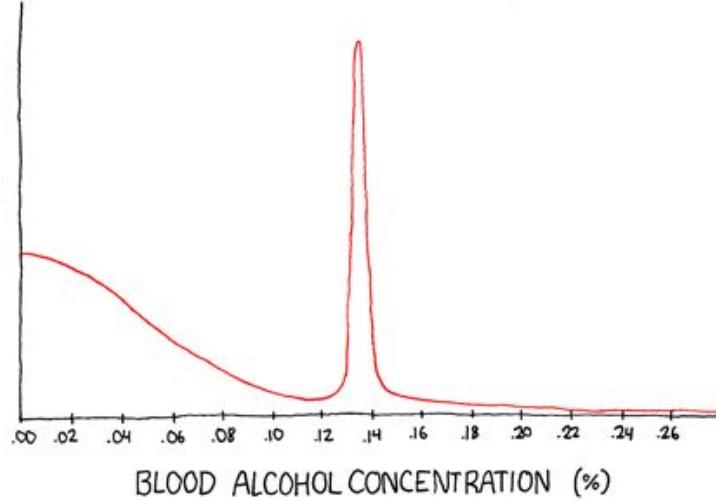
CSS

JS

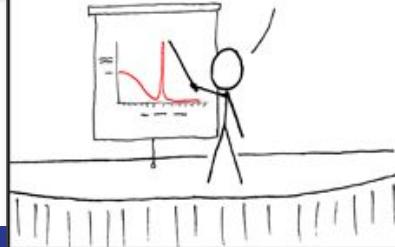
HTML

CSS  
HTML

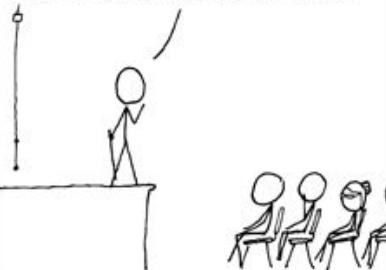
PROGRAMMING  
SKILL



CALLED THE BALLMER PEAK, IT WAS DISCOVERED BY MICROSOFT IN THE LATE 80's. THE CAUSE IS UNKNOWN, BUT SOMEHOW A B.A.C. BETWEEN 0.129% AND 0.138% CONFERS SUPERHUMAN PROGRAMMING ABILITY.

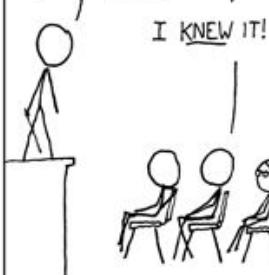


HOWEVER, IT'S A DELICATE EFFECT REQUIRING CAREFUL CALIBRATION - YOU CAN'T JUST GIVE A TEAM OF CODERS A YEAR'S SUPPLY OF WHISKEY AND TELL THEM TO GET CRACKING,



...HAS THAT EVER HAPPENED?

REMEMBER  
WINDOWS ME?  
I KNEW IT!



## Patarimai baigiamojo temai:

- ★ Automatizuokite kasdieninę rutiną
- ★ [Sukurkite abstraktaus meno generatoriu](#)
- ★ [Sukonstruokite kraštovaizdj](#)
- ★ [Virtualų augintinio simuliatoriu](#)
- ★ [Labirinto generatoriu](#)
- ★ [Tekstinj RPG žaidima](#)
- ★ [Pirkinių sąrašo aplikacija](#)
- ★ Muzikos svetainę
- ★ Knygų dalinimosi svetainę
- ★ Peržiūrėtų filmų/serialų aplikaciją
- ★ [Online piešimo svetaine](#)

...

# Pirma paskaita

Techinai pradmenys

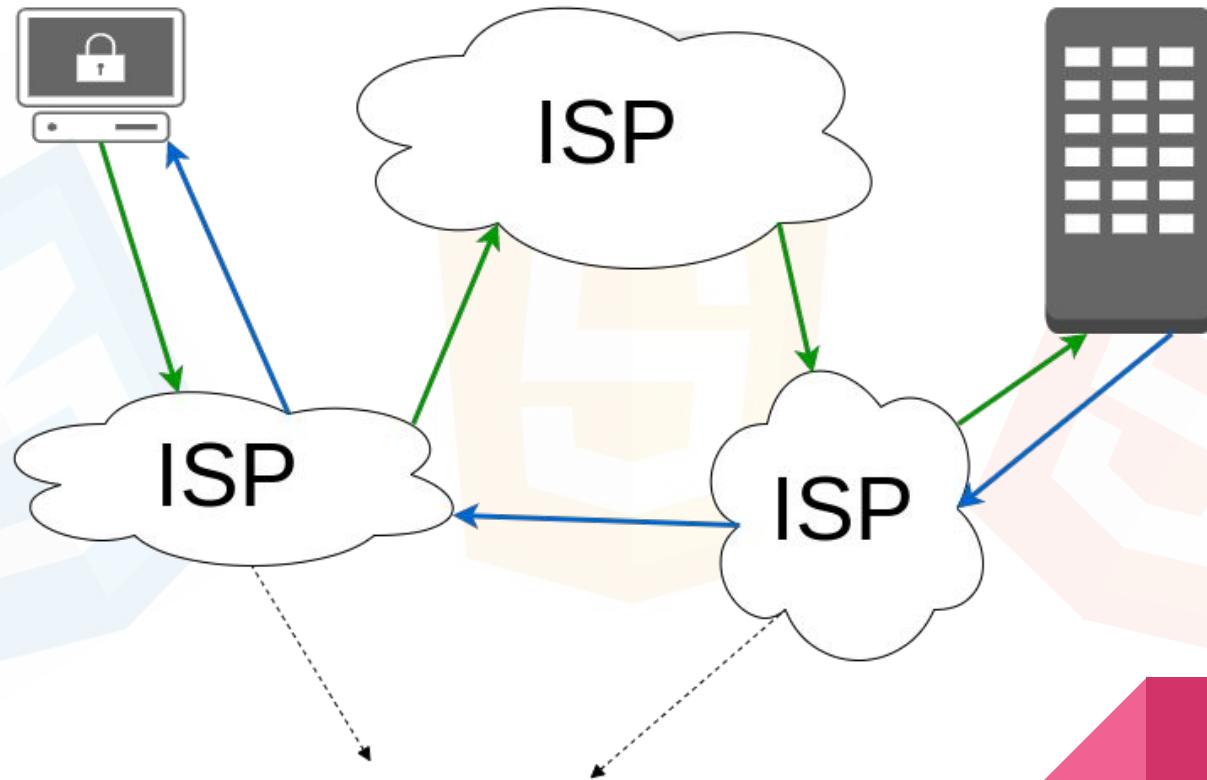


# BIG QUESTION



## HOW THE INTERNET WORKS



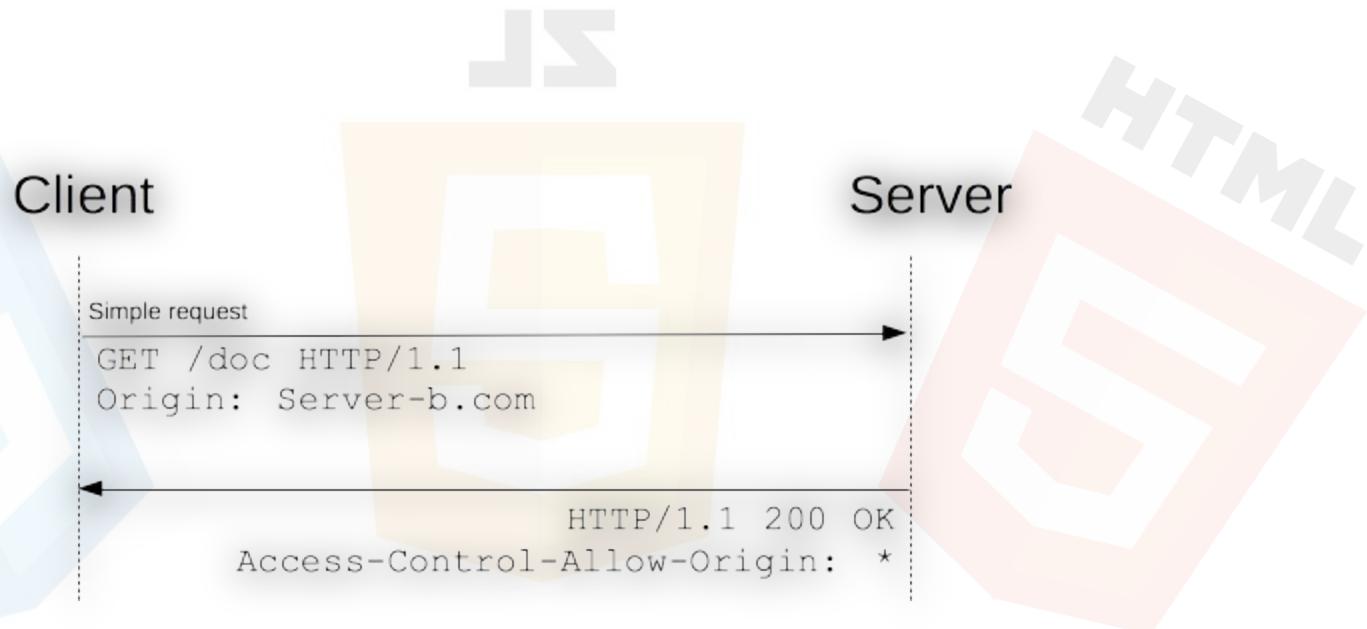


CSS

HTML

HTML

# Užklausa

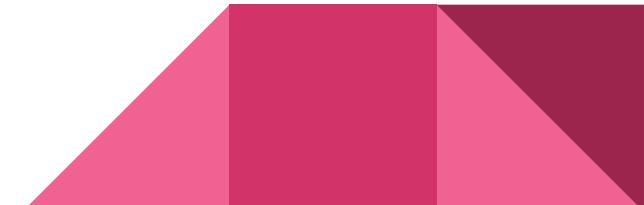


# Kurso metu naudojama medžiaga

JS



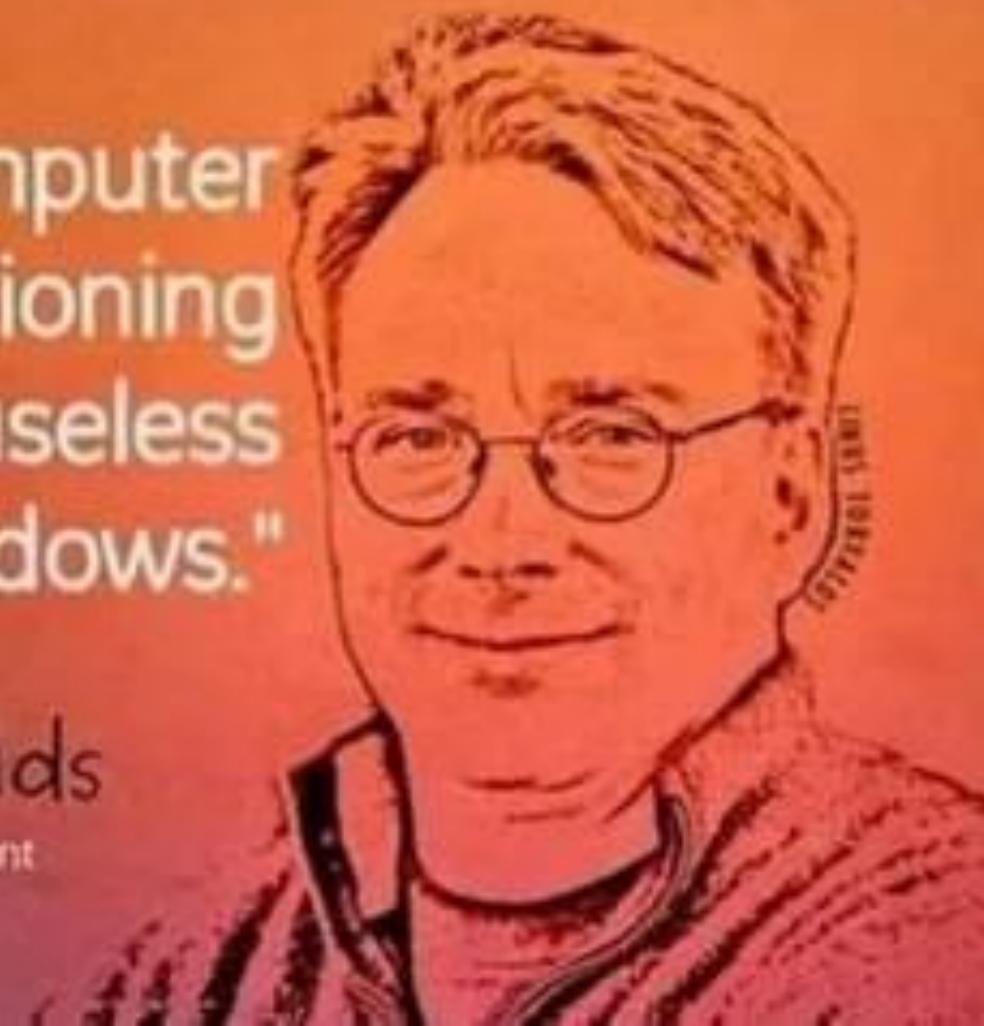
<https://kaunas-coding-school.github.io/webKursas/>



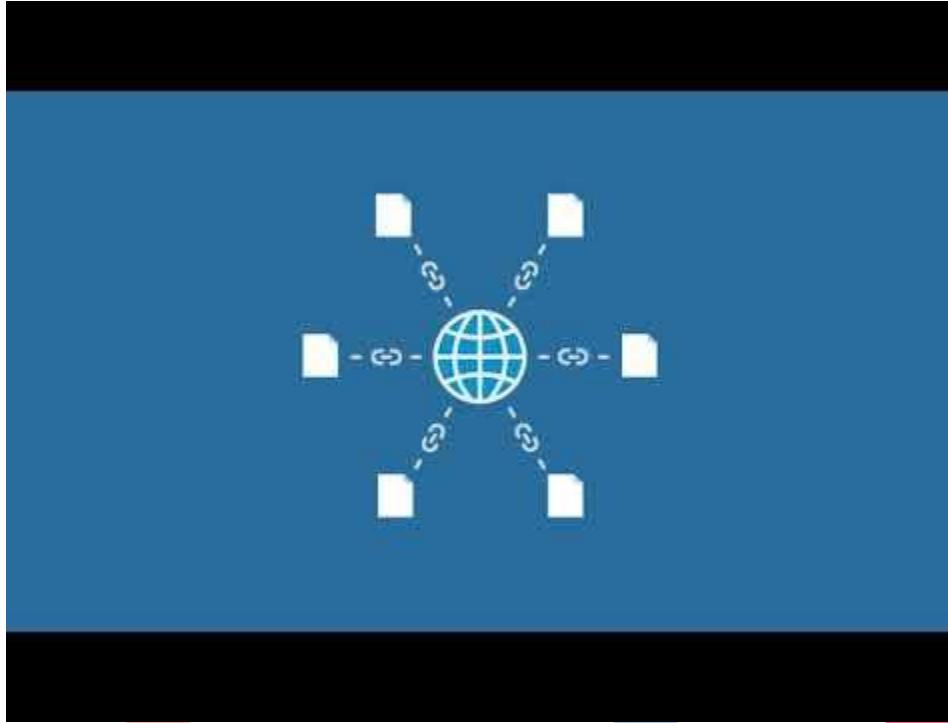
"A computer  
is like air conditioning  
- it becomes useless  
when you open Windows."

Linus Torvalds

December 28, 1969 - Present



# Tinklalapio struktūra



# Mobili aplikacija mokytis programavimo

[Mimo: Learn coding, programming - Apps on Google Play](#)

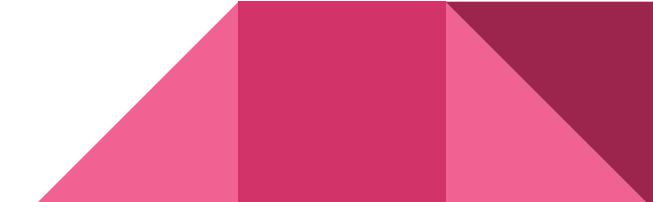
# STUDIJUOJAMŲ DALYKŲ SĄRAŠAS

Web svetainės kūrimas, administravimas ir talpinimas serveryje

**HTML5** (tinklapio struktūra, formos elementai, grafiniai elementai, kt.)

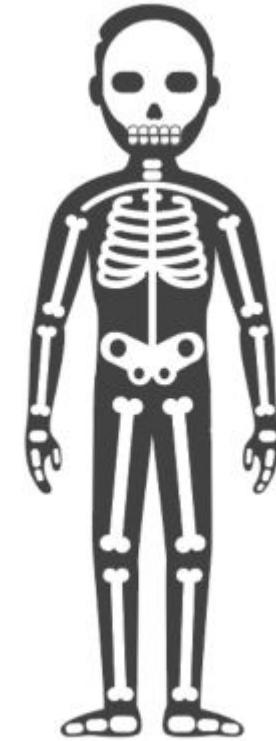
**CSS3** (spalvos, rėmeliai, animacija, transformacijos. kt.)

**JavaScript** pavyzdžiai, pritaikymas



# HTML - Hyper Text Markup Language

- Dokumento struktūra, antraštė, svarbiausi elementai
- Atributai
- Teksto formatavimas
- Grafiniai vaizdai, spalvų kodai, vardai
- Nuorodos / Hipertekstas
- Sąrašai, apibrėžimai, lentelės



# Frontend



# CSS3 (spalvos, rėmeliai, animacija, ...)

- Kas tai?
- Sintaksė
- Spalvos
- Matmenys ir Rėmeliai
- Formatavimas
- Elementų vaizdavimas
- Elementų pozicijos
- Pseudo klasės
- Animacija
- .....

Lėkščių žaidimas: <https://flukeout.github.io/>

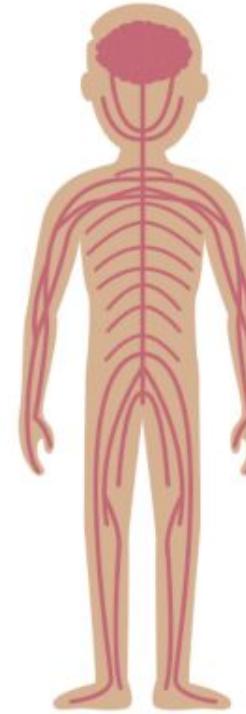


# Backend



# JavaScript pavyzdžiai, pritaikymas

- Kas tai?
- Sintaksė panaši į daugelį kalbų
- Kam skirtas JS?
- JS bibliotekos.
- Kaip jas naudoti?
- JS populiariausios karkasinės sistemos  
(Frameworks)
- Kaip juos naudoti?
- Limitation is only your imagination





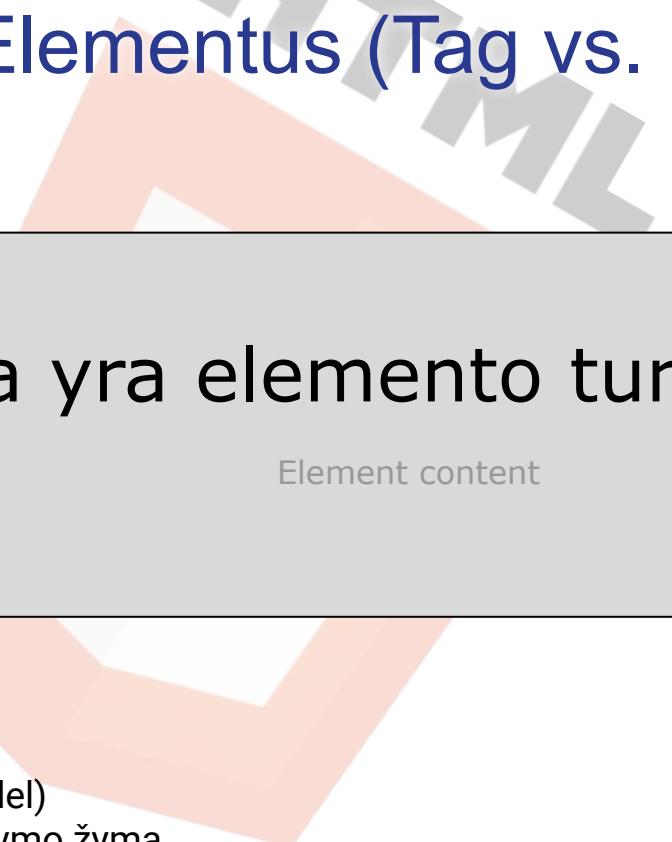
Ar viskas aišku :)



**Tuomet  
Pradedam  
programuoti**

JS

# Žymos prieš Elementus (Tag vs. Element)



<žyma>Čia yra elemento turinys</žyma>

| Opening tag name |

Element content

| Closing tag name |

\* Tag,  
Tag Name,  
DOM (Document Object Model)

\*\* Ne visi elementai turi uždarymo žymą

# Kodo struktūra

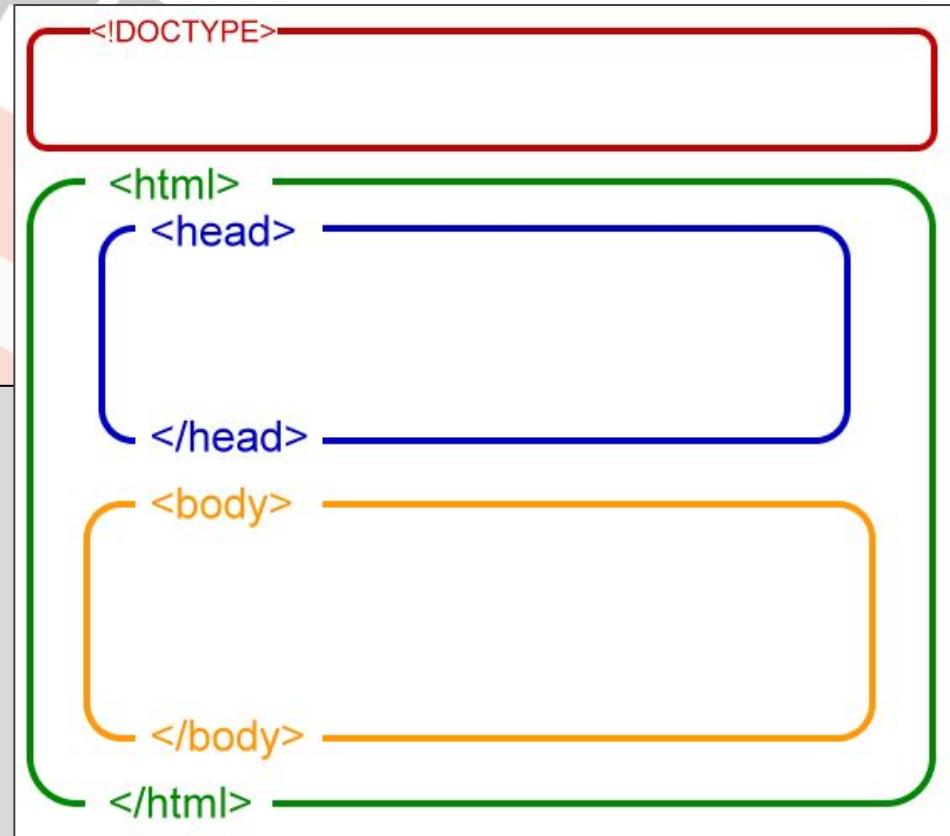
Kaip atrodo kodo struktūra:

Tai **index.html** failas

```
<!DOCTYPE html>
<html>
  <head>
    <title>Puslapio pavadinimas</title>
  </head>
  <body>

    <h1>Tai yra antraštė</h1>
    <p>Tai yra paragrafas.</p>
    <div>O čia yra turinio skyrius.</div>

  </body>
</html>
```



# HTML komentarai

```
<!-- This is a comment -->

<p>This is a paragraph.</p>

<!-- Remember to add more information here -->
```

```
<!--[if IE lt 10]>
    .... some HTML here ....
<![endif]-->
```

# HTML Formatavimo elementai

`<b> Bold text</b>`

`<strong>Important text</strong>`

`<i>Italic text</i>`

`<em> Emphasized text</em>`

`<mark> - Marked text</mark>`

`<small>` - Small text`</small>`

`<del>` - Deleted text`</del>`

`<ins>` - Inserted text`</ins>`

`<sup>` - Superscript text`</sup>`

`<sub>` - Subscript text`</sub>`

`</br>` - Naujos eilutės

identifikatorius

# Semantiniai elementai

- 
- `<div>` Aprašo sritį dokumente arba kitoje srityje
  - `<span>` Nurodoma nedidelė dalis informacijos dokumente arba srityje
  - `<header>` Nustato antraštės reikšmę dokumente arba sekcijoje
  - `<footer>` Apibrėžia poraštę dokumente arba sekcijoje
  - `<main>` Nurodo pagrindinį turinį dokumente
  - `<section>` Aprašo sekcija dokumente
  - `<article>` Aprašo straipsnio sritį
  - `<aside>` Apibrėžia atskirą turinį, bet ne pagrindinį puslapio turinį

# HTML5 Elementai

Įdomiausi HTML5 elementai:

**Semantiniai elementai:** `<header>`, `<footer>`, `<article>`, ir `<section>`

**Formų atributai:** `number`, `date`, `time`, `calendar`, ir `range`.

**Grafiniai elementai:** `<svg>` ir `<canvas>`

**Multimedia elementai:** `<audio>` ir `<video>`

# HTML lentelės

```
<table>
  <tr>
    <th>Firstname</th>
    <th>Lastname</th>
    <th>Age</th>
  </tr>
  <tr>
    <td>Jill</td>
    <td>Smith</td>
    <td>50</td>
  </tr>
  <tr>
    <td>Eve</td>
    <td>Jackson</td>
    <td>94</td>
  </tr>
</table>
```

# HTML Elementai

HTML

ARBA

ML/Element

Elementų tipai

<https://developer.mozilla.org/en-US/docs/Web/HT>

<https://www.w3schools.com/tags/>

# &copy; All rights reserved

© Laikui bėgant, aš prijau išvadą, kad, kaip ir *teisininkai, mirtis ir mokesčiai*, programinės įrangos licencijos pasirinkimas yra **neišvengiamas**. Žinoma, nesvarbu, ar tavo akys yra vienintelės akys, kurios kada nors matys šį kodą. Bet tinkama programinės įrangos licencija yra būtinas **blogis** bet kokiam kodui, kurį planuojate išleisti į visuomenę.

© <https://blog.codinghorror.com/pick-a-license-any-license/>



# GIT

(Nemalonus Žmogus)

[V]ersion [C]ontrol [S]ystem for tracking changes in files

## Local

## Remote

working  
directory

staging  
area

localrepo

remote  
repo

git add

git commit

git push

git pull

git checkout

git merge

# Keyboard Shortcuts

Spartieji klavišai dažnai naudojami šiuolaikinėse operacinėse sistemoje ir kompiuterių programinės įrangos programose.

[https://www.w3schools.com/tags/ref\\_keyboardshortcuts.asp](https://www.w3schools.com/tags/ref_keyboardshortcuts.asp)

# Git konfiguracija

```
$ git config --global user.name "Vardenis Pavardenis"  
$ git config --global user.email "v.pavardenis@pastas.lt"
```

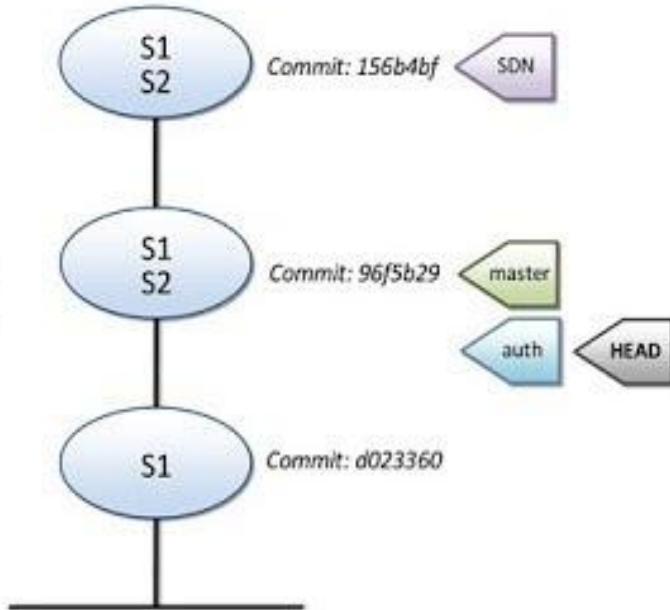
# .gitignore

```
$ touch .gitignore
```

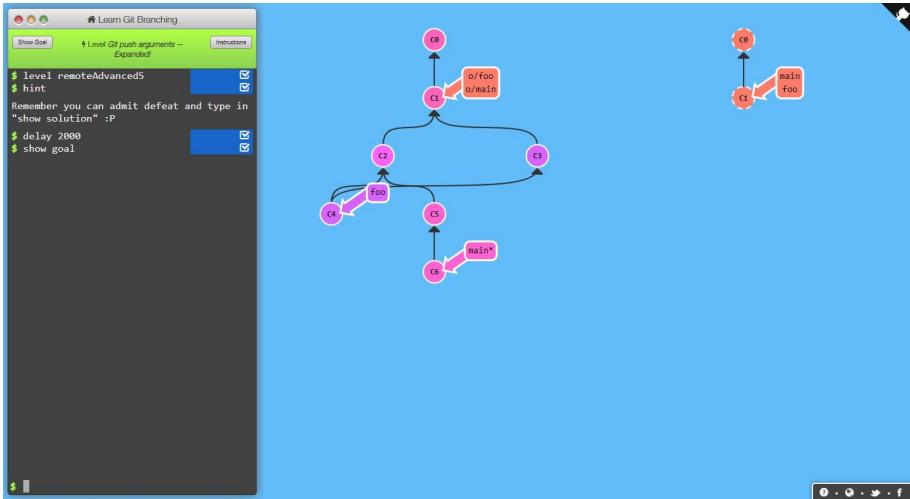
# Projekto inicijavimas

```
$ git init  
  
$ git add .  
  
$ git commit -m "Mano pirmasis komitas"  
  
$ git status
```

# Git: Branch Merge



# Learn Git Branching





## IN CASE OF FIRE

- ☛ GIT COMMIT
- ☛ GIT PUSH
- ☛ LEAVE BUILDING

## Nuorodos, paveikslėliai, atributai, Formos, HTML5 tagai



# Atributai

Visi HTML elementai gali turėti atributus.

Atributai suteikia papildomos informacijos apie elementą.

Atributai visada nurodomi atidaromoje žymoje (tag).

Atributai įprastai aprašomi taip:

```
<p title="Esu paragrafo užuomina">  
    Tai paragrafas  
</p>
```

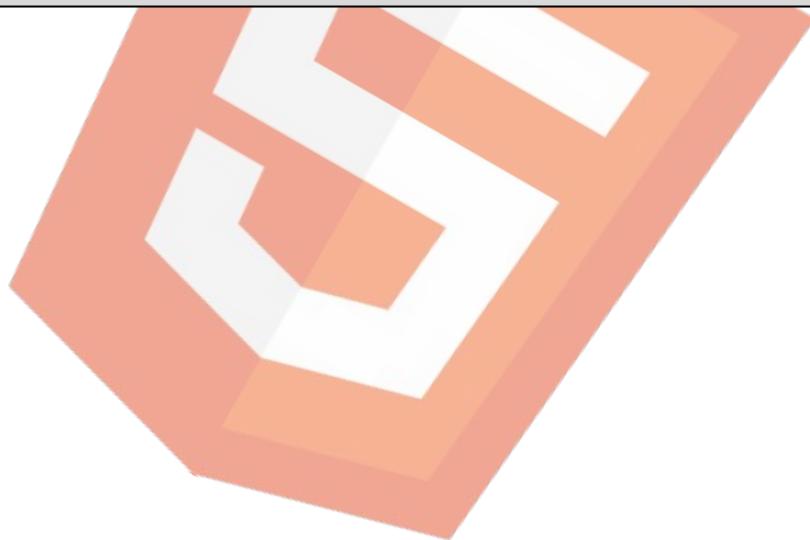
# Atributai

```
<div title="Esu div bloko užuomina" class="aplankas">
    <ul class="saransas">
        <li class="elementas raudonas">Pirmasis saraso elementas</li>
        <li class="elementas melynas">Antrasis saraso elementas</li>
        <li class="elementas raudonas">Treciasis saraso elementas</li>
        <li class="elementas raudonas" id="unikalus_elementas" >Ketvirtasis saraso elementas</li>
        <li class="elementas raudonas">Penktasis saraso elementas</li>
    </ul>
</div>
```

# Nuoroda

HTM

```
<a href="http://google.lt">Nuoroda | Google</a>
```



# Paveikslėlis

```

```



 - logo.jpg yra tame pačiame kataloge kur ir esamas dokumentas

 - logo.jpg yra img kataloge atskaitoje nuo esamo dokumento

 - logo.jpg yra img kataloge atskaitoje nuo pagrindinio puslapio

 - logo.jpg yra kataloge vienu katalogo lygiu aukščiau nuo esamo dokumento

# Paveikslėlis

```
<picture>
  <source media="(min-width: 465px)" srcset="img/r2/img_white_flower.jpg">
  
  <source media="(min-width: 650px)" srcset="img/r1/img_pink_flowers.jpg">
</picture>
```

# HTML Formų elementai

```
<form action="http://localhost/kontaktai" method="POST" autocomplete="on" novalidate>
    ...
    form elements
    ...
</form>
```

**action** - Veiksmo atributas apibrėžia veiksmą, kuris turi būti atliekamas, kai pateikiama (submitinama) forma.

**method** - Metodo atributas nurodo HTTP metodą (**POST | GET | PUT | PATCH**), kuris turi būti naudojamas pateikiant formos duomenis

\* - Daugiau atributų čia <https://prnt.sc/vlio4j>

# HTML Formų elementai

```
<label for="fn">Vardas</label>
<input name="firstname" type="text" id="fn" />

<label for="masina">Automobilis</label>
<select name="cars" id="masina">
  <option value="auto1">Volvo</option>
  <option value="auto2">Saab</option>
  <option value="kitas3">Fiat</option>
  <option value="a4">Audi</option>
</select>

<label for="zinute">Tavo Zinute man</label>
<textarea name="message" rows="10" cols="30" id="zinute">
The cat was playing in the garden.
</textarea>
<button type="button" onclick="alert('Hello World!')">Labas!</button>
<input value="Siūsti" type="submit" />
```

Galimos **input** elemento **type** atributo reikšmės: **button**, **checkbox**, **color**, **date**, **datetime-local**, **email**, **file**, **hidden**, **image**, **month**, **number**, **password**, **radio**, **range**, **reset**, **search**, **submit**, **tel**, **text**, **time**, **url**, **week**

# HTML Formų elementai

```
1 <input>
2
3 <label>
4
5 <select>
6
7 <option>
8
9 <textarea>
10
11 <button>
12
13 <fieldset>
14
15 <legend>
16
17 <datalist>
18
19 <output>
20
21 <optgroup>
```

**<input>**: Tai elementas, leidžiantis vartotojui įvesti duomenis į formą, pvz., tekstą, skaičius ar datas.

**<label>**: Tai elementas, skirtas nurodyti formos elemento pavadinimą ar aprašymą, siekiant padidinti naudojimo patogumą.

**<select>**: Tai elementas, leidžiantis vartotojui pasirinkti vieną iš kelių paruoštų variantų iš išskleidžiamamo sąrašo.

**<option>**: Tai elementas, naudojamas kartu su **<select>**, norint nurodyti galimus pasirinkimo variantus išskleidžiamajame sąraše.

**<textarea>**: Tai elementas, leidžiantis vartotojui įvesti ilgesnį teksto kiekį, dažnai naudojamas daugiaelėms tekstu atsakymams.

**<button>**: Tai elementas, skirtas sukurti mygtuką, kuris gali būti naudojamas veiksmams, pvz., formos siuntimui, atlikti.

**<fieldset>**: Tai elementas, naudojamas grupuoti susijusius formos elementus, kad padėtų vartotojams suprasti formos struktūrą.

**<legend>**: Tai elementas, naudojamas kartu su **<fieldset>** ir nurodantis formos elementų grupės pavadinimą ar antraštę.

**<datalist>**: Tai elementas, suteikia vartotojui galimybę pasirinkti iš pateiktų variantų ar įvesti savo reikšmę, naudojant **<input>** elementą.

**<output>**: Tai elementas, skirtas rodyti apskaičiuotą rezultatą, gaunamą iš formos duomenų ar kodo.

**<optgroup>**: Tai elementas, naudojamas kartu su **<select>**, grupuoja pasirinkimo variantus išskleidžiamajame sąraše pagal kategorijas.

# Formų apribojimai (Constraints)

HTML formos elementai turi įvairius atributus, kurie nustato jų elgseną ir apribojimus. Čia yra keletas pagrindinių atributų, taikomų dažniems formos elementams:

1. **type**: Nurodo formos elemento tipą, pvz., `text`, `email`, `password`, `number`, `checkbox`, `radio` ir kt.
2. **name**: Suteikia formos elementui pavadinimą, kuris naudojamas duomenų perdavimui į serverį.
3. **value**: Nurodo formos elemento pradinę reikšmę arba, priklausomai nuo elemento tipo, gaunamą reikšmę.
4. **placeholder**: Nurodo trumpą tekštą, kuris rodomas elemente, kol jis nėra užpildytas.
5. **required**: Šis atributas nurodo, kad formos elementas yra privalomas ir turi būti užpildytas prieš siunčiant formą.
6. **disabled**: Šis atributas padaro formos elementą neaktyvų, t. y. vartotojas negali jo keisti arba pasirinkti.
7. **readonly**: Nurodo, kad formos elementas yra tik skaitymui, ir vartotojas negali jo keisti.
8. **maxlength** (tik `<input>` ir `<textarea>` elementams): Nurodo didžiausią leistiną simbolių skaičių, kurį galima įvesti.
9. **min** ir **max** (tik `<input>` elementui su `type="number"` arba `type="range"`): Nurodo leistiną mažiausią ir didžiausią reikšmes.
10. **pattern** (tik `<input>` elementui su `type="text"` ar pan.): Nurodo reguliarųjį išraišką, kuriai turi atitikti įvesties tekstas.
11. **autocomplete**: Nurodo, ar naršykė turėtų siūlyti ankstesnes įvestis, kai vartotojas pradeda rašyti į formos elementą.
12. **multiple** (tik `<input type="file">` ir `<select>` elementams): Leidžia vartotojui pasirinkti kelis failus arba kelias `<option>` reikšmes.
13. **accept** (tik `<input type="file">` elementui): Nurodo leidžiamus failų tipus, kuriuos vartotojas gali pasirinkti.
14. **step** (tik `<input type="number">` ir `<input type="range">` elementams): Nurodo didžiausią leistiną skaičių, kurį galima padidinti arba sumažinti.

# Formų apribojimai (Constraints)

15. **size** (tik `<input>` ir `<select>` elementams): Nustato formos elemento plotį simboliais (teksto laukams) arba pasirinkimo variantų skaičiu (išskleidžiamiesiems sąrašams).
16. **autofocus**: Nurodo, kad formos elementas turėtų automatiškai gauti fokusą, kai puslapis yra užkraunamas.
17. **checked** (tik `<input type="checkbox">` ir `<input type="radio">` elementams): Nustato, ar elementas turėtų būti pažymėtas pagal nutylėjimą.
18. **selected** (tik `<option>` elementui): Nustato, ar pasirinkimo variantas turėtų būti pasirinktas pagal nutylėjimą.
19. **for** (tik `<label>` elementui): Susieja `<label>` elementą su formos elementu, nurodydama formos elemento `id` atributą.
20. **form** (kai kuriais atvejais): Nurodo formos, kuriai priklauso elementas, `id` atributą, leidžia susieti elementą su forma net jei jis yra ne formos viduje.
21. **list** (tik `<input>` elementui): Susieja `<input>` elementą su `<datalist>` elementu, nurodydama `<datalist>` elemento `id` atributą.
22. **src** (tik `<input type="image">` elementui): Nurodo nuotraukos failo URL, kuris bus naudojamas kaip mygtuko paveikslėlis.
23. **alt** (tik `<input type="image">` elementui): Nurodo alternatyvų tekštą, kuris bus rodomas, jei paveikslėlis nebus užkrautas.
24. **formaction** (tik `<button>` ir `<input type="submit">` elementams): Nurodo URL, kuriuo bus siunčiami formos duomenys, kai paspaudžiamas mygtukas.
25. **formmethod** (tik `<button>` ir `<input type="submit">` elementams): Nurodo formos siuntimo metodą (`GET` ar `POST`), kai paspaudžiamas mygtukas.

Tai dauguma atributų, naudojamų HTML formos elementams. Vis dėlto, reikėtų prisiminti, kad ne visi atributai gali būti taikomi visiems elementams, kai kurie yra specifiniai tam tikriems elementų tipams.

# HTML5 Uždavinys

## HTML form task1

### Building a form

New widgets:

- <abbr> - abbreviation
- <select>
  - <option>
- <input type="checkbox">

### Payment form

Required fields are followed by \*.

#### Contact information

Name: \*

E-mail: \*

Password: \*

#### Payment information

Card type:  Visa

Card number: \*

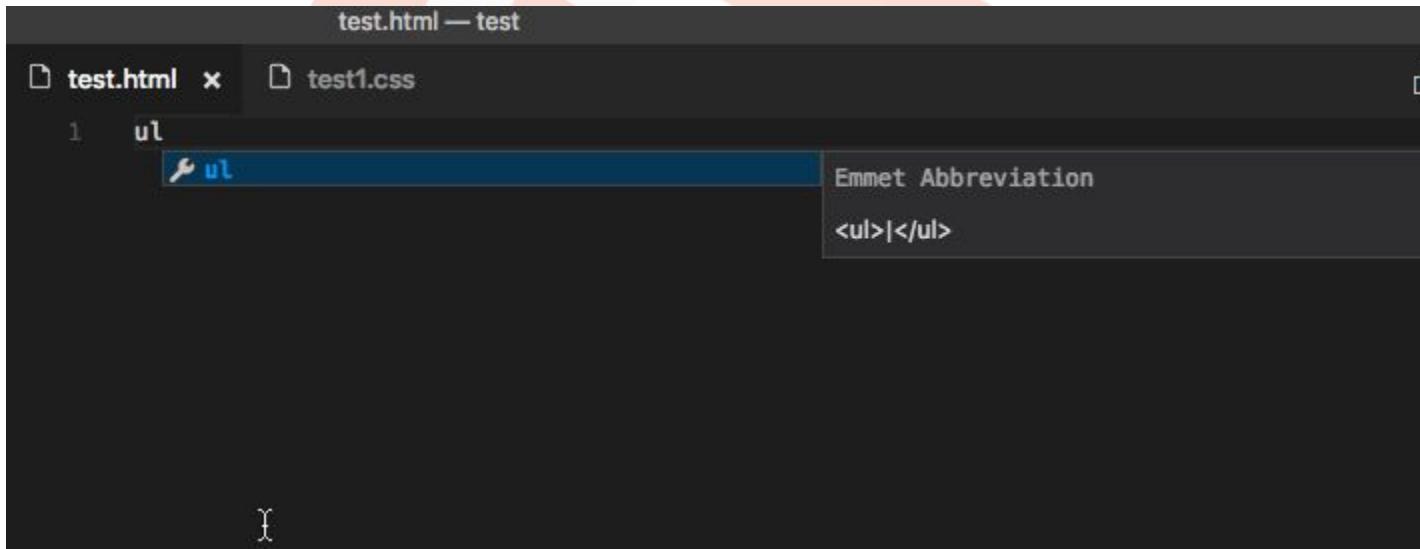
Expiration date: \*  dd/mm/yyyy, --:--

Agree with terms and conditions

Validate the payment

# Greitas kodo rašymas

<https://emmet.io>



A screenshot of a code editor window titled "test.html — test". The file contains the following code:

```
1  ul
```

The cursor is positioned after the "ul" tag. A tooltip labeled "Emmet Abbreviation" is displayed, showing the expansion of the abbreviation: "<ul>|</ul>".

**CSS**



# CSS?

CSS yra kalba, kuri apibūdina HTML dokumento stilių.

CSS aprašoma, taip kaip turi būti rodomi HTML elementai.

```
body {  
    background-color: lightblue;  
}  
  
h1 {  
    color: white;  
    text-align: center;  
}  
  
p {  
    font-family: verdana;  
    font-size: 20px;  
}
```

- Geresnis būdas CSS klasiu pavadinimams kurti
- Kaip pasirinkti ką stilizuoti

# Sintakse

CSS taisyklė/rinkinys susideda iš selektorių ir deklaravimo blokų:



Panaudojimas:

```
p.demesio {  
    text-align: center;  
    color: red;  
}  
  
#unikalus {color: blue;}
```

```
<p class="demesio large">  
    Šis paragrafas susietas su dviem klasėms.  
</p>  
<p class="large" id="unikalus">  
    Šis paragrafas susietas su viena klase.  
</p>
```

<https://flukeout.github.io/>

# CSS selektoriai

Svarbu suprasti, kad CSS selektoriai yra tam, kad pasakyti naršyklei, kuriuos HTML elementus turi paveikti taisyklės, aprašytos CSS faile.

Sugalvokime pasaką apie tris spalvingas mašinas. Kiekviena mašina turi savo unikalų vardą: **Raudonąjā**, **Mėlynąjā** ir **Žaliojā**.

Vardai:

- **Raudonoji** mašina
- **Mėlynoji** mašina
- **Žalioji** mašina

Ypatybės/Savybės:

- **Raudonoji mašina** yra ryškiai raudona ir greita kaip vėjas.
- **Mėlynoji mašina** yra šviesiai mėlyna ir gali važiuoti labai toli.
- **Žalioji mašina** yra tamsiai žalia ir vežti sunkius krovinius.

CSS selektorius galima palyginti su būdu, kaip pasirinkti skirtinges mašinas. Pvz., norint pasirinkti Raudonąjā mašiną, reikia pasakyti "**Raudonoji mašina**". Tai panašu į tai, kaip mes pasakoje kalbame apie kiekvieną mašiną pagal jų vardą.

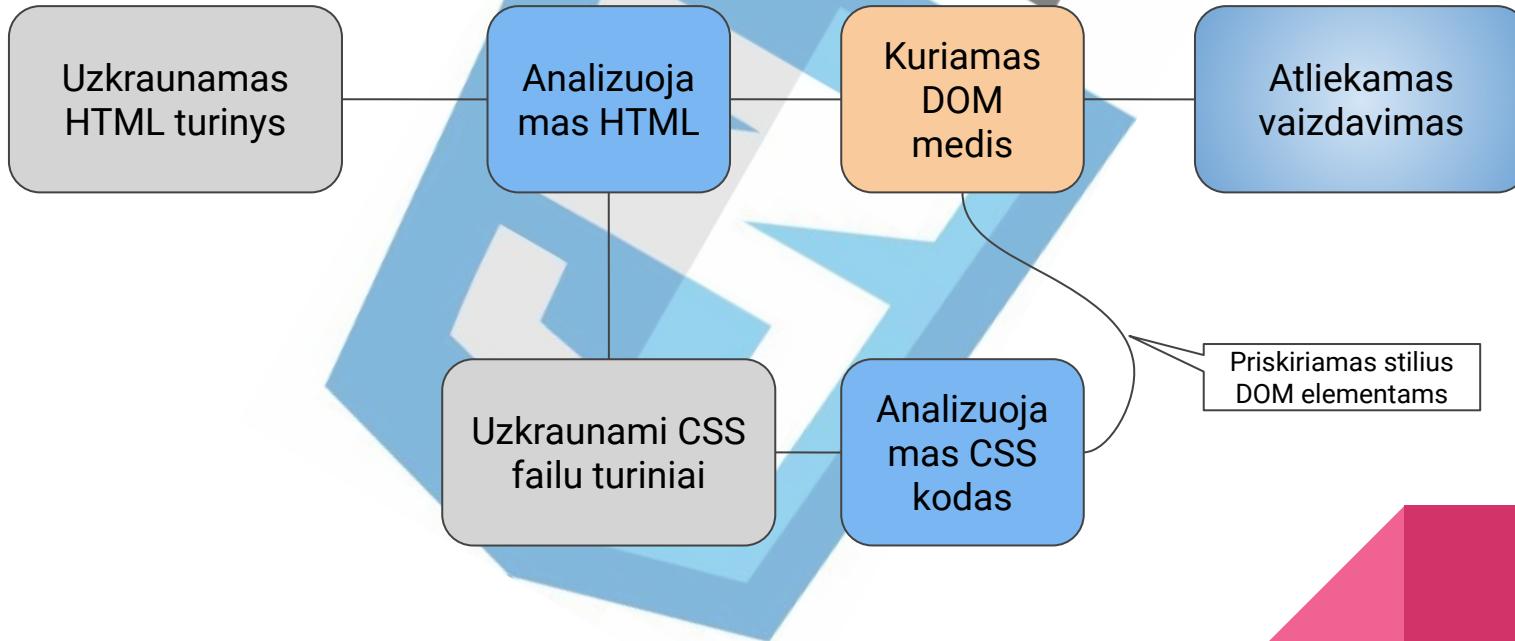
# CSS selektorų praktika

CSS Diner - Where we feast on CSS Selectors!

<https://flukeout.github.io/>



# Kaip veikia CSS ?



# Inline CSS, Internal CSS or External CSS

Inline CSS :

```
<p style="color: blue;">This is a paragraph.</p>
```

Internal CSS :

```
<head>  
  <style>  
    p {  
      color: red;  
    }  
  </style>  
</head>
```

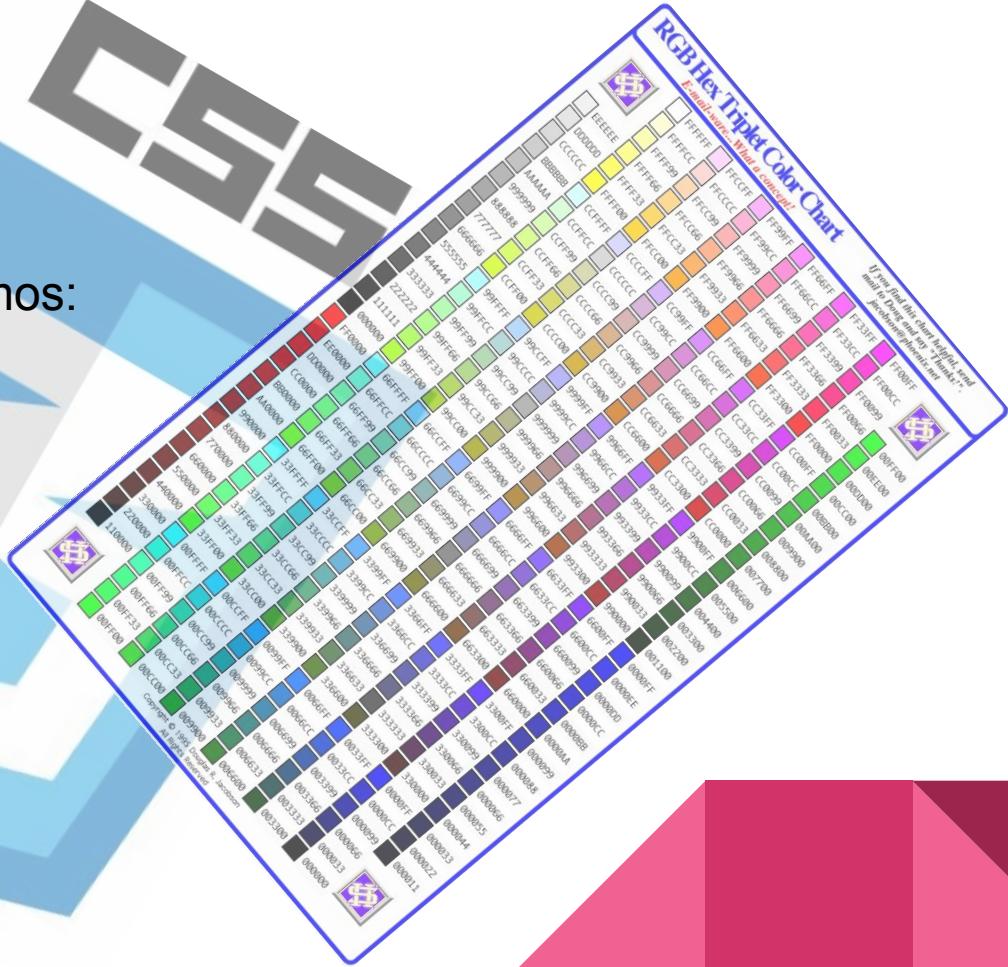
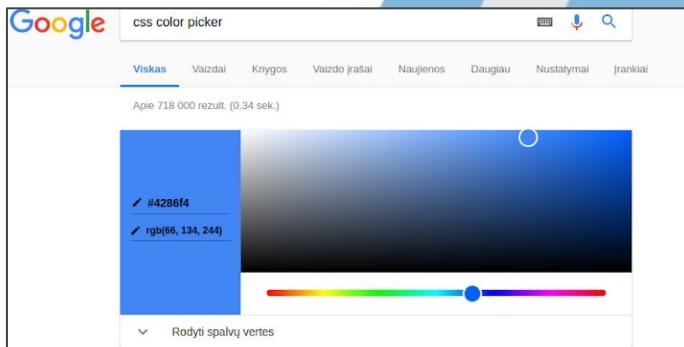
External CSS :

```
<head>  
  <link rel="stylesheet" href="mystyle.css">  
</head>
```

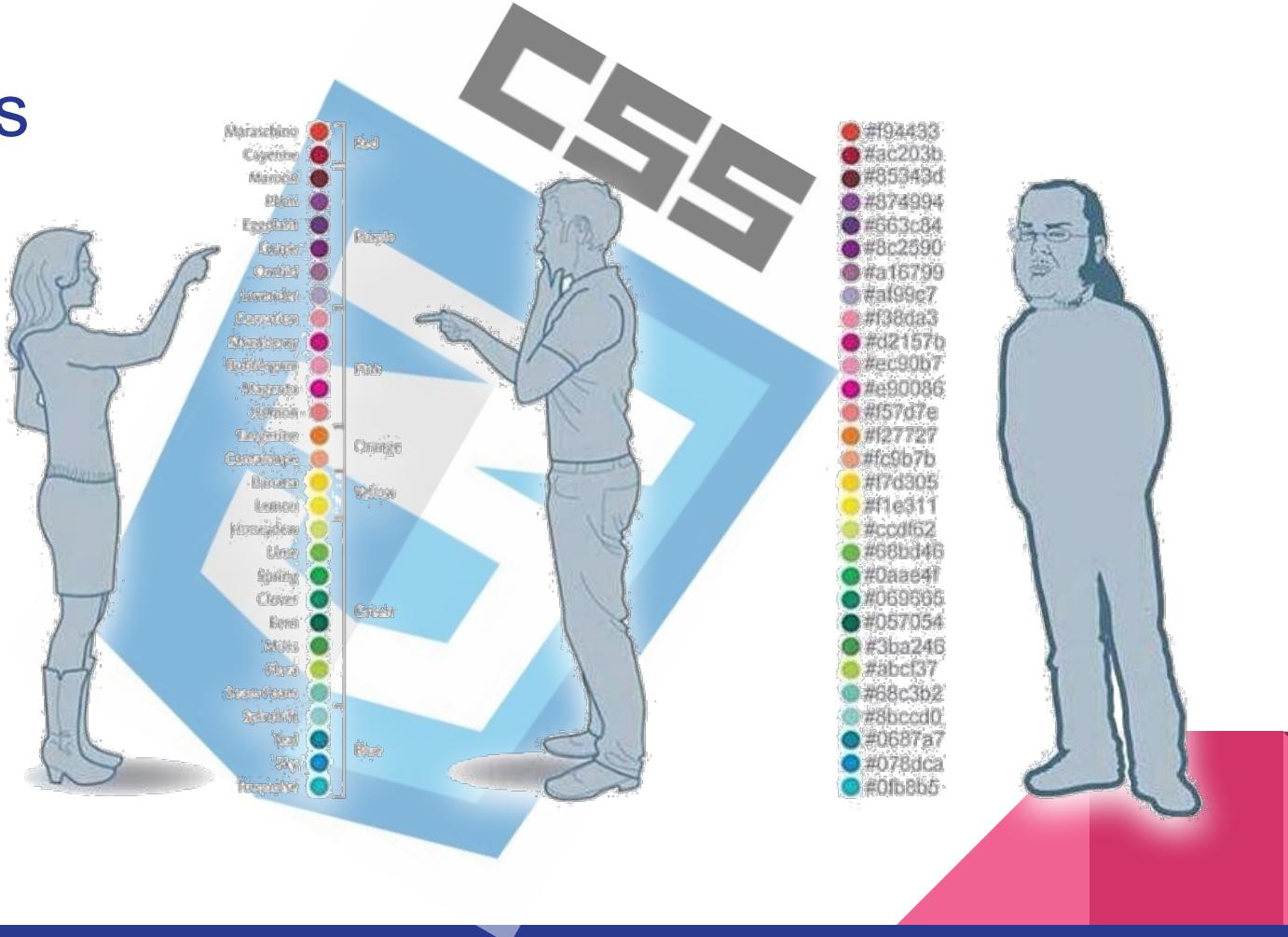
# Spalvos

CSS spalvos dažniausiai nurodomos:

- Spalvos pavadinimu - pvz. "red"
- RGB reikšme - pvz."rgb(255, 0, 0)"
- HEX kodu- pvz. "#ff0000"



# Spalvos



# Spalvos

```
#chuck-norris {  
    color: #BADA55;  
}
```

Joke

# Formatavimas

Teksto formatavimą apima:

- Spalva
- FontFamily (Šriftas)
- Dydis
- Lygiavimas
- Dekoracijos
- Transformacijos
- Pirmos eilutės atitraukimas
- Tarpavimai
- Kryptis
- Šešėliavimas

```
body {  
    color: black;  
    font-family: verdana;  
    font-size: 20px;  
    text-align: center;  
    text-decoration: line-through;  
    text-transform: capitalize;  
    text-indent: 50px;  
    letter-spacing: 3px;  
    line-height: 0.8;  
    word-spacing: 10px;  
    direction: rtl;  
    text-shadow: 3px 2px red;  
}
```

# Teksto nuleistinės (*Descenders*)



# CSS Normalize

„Normalize.css“ tai CSS failas, verčiantis naršykles visus elementus pateikti nuosekliau ir atsižvelgiant į šiuolaikinius standartus.

```
<link rel="stylesheet"  
      href="https://cdnjs.cloudflare.com/ajax/libs/normalize/7.0.0/normalize.min.css">
```

Matmenys, rėmeliai, pozicionavimas, CSS Grid



Tip



Nedaryk kas liepiama.

Tip

chen000

how to draw a sheep: draw a cloud, legs, a circle for the head and there you have it  
a sheep

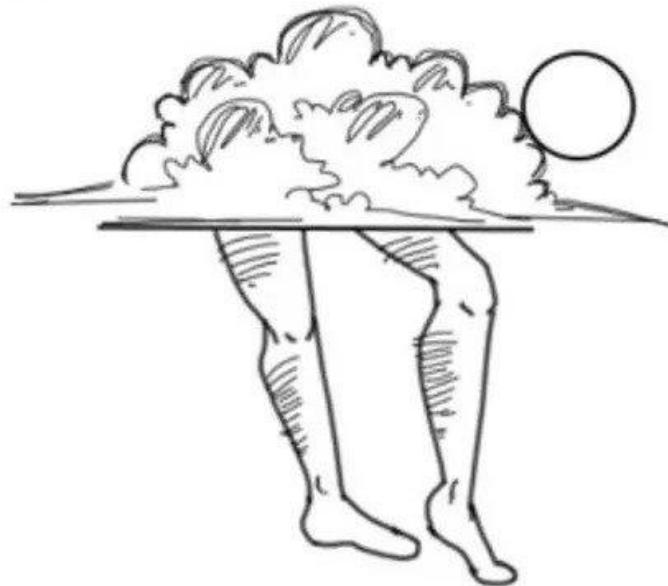
chen000

someone draw a sheep using these instructions

Tip



trombono



this rly helped i think this is the best sheep i have EVER drawn!!!

Tip



css

Klausk, net jei ir atrodo akivaizdu

# Elementų vaizdavimas

Kiekvienas HTML elementas turi numatytają vaizdavimo vertę, priklausomai nuo kokio tipo elemento jis yra. Pagal nutylėjimą vaizdavimo vertę daugumai elementų yra **block** arba **inline**.

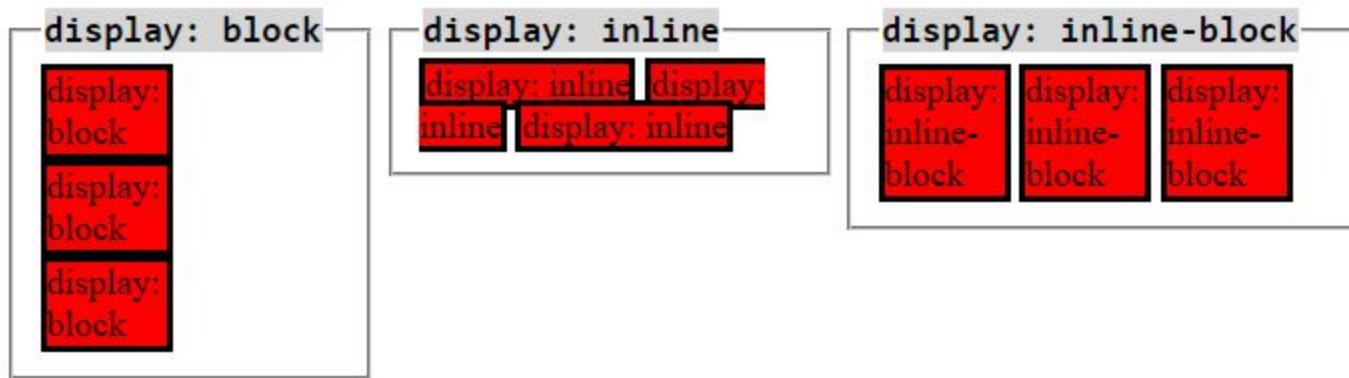
**Block** tipo elementai:

- <div>
- <h1> - <h6>
- <p>
- <form>
- <header>
- <footer>
- <section>

**Inline** tipo elementai:

- <span>
- <a>
- <img>

# Elementų vaizdavimas



# Rémeliai



A dotted border.

A dashed border.

A solid border.

A double border.

A groove border. The effect depends on the border-color value.

A ridge border. The effect depends on the border-color value.

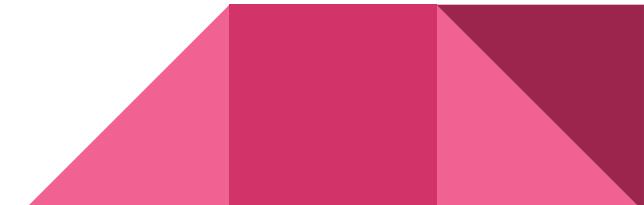
An inset border. The effect depends on the border-color value.

An outset border. The effect depends on the border-color value.

No border.

A hidden border.

A mixed border.



# Matmenys ir Rémeliai

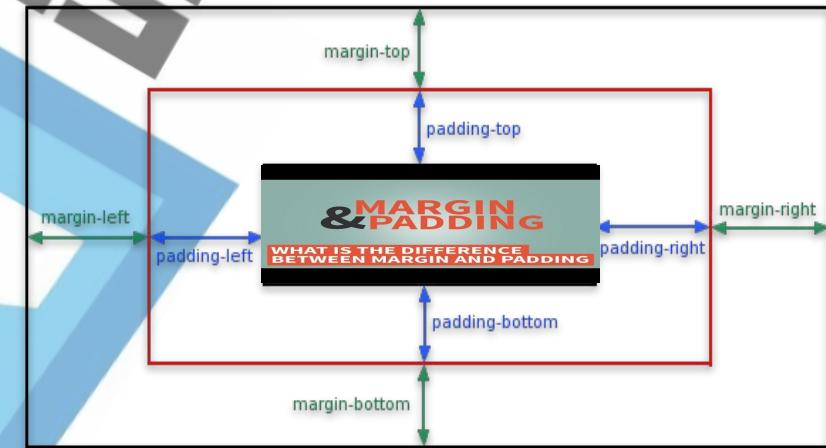
I have borders on all sides.

I have a red bottom border

I have rounded borders.

I have a blue left border.

Centering elements: <http://howtocenterincss.com>





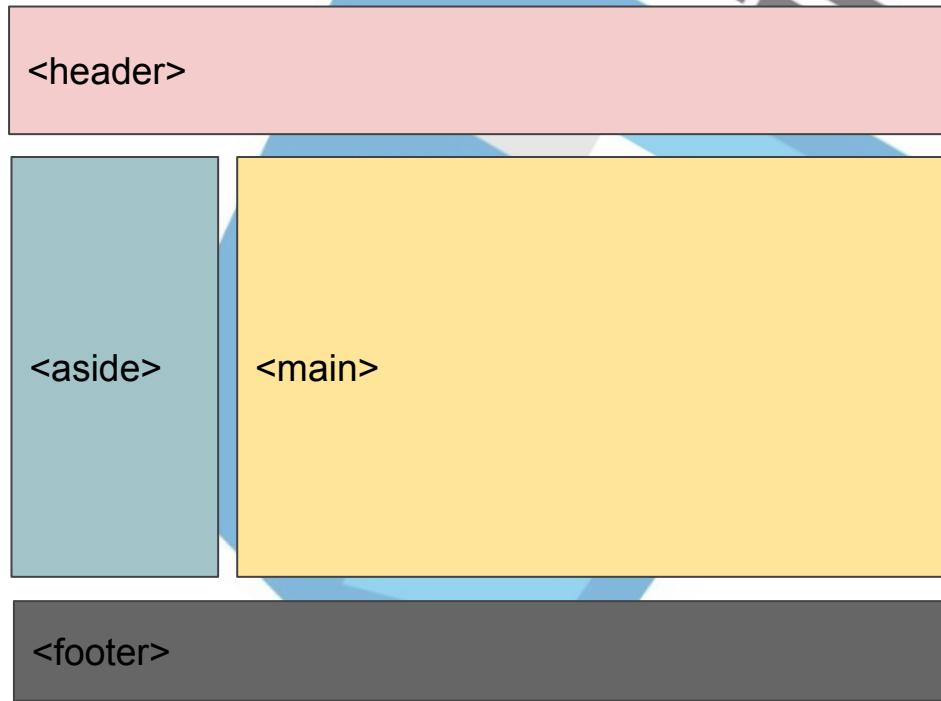
Joke

```
#titanic {  
    float: none;  
}
```

# Šablono spalvinimas



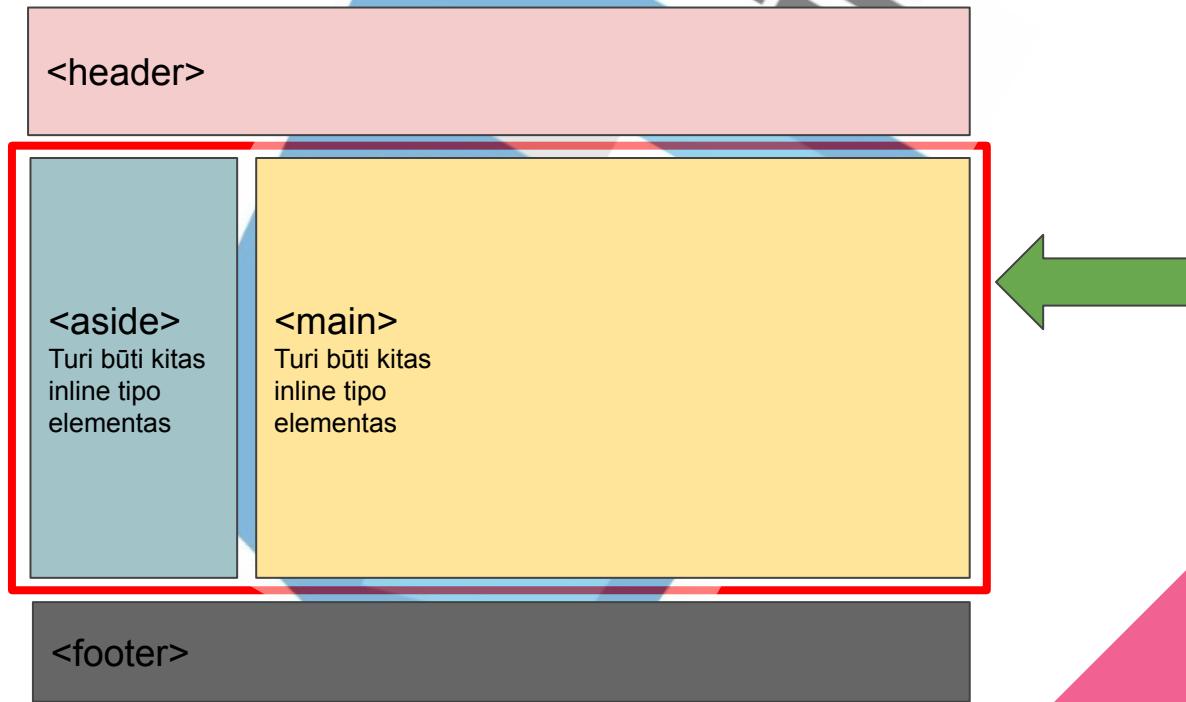
# Šablono dėliojimas



# Šablono spalvinimas



# Šablono dėliojimas



# Elementų pozicijos

Pozicija nurodo elementų padėties nustatymo tipą (statinis, santykinis, fiksuotas arba absolitusis). [LINK](#) arba [LINK](#)

```
div {  
    position: static;  
    ...  
    position: relative;  
    ...  
    position: fixed;  
    ...  
    position: absolute;  
    ...  
    position: sticky;  
}
```

```
img {  
    position: absolute;  
    left: 0;  
    top: 0;  
    z-index: -1;  
}
```

# Užduotis: Lenta



css

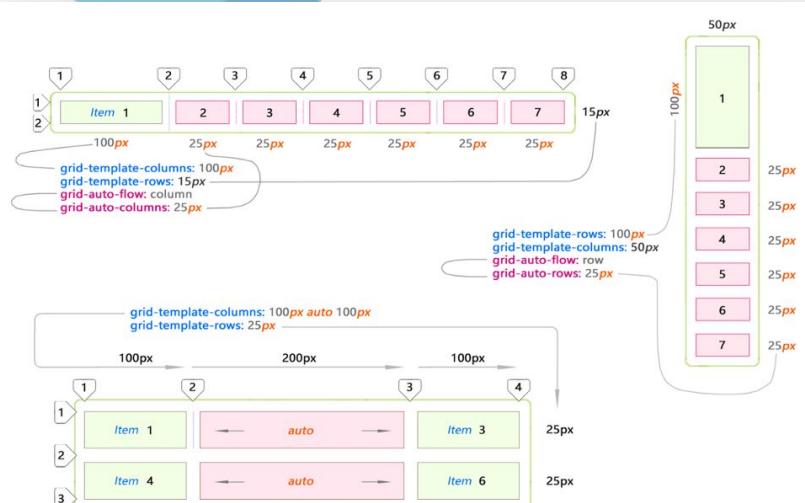
EMMET:  
div.sachmatai>div{<DIV \$>}\*25

```
.sachmatai > div {  
    width: 75px;  
    height: 75px;  
    margin: 5px;  
    padding: 10px;  
    ... ??? ...  
}
```

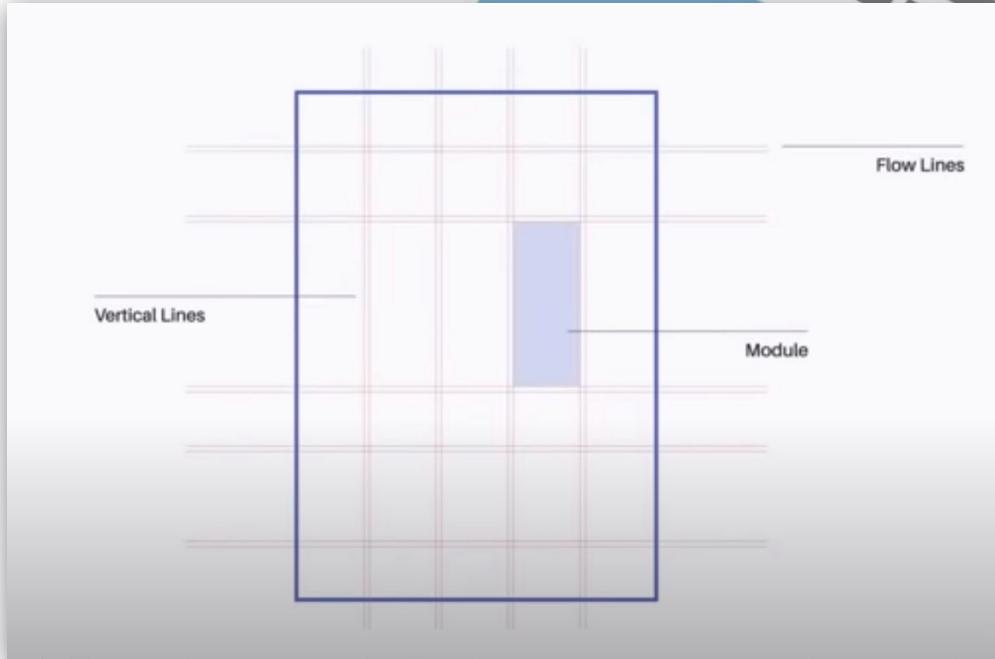
# CSS Grid

```
/* This is our CSS grid parent container */
div#grid {
  display: grid;
  grid-template-columns: 100px 100px 100px 100px 100px; /* 5 cols */
  grid-template-rows: 100px 100px 100px 100px; /* 4 rows */
}
```

Detaliau apie CSS Grid: <https://goo.gl/9oypjy>

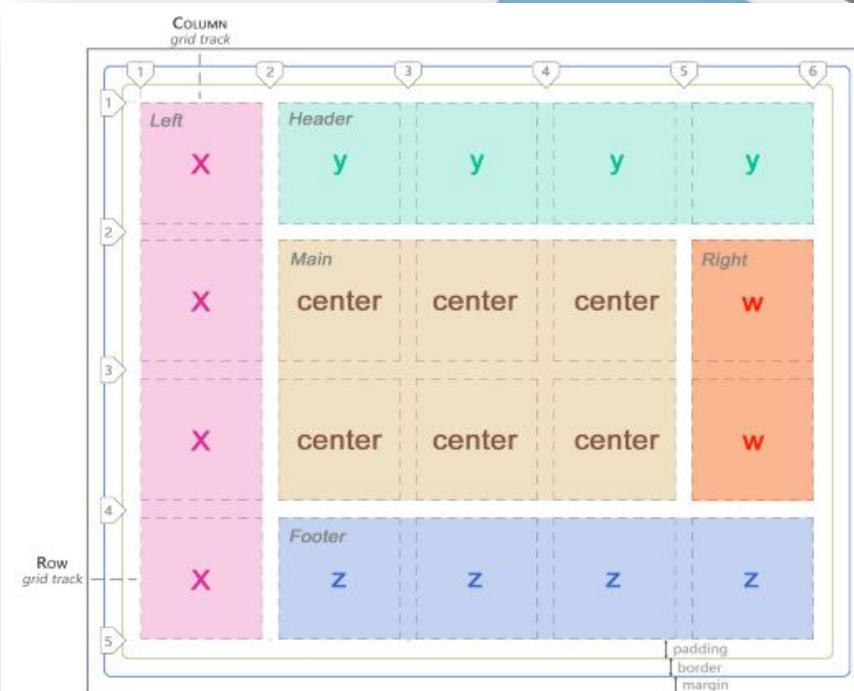


# CSS Grid



Complete Course On Layout Design

# CSS Grid



## Area Layout:

`grid-template-areas:`

```
'x y y y'  
'x center center center w'  
'x center center center w'  
'x z z z';
```

## Element Layout:

`<div id = "grid">`

```
<div style = "grid-area: x"> Left </div>  
<div style = "grid-area: y"> Header </div>  
<div style = "grid-area: z"> Footer </div>  
<div style = "grid-area: w"> Right </div>  
<div style = "grid-area: center"> Main </div>  
</div>
```

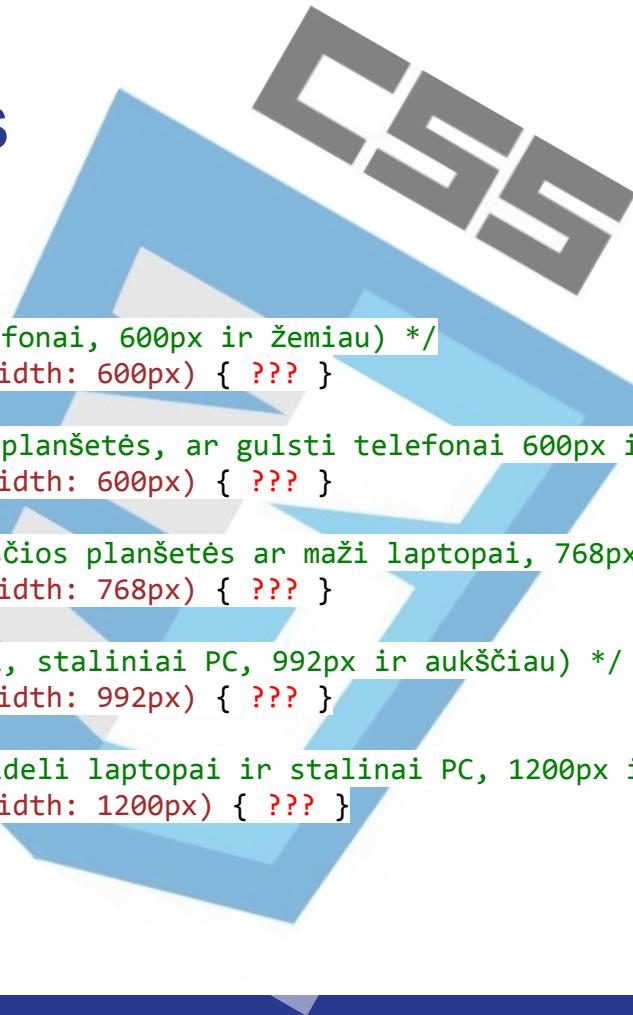
# RWD ?

## Responsive Web Design



[https://www.w3schools.com/css/css\\_rwd\\_templates.asp](https://www.w3schools.com/css/css_rwd_templates.asp)

# Media queries



```
/* Extra maži įrenginiai (telefonai, 600px ir žemiau) */
@media only screen and (max-width: 600px) { ??? }

/* Maži įrenginiai (statmenos planšetės, ar gulsti telefonai 600px ir aukščiau) */
@media only screen and (min-width: 600px) { ??? }

/* Vidutiniai įrenginiai (gulsčios planšetės ar maži laptopai, 768px ir aukščiau) */
@media only screen and (min-width: 768px) { ??? }

/* Dideli įrenginiai (laptopai, staliniai PC, 992px ir aukščiau) */
@media only screen and (min-width: 992px) { ??? }

/* Extra dideli įrenginiai (dideli laptopai ir stalinai PC, 1200px ir aukščiau) */
@media only screen and (min-width: 1200px) { ??? }
```

# Media queries

...

```
/* Dideli įrenginiai (laptopai, stalinių PC, 992px ir aukščiau) */  
@media only screen and (min-width: 992px) {
```

```
    body {  
        background-color: red;  
    }
```

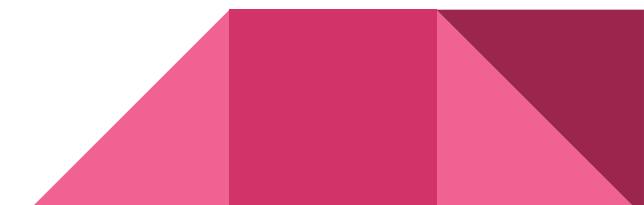
```
    div .klase {  
        width: 200px;  
    }
```

```
    #unikalus_elementas {  
        display: block;  
        width: 100%;  
    }
```

```
}
```

...

CS5



# Viewport



BE



SU

# Viewport

Tai specialus HTML **<meta>** elementas, kuris naudojamas nurodyti naršyklės viewport nustatymus. Tai yra labai svarbi eilutė, ypač kuriant adaptyvūjį dizainą svetainese, kad būtų užtikrintas tinkamas turinio vaizdas ir prisitaikymas prie įvairių įrenginių ekrano dydžių.

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta name="viewport" content="width=device-width, initial-scale=1.0">
5   
6 </head>
7 <body>
8   
9 </body>
10 </html>
```

# Viewport

- **name="viewport"** - tai nustato, kad tai yra viewport nustatymų meta informacija.
- **content="width=device-width, initial-scale=1.0"** - tai yra viewport nustatymų reikšmės. Jame yra du nustatymai:
  - **width=device-width** - šis nustatymas pritaiko viewport plotį pagal įrenginio ekrano plotį. Tai užtikrina, kad viewport plotis atitiks įrenginio ekraną, taip suteikiant galimybę tinkamai rodyti turinį be horizontalaus slinkimo ar išstempimo.
  - **initial-scale=1.0** - šis nustatymas nurodo, kad pradinis mastelis (pradinės prieigos stadijoje) bus 1.0. Tai reiškia, kad turinys bus rodomas natūraliu dydžiu be pradinio padidinimo ar sumažinimo.

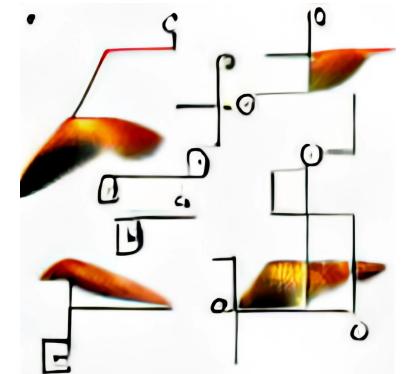
```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta name="viewport" content="width=device-width, initial-scale=1.0">
5   <!-- kitos meta informacijos ir stiliai -->
6 </head>
7 <body>
8   <!-- turinio dalys -->
9 </body>
10 </html>
```

# Pseudo klasės

Pseudo-klasės yra naudojamos apibrėžti ypatingos būsenos elementams.

Pavyzdžiu:

- Stilizuoti elementą, kai vartotojas virš jo su pelės kursoarium.
- Stilizuoti aplankytą ir neaplankytą nuorodą.
- Stilizuoti elementą, kai jis tampa aktyvus/sufokusuotas
- ...



```
selector:pseudo-class {  
    property:value;  
}
```

# Animacija

CSS3 animacija leidžia animaciją daugumai HTML elementų, nenaudojant JavaScript arba "Flash"!

```
/* Animacijos kodas */
@keyframes mano_animacija {
    0% {background-color: red;}
    50% {background-color: green;}
    100% {background-color: yellow;}
}

/* Elementas kuriam taikysime animaciją */
#elementas {
    width: 100px;
    height: 100px;
    background-color: red;
    animation: mano_animacija 4s infinite alternate;
}
```

# Animacija

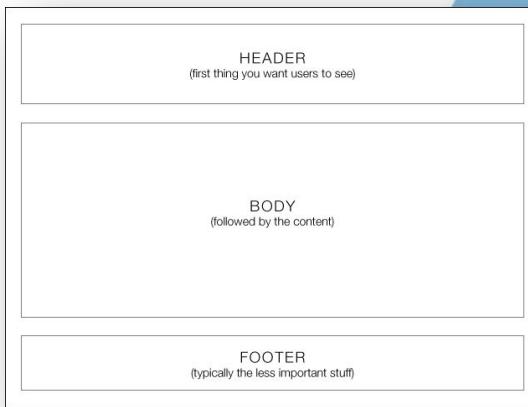
Joke

```
.government {  
    transition: all 4yr ease-out;  
}
```

# Kita...

## WireFramingas:

<https://webdesign.tutsplus.com/articles/a-beginners-guide-to-wireframing--webdesign-7399>



Home  
About  
Services  
Case Studies  
Testimonials  
F.A.Q.  
Say Hello

building great communities powered by drupal

Introducing...  
WebWise Solutions is dedicated to one goal: making your customers insanely happy with your products.  
Everything we do is designed to attract, convert, delight, tantalize, and educate people so they can't imagine life without you.

How do we do it? We have spent the last twelve years creating corporate web sites and web communities, and have learned what works and what doesn't work.

Clients we serve

selected testimonial

Put us to work for you. We live to serve.  
Use our Talk to Us form to drop us a note.

What we specialise in  
We build engaging communities powered by the powerful open-source software, Drupal.

- Full Services Packages
- Editorial and Community Management Only
- Communities On-Demand

Advertising opportunities  
Helping clients with their communities gives us the occasional opportunity to match up advertisers and available space. Click here to view the inventory we are currently offering to M.

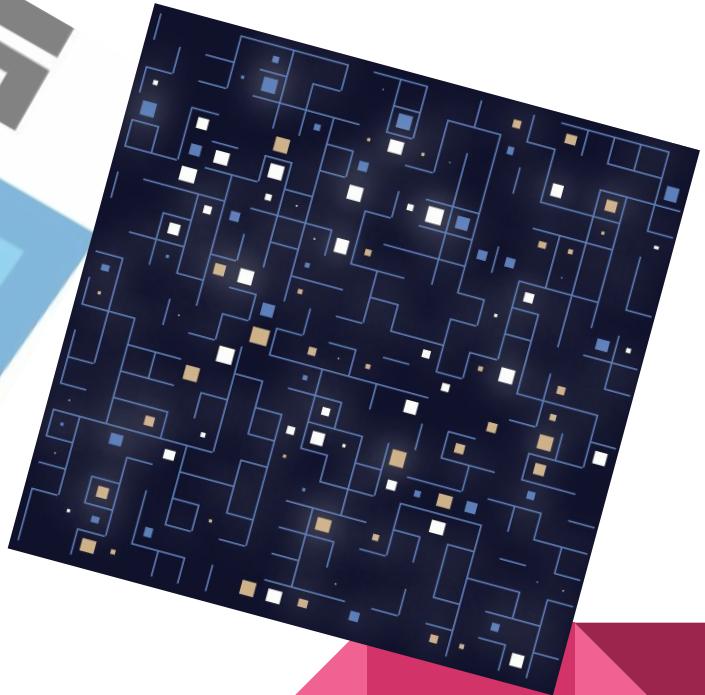
FAQ  
Q: What's the difference between a web community and an online forum?  
A: This is a common question we run into with our clients. As we consult with them about their needs, they often say, "Oh, we already have a forum, so I guess that's our community..." see full answer

Home  
About  
Services  
Case Studies  
Testimonials  
Say Hello  
FAQ

© WebWise Solutions 2009  
Terms, Conditions, and Notices.

<css-doodle />

css



# Elementų vaizdavimas (pavyzdžiai)

Naming: <http://getbem.com/naming/>

Flexbox: <https://yoksel.github.io/flex-cheatsheet/#display>

Box model: <https://learn.shayhowe.com/html-css/opening-the-box-model/>

# Inspiracijos

- <https://gfycat.com/gifs/detail/ShyMajorAlbatross>
- <https://www.wix.com/website/templates>

# Kas toliau ?



ANGULARJS  
Material

{less}

<https://www.w3schools.com/css>



w3schools.com

CSS

Tip nr.: 4

Sass

Bootstrap

LOGO

Search



Section 1

Section 2

Section 3

Section 4

Section 1

Section 2

Section 3

Section 4

Section 5

Section 6

Section 7

Section 8

Section 9

Section 10

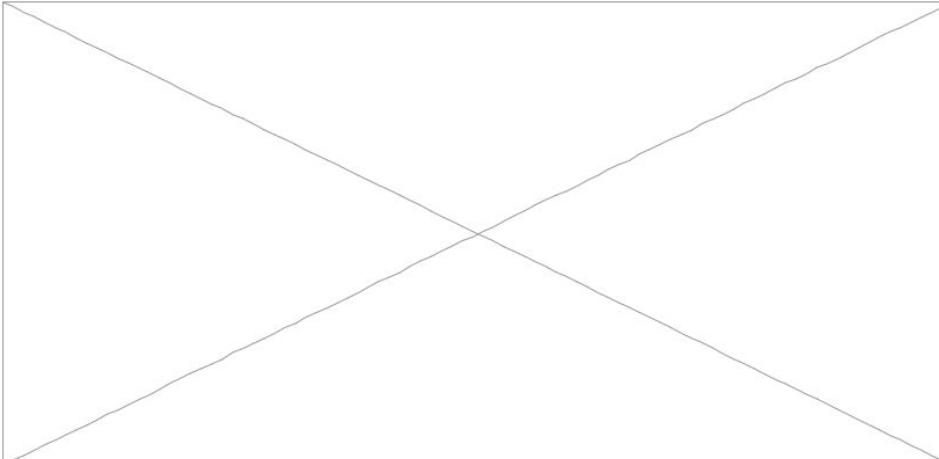
Section 11

Section 12

Section 13

Section 14

Section 15



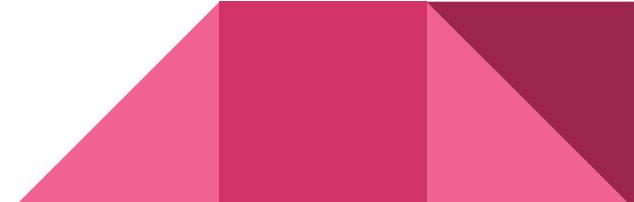
Lore ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum. Sed ut perspiciatis unde omnis iste natus error sit voluptatem accusantium doloremque laudantium, totam rem aperiam, eaque ipsa quae ab illo inventore veritatis et quasi architecto beatae vitae dicta sunt explicabo. Nemo enim ipsam voluptatem quia voluptas sit aspernatur aut odit aut fugit, sed quia consequuntur magni dolores eos qui ratione voluptatem sequi nesciunt.

External Links

JS



# JS (Javascript pagrindai), kintamieji, operatoriai, aritmetika



# Kas tai JavaScript ?



**JavaScript** – objektiškai orientuota skriptų programavimo kalba, besiremianti prototipų principu.

Dažniausiai kalba naudojama internetinių puslapių interaktyvumo realizacijai, bet taip pat naudojama ir kaip galimybė skriptais manipuliuoti tam tikromis programomis.

# Rekomenduojami įrankiai

- **Dev Tools:** DOM inspektorius ir JS debugeris
- **npm:** Standartinis open-source paketų repositorių tvarkymas. (Node Package Manager)
- **Git ir GitHub:** Tai populiarusia VCS(version control system) failų versijavimo sistema ir platforma.
- **Atom, VSCode, or PHPStorm:** Jums bus reikalingas koks nors IDE redaktorius. Atom ir VSCode šiuo metu yra populiarusieji JS redaktoriai. PHPStorm tai dar vienas sprendimas, užtikrinantis labai kokybišką įrankių palaikymą.
- **ESLint:** Sintaksės klaidų ir stiliaus sugaudymas. Po kodo peržiūros (Code Review) ir TDD, tai trečias geriausias dalykas, kurį galite padaryti kad sumažintumėte kodo klaidų kiekį.

# Frameworks (Karkasai) ?



ANGULARJS  
by Google



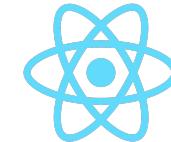
Redux



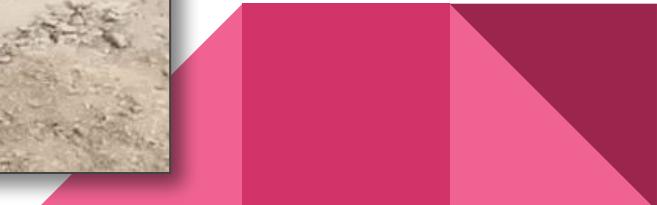
ReactiveX



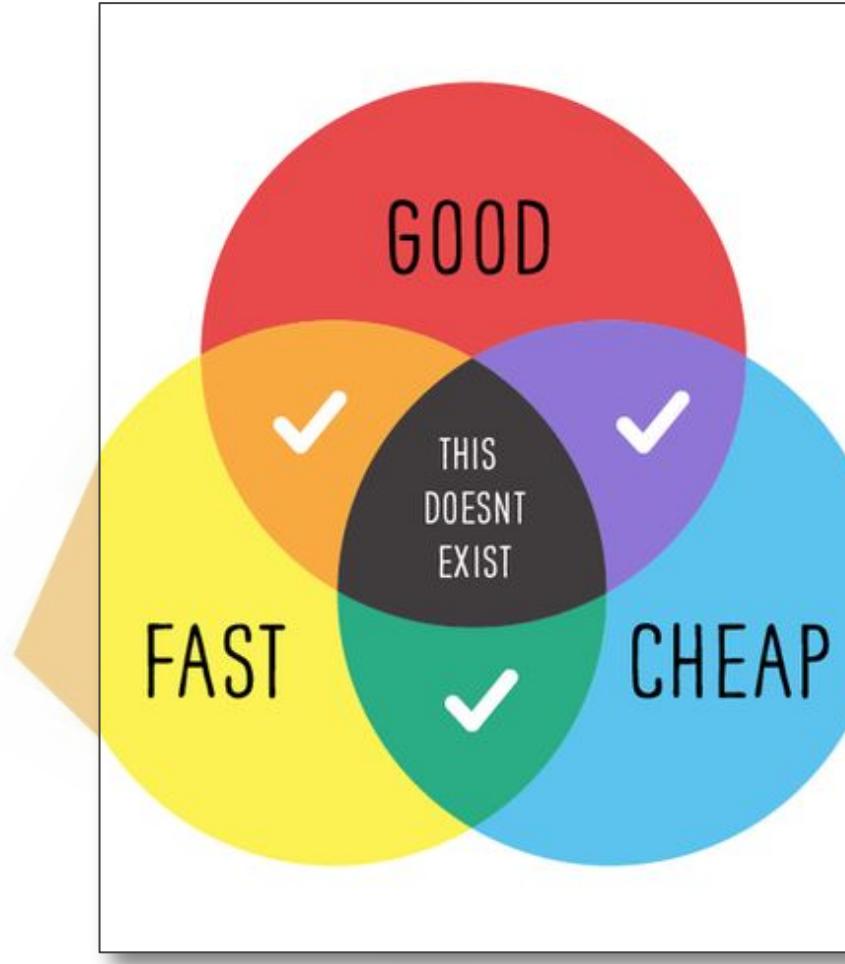
ember



jQuery



Tip



# Detaliau apie JavaScript ?

Kintamieji ir jų tipai

[https://www.w3schools.com/js/js\\_variables.asp](https://www.w3schools.com/js/js_variables.asp)

Masyvai

[https://www.w3schools.com/js/js\\_arrays.asp](https://www.w3schools.com/js/js_arrays.asp)

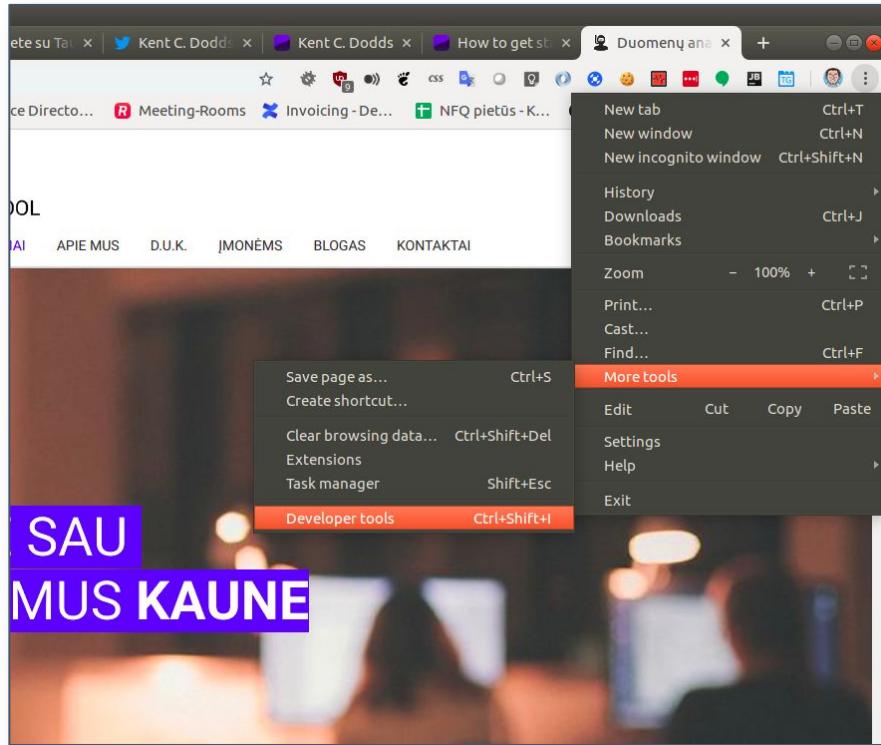
Output

[https://www.w3schools.com/js/js\\_output.asp](https://www.w3schools.com/js/js_output.asp)

Funkcijos

[https://www.w3schools.com/js/js\\_functions.asp](https://www.w3schools.com/js/js_functions.asp)

# Developer tools

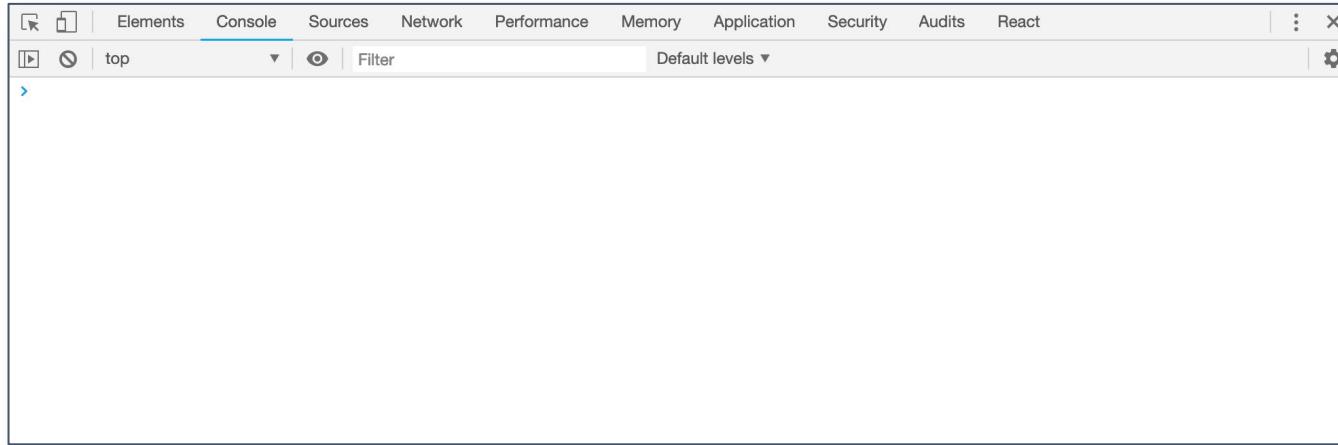


# Developer tools

The screenshot shows a browser developer tools window with the following panels:

- Elements Panel:** Shows the DOM tree. The `<body>` node is selected, highlighted in grey. The tree includes:
  - `<!doctype html>`
  - `<html lang="en" data-react-helmet="lang">`
  - `> <head>...</head>`
  - `... <body> == $0`
    - `<noscript>This site runs best with JavaScript enabled.</noscript>`
    - `> <div id="__gatsby" style="background-color: #fafafa;">...`
    - `<script src="/commons.js" style="color: #000; font-family: Inter Regular, sans-serif;">`
  - `</body>`
  - `</html>`
- Styles Panel:** Shows the CSS styles applied to the selected element (`<body>`). It includes:
  - element.style {**
  - body {**
    - `color: #000; font-family: Inter Regular, sans-serif;`
    - `background-color: #fafafa;`
  - html, body {**
    - `font-family: Inter Regular, sans-serif;`
    - `font-style: normal;`
    - `padding: 0;`
    - `margin: 0;`
  - body {**
    - `color: #000; font-family: 'Inter Regular', sans-serif;`

# Developer tools



# Developer tools

```
> 1 + 2
<- 3
>
```

```
function add(a, b) {
    return a + b;
}
const result = add(1, 2);
const message = '1 + 2 atsakymas ';
console.log(message, result);
```

# Kintamasis



A = '';



A = 'Malkos';



B = '';

B = null;



C = '';



C = {'Jonas', 'Petras'};

# Kintamasis



=



A = 'Jonas';



=



B = 'Petras';



=



C = { A , B };

# Kas yra aplikacija?



Duomenys



Darbas su duomenimis



Rezultatas

# Kintamieji ir jų tipai

```
let x = 5;
const y = 6;
let z = x + y; // 11

let pi = 3.14;
let person = 'John Doe';
let cars = ['Saab', 'Volvo', 'BMW'];

person = {
  firstName:'John',
  lastName:'Doe',
  age:50,
  eyeColor:'blue'
};

-----
let a = '5' + 2 + 3; // ???
let b = 2 + 3 + '5'; // ???
```

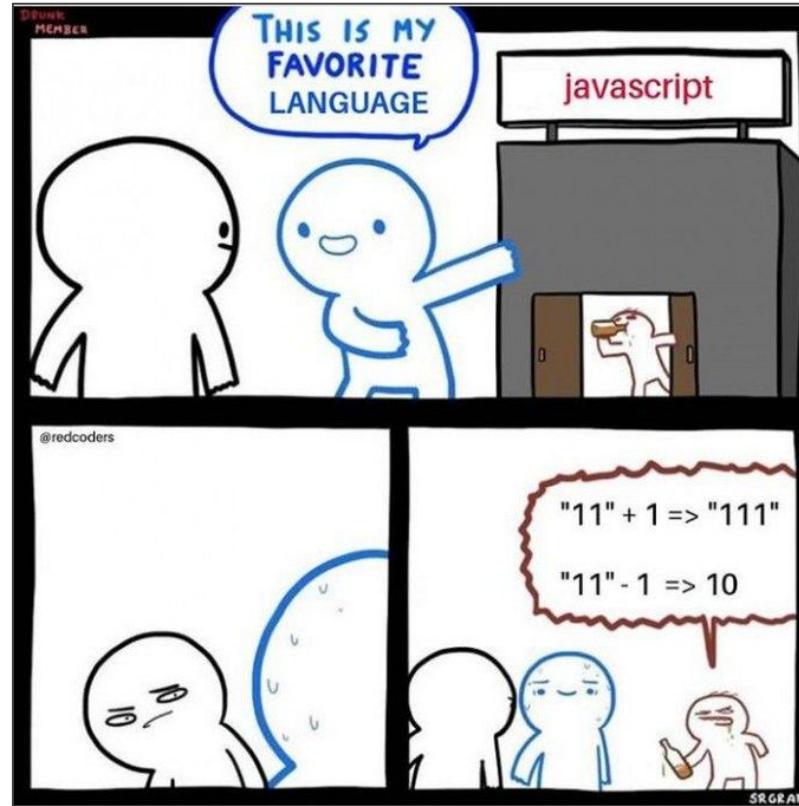
```
var x;           // undefined
x = 5;          // Number
x = 'John';     // String

// Viena vienguba kabutė, tarp dvigubų kabučių
var answer1 = "It's alright";

// Dvi viengubos kabutės, tarp dvigubų kabučių
var answer2 = "He is called 'Johnny'";

// Dvi dvigubos kabutės, tarp viengubų kabučių
var answer3 = 'He is called "Johnny"';
```

# Kintamieji ir jų tipai



# Kintamieji ir jų tipai

```
typeof 'John'           // "string"  
typeof 3.14             // "number"  
typeof true              // "boolean"  
typeof false             // "boolean"  
typeof x                 // "undefined"  
(Jeigu x prieš tai neturėjo reikšmės)
```

```
typeof {name:'John', age:34} // "object"  
typeof [1,2,3,4]           // "object"  
typeof null                // "object"  
typeof function myFunc(){} // "function"
```

## Sudėtingesni tipai

`typeof` operatorius gali gražinti viena arba iš dviejų tipų:

- `function`
- `object`

`typeof` operatorius objektams, masyvams, and null gražins `object`.

`typeof` operatoriaus atsakymas funkcijoms bus `function`.

# Aritmetiniai Operatoriai

Aritmetiniai operatoriai naudojami skaičiams atliekant aritmetinius veiksmus:

## **Operatorius      Aprašymas**

**+**              Sudėties

**-**              Atimties

**\***              Daugybos

**\*\***              kėlimas laipsniu (ES2016)

**/**              Dalybos

**%**              Dalybos liekana

**++**              Didinimas

**--**              Mažinimas

# Priskyrimo Operatoriai

Priskyrimo operatoriai priskiria reikšmes „JavaScript“ kintamiesiems.

Operator	Example	Same As
=	<code>x = y</code>	<code>x = y</code>
<code>+=</code>	<code>x += y</code>	<code>x = x + y</code>
<code>-=</code>	<code>x -= y</code>	<code>x = x - y</code>
<code>*=</code>	<code>x *= y</code>	<code>x = x * y</code>
<code>/=</code>	<code>x /= y</code>	<code>x = x / y</code>
<code>%=</code>	<code>x %= y</code>	<code>x = x % y</code>
<code>**=</code>	<code>x **= y</code>	<code>x = x ** y</code>



**JavaScript testai savarankiškam darbui**

[https://www.w3schools.com/js/exercise\\_js.asp](https://www.w3schools.com/js/exercise_js.asp)

# Inline JS, Internal JS or External JS

Inline JS:

```
<p onclick="alert('Labas')">Paspausk mane</p>
```

Internal JS :

```
<head>
  <script>
    alert('Labas');
  </script>
</head>
```

External JS:

```
<head> // bloga praktika JS kodą dėti į HEAD dalį
  <script src="manoKodas.js" defer>
</head>
```

# Kas paskutinis tas ir tėvas

**Svarbu** kad java script failų deklaravimas būtų aprašomas HTML failo pabaigoje.

```
....  
  
</div>  
  
<script src="/js/pirmas_failas.js"></script>  
  
<script src="/js/antras_failas.js"></script>  
  
</body>  
  
</html>
```

# Užduotis Nr. 1



Sukurkite kintamąjį a ir priskirkite jam reikšmę 1.  
Atspausdinkite kintamojo reikšmė kartu su tekstu  
'Kintamojo a reikšmė: '. Tuomet pskeiskite kintamojo  
reikšmę į 2 atspausdinkite rezultatą tuo pačiu formatu.

Rez.: :

Kintamojo a reikšmė: 1

Kintamojo a reikšmė: 2

## Užduotis Nr. 2

Sukurkite kintamajį **b** ir priskirkite jam reikšmę 'Jūsų vardas'. Atspausdinkite kintamojo reikšmę kartu su kintamojo **c** tekstu 'Mano vardas'

Rez.:

Mano vardas Vardenis Pavardenis

# Užduotis Nr. 3



Parašykite skriptą, kuris iš jūsų gimimo datos (*let metai, let mėnuo, let diena*) paskutinių skaitmenų sudarytų skaičių sumą ir ją parodykite developer tools konsolėje

PVZ-1: Jei gimimo data yra 1999 12 28, tai suma bus lygi 19

PVZ-2: Jei gimimo data yra 2000 01 01, tai suma bus lygi 2

*Hint: Dalyba*

1 Mano gimimo data yra 1955-9-15.

2 Paskutinių skaitmenų suma  $5 + 9 + 5 = 19$

# Užduotis Nr. 4



Markas ir Džonas nori palyginti savo KMI (Kūno Masės Indeksus), kuris yra apskaičiuojamas su šia formule: **KMI = svoris / ūgis \*\* 2 = svoris / (ūgis \* ūgis)**  
(svoris ir ūgis privalo naudoti metrinę sistemą)

1. Priskirkite Marko ir Džono svorius ir ūgius į kintamiesiems.
2. Apskaičiuokite abiejų KMI naudodamiesi formule (galite panaudoti abejes versijas)
3. Sukurkite Boolean tipo kintamaji 'markoDidesnisKMI' kuris saugos informacija ar Markas turi didesnį KMI nei Džonas

TESTINIAI DUOMENYS 1: Marko svoris 78 kg ir ūgis 1.69 m . Džono svoris 92 kg ir ūgis 1.95 m

TESTINIAI DUOMENYS 2: Marko svoris 95 kg ir ūgis 1.88 m . Džono svoris 85 kg ir ūgis 1.76 m

SĖKMĖS :)

## "Duomenų ir rezultatų išvedimas"



# Output 1 / 5

bandymas-1.html

```
<!DOCTYPE html>
<html>
<body>
<h1>Mano pirmas puslapis</h1>
<p>Mano pirmas paragrafas.</p>

<p id="demo">Cia bus naujas tekstas</p>

<script>
    let carName = 'Volvo';
    document.getElementById('demo').innerHTML = carName;
</script>

</body>
</html>
```

# Output 2 / 5

bandymas-2.html

```
<!DOCTYPE html>
<html>
<body>

<script>
    document.write(5 + 6);
</script>
<h1>Mano antras puslapis</h1>
<p>Mano antras paragrafas.</p>
<script>
    document.write(5 + 6);
</script>

</body>
</html>
```

# Output 3 / 5



## bandymas-3.html

---

```
<!DOCTYPE html>
<html>
<body>

<h1>Mano trečias puslapis</h1>
<p>Mano trečias paragrafas.</p>

<button type="button" onclick="let x = generuotiHtml();document.write(x)">
    Try it
</button>

</body>
</html>
```

# Output 4 / 5

bandymas-4.html

```
<!DOCTYPE html>
<html>
<body>

<h1>Mano ketvirtas puslapis</h1>
<p>Mano ketvirtas paragrafas.</p>

<script>
    window.alert('Labas pasauli');
</script>

</body>
</html>
```

# Output 5 / 5

bandymas-5.html

```
<!DOCTYPE html>
<html>
<body>

<h1>Mano penktas puslapis</h1>
<p>Mano penktas paragrafas.</p>

<script>
    console.log(5 + 6);
</script>

</body>
</html>
```

# JavaScript užduotis



Parašykite skriptą kuris iš dviejų taisyklingų daugiakampių viršūnių skaičius let n1 ir let n2 (pvz.: trikampio – 3 ir keturkampio – 4). Apskaičiuotų ir elemente kurio **id** yra 'ats' parodytu kiekvieno daugiakampio kampų sumą. Bei abiejų daugiakampių kampų sumą.

Vieno daugiakampio kampų suma skaičiuojama pagal formulę:

$$S = (n - 2) * 180$$

kur **n** – daugiakampio kampų skaičius.

# Sąlyga if / else / elseif



\* Ne JS kodas

# Sąlyga if / else / elseif

```
if (sąlyga1 || sąlyga2) {  
    // Veiksmai  
}
```

\*\*\*\*\*

```
if (sąlyga1 && sąlyga2) {  
    // Veiksmai  
} else {  
    // Kiti veiksmai jei neatitiko sąlygos  
}
```

\*\*\*\*\*

```
if (sąlyga) {  
    // Veiksmai  
} else if (sąlyga2) {  
    // Kiti veiksmai jei neatitiko pirmosios sąlygos bet atitiko antrają  
} else {  
    // Priešingu atveju jei neatitiko jokių sąlygų  
}
```

*JavaScript Comparison and Logical Operators*

*if(a = b) { /\* Suprantama kaip TRUE, net  
jei a yra 1, o b yra 2 \*/ }*

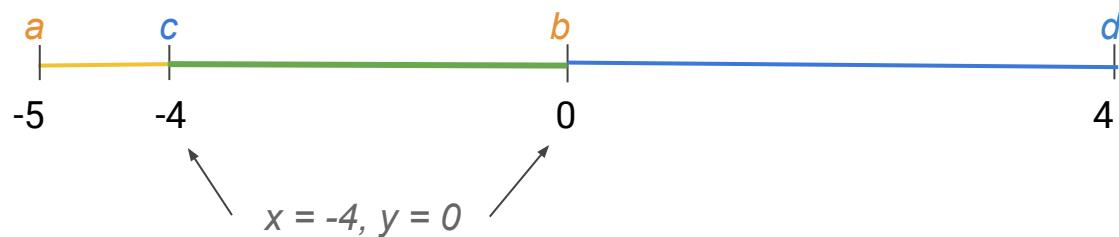
# Switch/Case

```
let zenklas = '*' ;  
  
switch (zenklas) {  
    case '+':  
        sudeti /* ... */;  
        break;  
    case '-':  
        atimti /* ... */;  
        break;  
    case '*':  
        dauginti /* ... */;  
        break;  
    case '/':  
        dalinti /* ... */;  
        break;  
    default:  
        console.log('Nepavyko');  
        break;  
}
```

## IF Užduotis Nr. 2

Duoti du sveikujų skaičių intervalai  $a \rightarrow b$  ir  $c \rightarrow d$ , kai  $a$  mažiau už  $b$  ir  $c$  mažiau už  $d$ . Sudarykite sąlygų seką, kuri nustatyta, ar egzistuoja šių intervalų sankirtą ir, jeigu sankirta egzistuoja, tai raskite jos rėžius  $x \rightarrow y$ .

Pvz., jei  $a = -5$ ,  $b = 0$ ,  $c = -4$ ;  $d = 4$ , tai  $x = -4$ ,  $y = 0$ .



## IF Užduotis Nr. 3

Turime tris kintamuosius **a**, **b** ir **c**. Reikia rasti, kuris iš trijų skaičių yra mažiausias, bet nėra lygus kitiem. Atsakymą išvesti į **#output** elementą.

// Pavyzdys:

Jei **a = 3**, **b = 1**, **c = 5**, tada turėtų grąžinti "**b**", nes "**b**" yra mažiausias ir nelygus kitiem skaičiams.

# CIKLAI

code

```
1 a = 1
2 while a < 10:
3     print(a)
4     a += 2
```

output

variables



```
1 a = 1
2 while a < 7 :
3     if(a % 2 == 0):
4         print(a, "is even")
5     else:
6         print(a, "is odd")
7     a += 1
```

code

output

variables

[www.penjee.com](http://www.penjee.com)

# CIKLAI

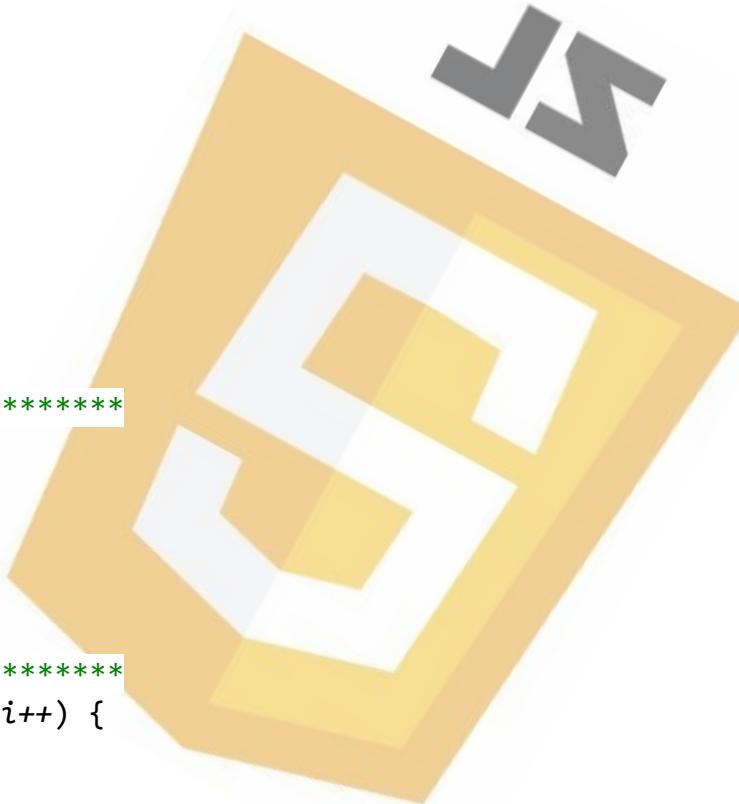


# Ciklai

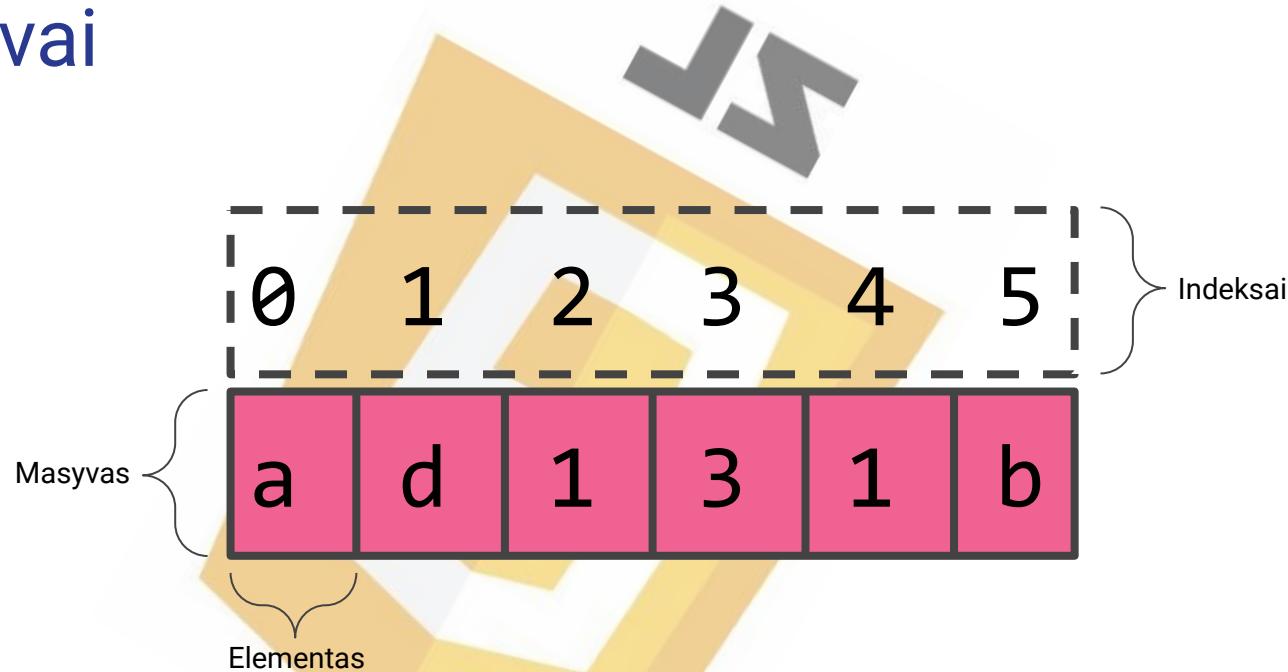
```
while (sąlyga) {  
    // Veiksmai  
}
```

```
*****  
do {  
    // Veiksmai  
} while(sąlyga);
```

```
*****  
for(let i = 0; i < 5; i++) {  
    // Veiksmai  
}
```



# Masyvai



<https://tome.app/tautiz/unleashing-the-power-of-javascript-arrays-clin4692v143dmu3bk7p91jqg>

# Masyvai

```
let nerykiuotasMasyvas = [9, 4, 6, 5, 8, 0, 7, 2, 3, 1];
let rykiuotasMasyvas = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10];
let stringuMasyvas = ['a', 'b', 'nulis', 'vienas', 'c', 'Vardas', 'adresas'];
let misrusMasyvas = [5, 'b', '0', 1, 'c', 777, {"vardas": "Jonas"}];
```

// Duomenų paėmimas iš masyvo

```
console.log(nerykiuotasMasyvas[0]);           // 9
console.log(rykiuotasMasyvas[0]);             // 1
console.log(nerykiuotasMasyvas[1]);           // 4
console.log(rykiuotasMasyvas[1]);             // 2
console.log(stringuMasyvas[3]);                // vienas
console.log(misrusMasyvas[2]);                 // '0' - bet tai nėra skaicius
console.log(misrusMasyvas[6].vardas);         // Jonas
```

# Skaičių masyvo rūšiavimas į atskirus masyvus

```
1 numbers = [12, 37, 5, 42, 8, 3]
2 even = []
3 odd = []
4 while len(numbers) > 0 :
5     number = numbers.pop()
6     if(number % 2 == 0):
7         even.append(number)
8     else:
9         odd.append(number)
```

[www.penjee.com](http://www.penjee.com)

# Masyvai

```
let array = ['a', 'b', 'c', 'd', 'e', 'f'];

for(let i = 0; i < array.length; i++) {

    let item = array[i];

    // ... aliekami veiksmai su kintamuoju item

    console.log(item);

}

// abcdef
```

# Naudingi metodai darbui su masyvu

	.map( □ → ○ )	→
	.filter( □ )	→
	.find( □ )	→
	.findIndex( □ )	→ 3
	.fill( 1, ○ )	→
	.copyWithin( 2, 0 )	→
	.some( □ )	→ true
	.every( □ )	→ false

# Elementų parametru/atributų gavimas

```
<span id="unikalusElementas">Labas pasauli</span>





<ul class="wrapper">
  <li class="element">Tekstas1</li>
  <li class="element">Tekstas2</li>
  <li class="element">Tekstas3</li>
</ul>
```

```
let tekstas = document.getElementById('unikalusElementas');

let paveiksleliai = document.getElementsByClassName('galerijosFoto');

let reklamos = document.querySelectorAll('.wrapper .element');
```

# Elementų parametru/atributų keitimas

```
tekstas.innerHTML = tekstas.innerHTML + '!';  
  
paveiksleliai[0].src = '/img/lempute_off.png';  
  
reklamos[1].style.backgroundColor = 'red';  
  
reklamos[2].style.display = 'none'; // 'block'
```

# Elementų parametru/atributų vaizdavimas

```
console.log(tekstas.innerHTML);
let i = 1;
console.log(paveiksleliai[i].src);

console.log(reklamos[0].style.display);
```

# Užduotis

- Suskaičiuoti kiek mūsų HTML faile yra <img> tagų. Rezultatas išvedamas konsolėje.
- Tą patį rezultatą išvesti į elementą kurio **id="uuid"**.  
*(pvz.: "Šiame puslapyje yra 8 paveikslėliai")*
- Paspaudus mygtuką, Window Alert'e atspausdinti paveikslėlių kiekj.
- Pakeisti kas antro paveikslėlio alt atributo reikšmę į žodj "testas".
- Elementui su "uuid" id reikšme, pakeisti css į  
`{font-size: 24px; background-color: green; width: 200px; height:200px;}`



uzduoties\_failas.html

# JSON/Object

```
let adresai = [
  {
    salis: "Lietuva",
    miestas: "Kaunas",
    gatve: "Vyrauto pr.",
    namoNr: 24,
    papildoma: "Duru kolas 1234"
  },
  {
    rajonas: "Vilniaus raj."
  }
];
let asmuo = {
  vardas: "Jonas",
  pavarde: "Jonaitis",
  kodas: "61212120001",
  adresas: adresai[0],
  papildoma: null
};
```

```
console.log(asmuo.vardas); // Jonas
console.log(typeof asmuo.adresas); // Object
console.log(asmuo.adresas.miestas); // Kaunas
let str = '{"salis": "Lietuva", "miest": "VL"}';
console.log(typeof str); // string
let obj = JSON.parse(str);
console.log(typeof obj); // object
obj.salis // Lietuva
str.salis // ERROR
```

# Grupinė užduotis

1. Esamą/dabartinį svetainės meniu perdaryti naudojant JSON objektus.
2. Aktyvus meniu elementas (šiuo metu atidarytas puslapis) turi būti išskirtinis dizaino prasme.

**BONUS:**

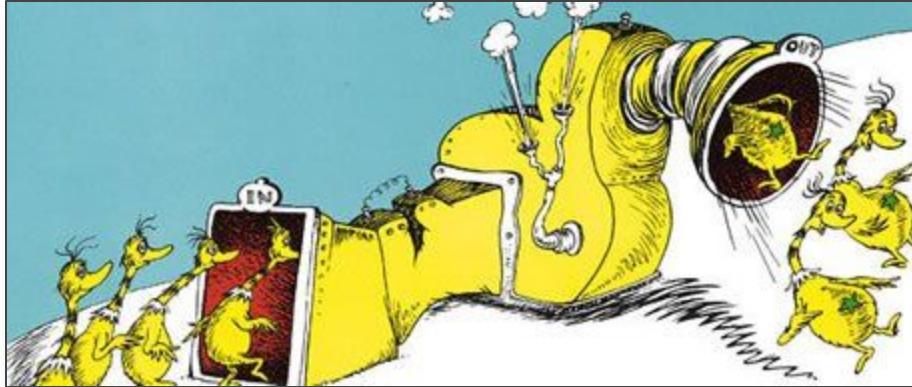
3. URL parametrus pakeisti taip kad jie neturėtų \*\*\*.html plėtinio

```
1 [
2   {
3     "title": "Pradžia",
4     "url": "/"
5   },
6   {
7     "title": "Portfolio",
8     "url": "/portfolio.html"
9   },
10 ...
11 ]
```

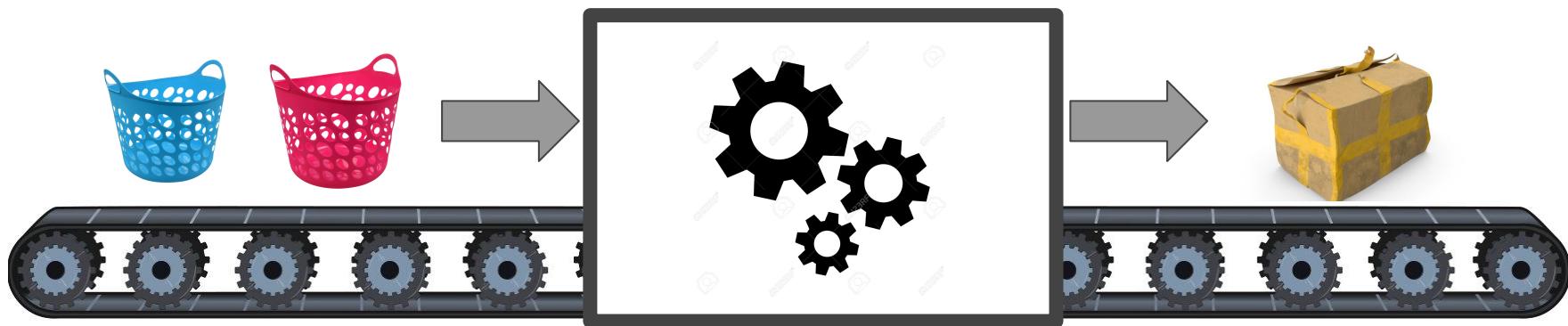
# Grupinės užduoties konцепcija

1. **Sukurkite HTML failą:** Sukurkite naują HTML failą, kuris bus pagrindinė jūsų svetainės puslapis.
2. **Įtraukite meniu konteinerį:** HTML faile sukurkite `<nav>` (navigacijos) žymę su unikaliu ID arba klase, kurioje bus talpinamas jūsų meniu.
3. **Sukurkite JSON objektą:** Sukurkite JSON tipo objektą, kuris saugos meniu elementų informaciją. Kiekvienas objekto turėtų turėti `href` ir `title` parametrus.
4. **Sukurkite funkciją:** Sukurkite JavaScript funkciją, kuri priims JSON objektą ir sugeneruos HTML meniu. Ši funkcija apdoros JSON objektą, sukurs meniu struktūrą ir sugeneruos HTML tekstą.
5. **Sąrašo generavimas:** Funkcija turi sukurti HTML sąrašo elementą (UL ar OL), į kurį bus įdėti meniu punktai (LI elementai).
6. **Eilutės generavimas:** Funkcija turi iteruoti per JSON objekto elementus ir kiekvienam elementui sukurti naują eilutę su nuoroda (A žymė) ir tekstu (turinys iš `title` parametro).
7. **Įdėkite generuotą HTML:** JavaScript funkcija turėtų įterpti sugeneruotą HTML tekstą į meniu konteinerį, kurį sukūrėte ankstesniame žingsnyje.
8. **Paleiskite scenarijų:** Pridėkite sukurtą JavaScript scenarijų į savo HTML failą, kad jis būtų vykdomas, kai puslapis įkeliamas.
9. **Peržiūrėkite rezultatą:** Atidarykite savo HTML failą naršyklėje ir patikrinkite, ar meniu buvo sėkmingai sugeneruotas su JSON objekto duomenimis.

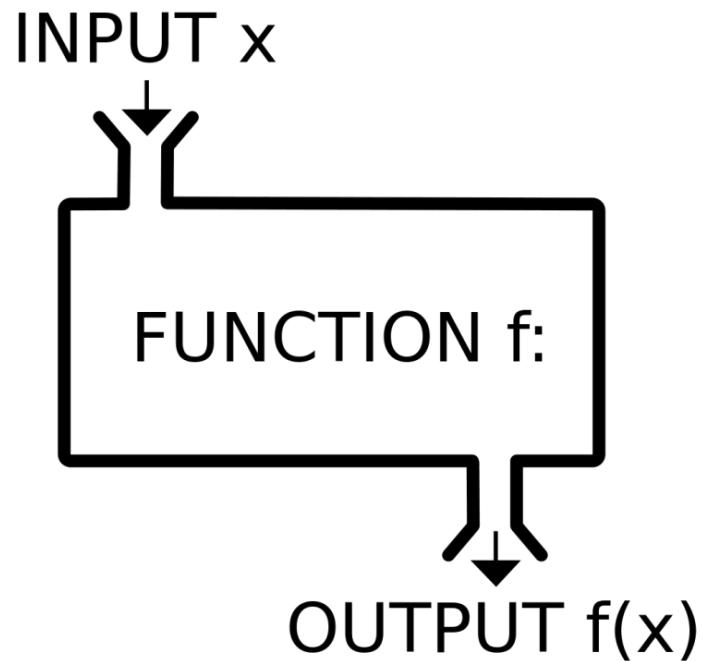
# Funkcija



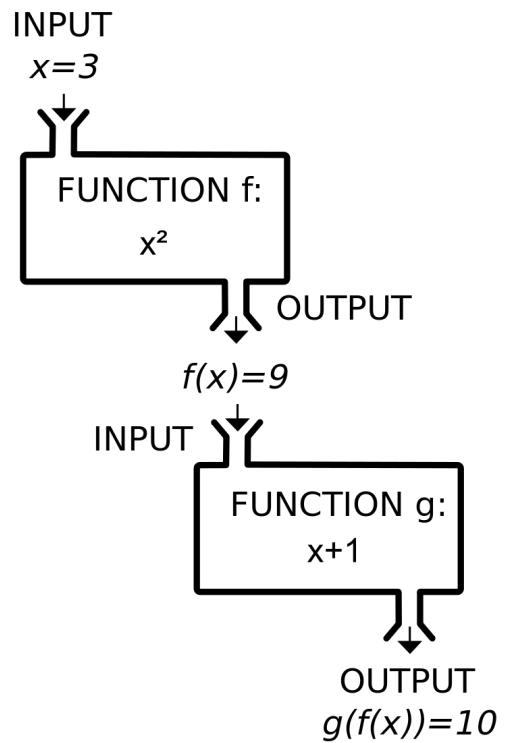
# Funkcija



# Funkcija



# Funkcija

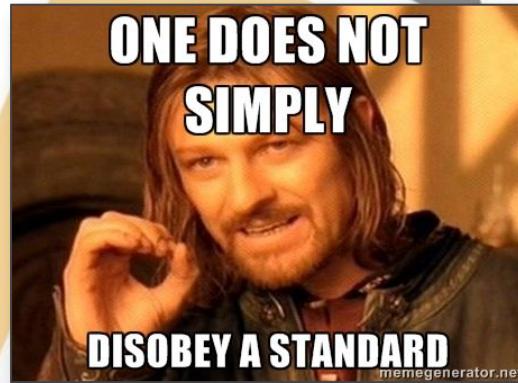


# Funkcija

```
function pavadinimas(parametras1, parametras2, ...parametrasN) {  
    // vykdomas kodas  
    return ...;  
}  
  
function pavadinimas(parametras1, parametras2) {  
    // vykdomas kodas  
}  
  
function pavadinimas() {  
    // vykdomas kodas  
    return rezultatas;  
}
```

# JavaScript kodo stilius

Tip nr.: 5



**JavaScript stiliaus gidai**  
[JavaScript Style Guide](#)  
arba

[Guidelines for writing JavaScript code examples - The MDN Web Docs project](#)

# Praktinė užduotis: Toggle



# Toggle algoritmas

```
let lemposBusena = false;
let lempa = document.getElementById('lempute');

function toggleLamp() {
    if (lemposBusena) {
        lempa.src = '/adresas/iki/off.jpg';
    } else {
        lempa.src = '/adresas/iki/on.jpg';
    }
    lemposBusena = !lemposBusena;
}
```

<https://codepen.io/tautvydas-dulskis/pen/PoNYjPj>

## lempa.html

```
<!DOCTYPE html>
<html>
...
<body>



</body>
</html>
```

# Užduotis

- Sukurti funkcijas sudetės, atimties, dalybos ir daugybos, kurioms bus perduodami du parametrai **a**, **b** ir rezultatas bus gražinamas per funkcijos return'ą.
- Panaudoti funkcijas kai kintamasis **a** keičiasi **a++** žingsniu, sukant ciklą **5** kartus. Atsakyma spausdinti elemente **#rezultatai**
- Panaudoti dar karta funkcijas (papildant sena koda) kai kintamasis **b** keičiasi **b += 2** žingsniu, sukant ciklą **10** kartų.

\* Rezultatai turi buti matomi tik kai paspausime ant <img> elemento

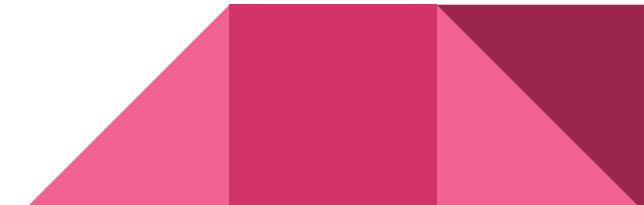
# Regex

Reguliari išraiška tai simbolių modelis.

Šablonas naudojamas teksto šablonų atitikimo funkcijoms „ieškoti ir pakeisti“.

„JavaScript“ sistemoje „RegExp“ objektas yra šablonas su savo savybėmis ir metodais.

Regex per 100 sec. : [Regular Expressions \(RegEx\) in 100 Seconds](#)



# Regex - match

```
let text = "Mr Blue has a blue house and a blue car";  
  
let result = text.match(/blue/gi);  
  
// ['Blue','blue', 'blue']
```

Funkcija **.match()** yra iškviečiama ant teksto eilutės ir priima reguliarujį išraiškos šabloną kaip argumentą. Ji ieško visų atitikmenų, kurie atitinka šį šabloną, ir grąžina juos masyvo pavidalu. Jei šablonas nenurodo jokių grupių, **.match()** grąžina masyvą, kuriame yra vienas elementas - originalus tekstas.

# Regex - replace

```
let text = "Mr Blue has a blue house and a blue car";  
  
let result = text.replace(/blue/gi, "red");  
  
// Mr red has a red house and a red car.
```

JavaScript funkcija **.replace()** naudojama pakeisti tekštą, kuris atitinka nurodytą reguliariają išraišką, kita eilute arba simbolių seka.

# Užduotis

## Raskite skaičius ir pakeiskite juos į kvadratus!

Turiu tekstą, kuriamo yra žodžių ir skaičių. Noriu rasti visus skaičius ir juos pakeisti į jų kvadratus.

Pavyzdys: Jei turime tekstą "Aš turiu 3 obuolius ir 2 slyvas.", mums reikia rasti skaičius (3 ir 2) ir juos pakeisti į jų kvadratus (9 ir 4). Galiausiai turėtume gauti tekstą "Aš turiu 9 obuolius ir 4 slyvas."

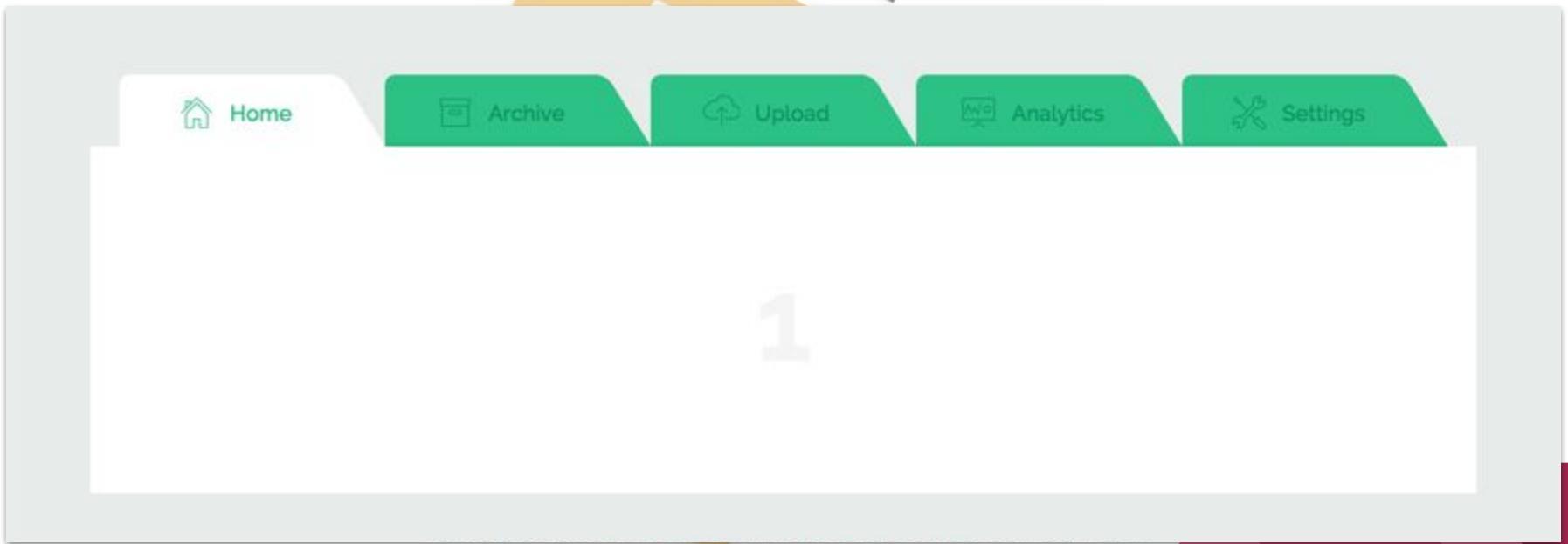
Norint tai padaryti, naudojame dvi funkcijas. Pirmoji funkcija yra `.match()`, kuri randa visus skaičius tekste. Ji veikia kaip žvejybos tinklas, ieškodama skaičių.

Antroji funkcija yra `.replace()`, kuri pakeičia skaičius į jų kvadratus. Tai kaip keisti mėlynas lapes į raudonas lapės.

Kai turime tekstą, kvadratuojame rastus skaičius ir pakeičiame juos tekste. Galiausiai gausime pakeistą tekštą su skaičiais pakeistais į jų kvadratus.

Todėl, jei matome tekstą "Aš turiu 3 obuolius ir 2 slyvas.", mums reikia panaudoti `.match()` funkciją, rasti skaičius (3 ir 2), ir pakeisti juos į jų kvadratus (9 ir 4). Galiausiai turėsime tekstą "Aš turiu 9 obuolius ir 4 slyvas."

# Tabs



# Tabai algoritmas

```
let sheets =
document.getElementsByClassName('sheet');

function changeTab (sheetId) {
  for (let i = 0; sheets.length > i; i++) {
    sheets[i].style.display ='none';
  }

  document.getElementById(sheetId)
    .style.display = 'block';
}
```

## tabai.html

```
...
<body>
<div class="tabs">
  <a href="#home" onclick="changeTab('home') ">HOME</a>
  <a href="#archive" onclick="changeTab('archive') ">ARCHIVE</a>
  <a href="#settings" onclick="changeTab('settings') ">
    SETTINGS
  </a>
  <a href="#other" onclick="changeTab('other') ">OTHER</a>
  <a href="#kita" onclick="changeTab('kita') ">KITA</a>
</div>

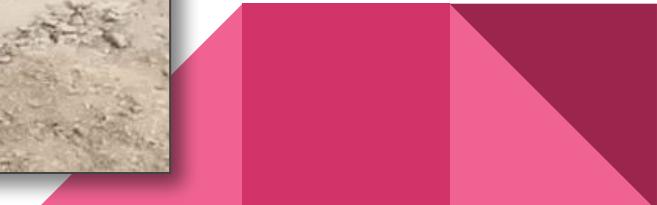
<div class="sheets">
  <div id="home" class="sheet">HOME DATA</div>
  <div id="archive" class="sheet">ARCHIVE DATA</div>
  <div id="settings" class="sheet">SETTINGS DATA</div>
  <div id="other" class="sheet">OTHER DATA</div>
  <div id="kita" class="sheet">KITA DATA</div>
</div>
</body>

...
```

<https://codepen.io/tautvydas-dulskis/pen/wvGweop>

# jQuery Tutorial





# Jquery tabai algoritmas

```
$('.tabai > a').click(function(e) {
    pakeistiTaba($('.this').attr('href'));
    e.preventDefault();
});

function pakeistiTaba (id) {
    $('.sheets > div').hide();
    $(id).show();
}
```

## tabai.html

```
<!DOCTYPE html>
<html>
<body>
<div class="tabai">
    <a href="#home" class="active">HOME</a>
    <a href="#archive">ARCHIVE</a>
    <a href="#settings">SETTINGS</a>
    <a href="#other">OTHER</a>
    <a href="#kita">KITA</a>
</div>
<div class="sheets">
    <div id="home">HOME DATA</div>
    <div id="archive">ARCHIVE DATA</div>
    <div id="settings">SETTINGS DATA</div>
    <div id="other">OTHER DATA</div>
    <div id="kita">KITA DATA</div>
</div>
</body>
</html>
```

<https://codepen.io/tautvydas-dulskis/pen/mdPbwWm>

# Papildoma 'active' klasė

```
$('.tabai > a').click(function(e) {
    pakeistiTaba($(this).attr('href'));
    priskirtiKlase('active', this);
    e.preventDefault();
});

function pakeistiTaba (id) {
    $('.sheets > div').hide();
    $(id).show();
}

function priskirtiKlase(klasesvardas, el){
    $('.tabai > a').removeClass(klasesvardas);
    $(el).addClass(klasesvardas);
}
```

## tabai.html

```
<!DOCTYPE html>
<html>
<body>
<div class="tabai">
    <a href="#home" class="active">HOME</a>
    <a href="#archive">ARCHIVE</a>
    <a href="#settings">SETTINGS</a>
    <a href="#other">OTHER</a>
    <a href="#kita">KITA</a>
</div>
<div class="sheets">
    <div id="home">HOME DATA</div>
    <div id="archive">ARCHIVE DATA</div>
    <div id="settings">SETTINGS DATA</div>
    <div id="other">OTHER DATA</div>
    <div id="kita">KITA DATA</div>
</div>
</body>
</html>
```

<https://codepen.io/tautvydas-dulskis/pen/rNeBwYX>

# Failo nuskaitymas



[Load JSON file locally using pure Javascript by Rich on CodePen](#)

# Formos saugojimas i DB neperkraunant psl.

```
$('document').ready(function () {
    $('#btnSubmit').click(function (event) {
        event.preventDefault();
        let form = $('#manoForma')[0];
        let data = new FormData(form);
        $('#btnSubmit').prop('disabled', true);
        $.ajax({
            type: 'POST',
            enctype: 'multipart/form-data',
            url: 'saugoti.php',
            data: data,
            processData: false,
            contentType: false,
            cache: false,
            timeout: 600000,
            success: function (data) {
                $('#result').html(data);
                console.log('SUCCESS : ', data);
                $('#btnSubmit').prop('disabled', false);
            },
            error: function (e) {
                $('#result').html(e.responseText);
                console.log('ERROR : ', e);
                $('#btnSubmit').prop('disabled', false);
            }
        });
    });
});
```

```
<form id="manoForma">
    <input name="vardas" placeholder="Koks jūsų vardas?" /><br/>
    <input name="elpastas" placeholder="Koks jusu el. paštas?" /><br/>
    <textarea name="zinute" placeholder="Čia žinute man"></textarea>
    <input type="button" id="btnSubmit" value="Klausti">
</form>
<div id="result"></div>
```

```
<?php
$dbName = 'manoProjektas';
$user = 'root';
$password = '';
$vardas = $_REQUEST['vardas'];
$elpastas = $_REQUEST['elpastas'];
$zinute = $_REQUEST['zinute'];
try {
    $db = new
PDO('mysql:host=localhost;dbname='.$dbName.';charset=utf8mb4', $user,
$password);
    $db->exec("INSERT INTO zinutes(vardas, elpastas, zinute)
VALUES('".$vardas."', '".$elpastas."', '".$zinute."')");
} catch(PDOException $ex) {
    echo 'Kilo klaida! ' . $ex->getMessage();
}
```

# Kas yra Fetch API?



**Fetch API** yra moderni technologija, kuri leidžia programuotojams "prašyti" ir gauti duomenis iš interneto naudojant JavaScript kodą. Tai yra būdas, kaip programa gali komuniuoti su interneto serveriais ir gauti informaciją, pavyzdžiui, tekstą, nuotraukas arba kitus failus.

KITAIP TARIANT

**Fetch API** yra taip vadinama technologija, kuri leidžia mums "prašyti" informacijos iš interneto. Panašiai kaip prašome mamos paskui vakarienės, kad gautume gardų maistą, programuotojai naudoja Fetch API norėdami gauti duomenis iš svetainių.

# Fetch API veikia ?



Panašiai kaip norint gauti informaciją iš interneto serverio naudojant naršyklę ar programą.

1. Programuotojas sukuria užklausos (**request**) objektą, kuriame nurodoma, kokią informaciją jis nori gauti ir iš kurio serverio.
2. Užklausos objektas tada yra perduodamas **Fetch API**, kuris **atlieka užklausą** ir pradeda bendrauti su nurodytu interneto serveriu.
3. Interneto serveris **gauna užklausą** ir apdoroja ją. Serveris grąžina duomenis, kurie yra įdėti į atsakymo (**response**) objektą.
4. Atsakymo **objektas** yra perduodamas atgal į **Fetch API**, kur programuotojas gali pasiekti duomenis, kuriuos serveris grąžino.
5. Programuotojas tada **gali atlikti norimus veiksmus** su gautais duomenimis, tokius kaip jų atvaizdavimas interneto puslapyje, analizė arba kitos manipuliacijos.

Šis procesas yra asinchroninis, tai reiškia, kad programa nepristabdo savo vykdymo laukdama atsakymo iš serverio. Tai leidžia naršyklei ar programai tęsti darbą, kol gauna atsakymą iš serverio.

# Fetch API veikia ? (Penkiamečiams :P )

Įsivaizduokime, kad esi mažas detektyvas ir nori sužinoti, ką tavo draugas žaidžia savo kambaryje. Tačiau, jis yra kitame kambaryje ir negali tau tiesiogiai pasakyti. **Taigi, kaip gauti šią informaciją?**

1. Tu užrašai ant lapo savo klausimą - "Ką tu žaidi savo kambaryje?" (Tai yra tavo **užklausa**)
2. Tu paduodi savo lapą savo mamai arba tėčiui (tai yra kaip tavo programa siunčia užklausą į **Fetch API**).
3. Tavo mama ar tėtis (**Fetch API**) pažiūri į tavo užklausą ir eina pas draugą (**interneto serverį**) ieškoti atsakymo.
4. Jie grįžta atgal pas tave su lapeliu, ant kurio parašyta "**Aš žaidžiu su Lego konstruktoriais**" (tai yra **atsakymas** iš interneto serverio).
5. Dabar tu žinai, ką tavo draugas veikia savo kambaryje! (Tai yra informacija, kurią gavai iš **Fetch API**).

Taigi, tavo **užklausa** yra kaip lapas su klausimu, **Fetch API** yra tavo mama ar tėtis, kuris pasiima užklausą ir perduoda ją draugui (**interneto serveriui**). Atsakymas iš draugo yra kaip lapas su informacija, kurį tu gauni atgal per **Fetch API**.

# Trūkumai



**Naršyklių palaikymas:** Nors dauguma modernių naršyklių palaiko Fetch API, senesnės naršyklos gali jo nepalaikyti arba palaikyti nepilnai. Tai reiškia, kad programuotojai turės įsitikinti, kad jų kodas veikia tinkamai visose naršyklose.

**Kryžminis tinklalapių užklausų apribojimas:** Saugumo sumetimais daugelis naršyklių apriboja kryžmines užklausas (**Cross-Origin Resource Sharing - CORS**). Tai reiškia, kad, norint atlikti užklausą į kitą interneto svetainę, ta svetainė turi būti konfigūruota leisti tokią užklausą. Kitaip tariant, negalėsite tiesiogiai pasiekti bet kokios svetainės duomenų.

**Saugumo rizika:** Fetch API gali būti naudojamas pažeisti saugumo taisykles. Pvz., jei svetainė leidžia siųsti užklausas kitoms svetainėms ir tam nepanaudoja tinkamų apribojimų, tai gali padidinti galimybę kenksmingai išnaudoti interneto svetainę.

**Klaidų valdymas:** Kai kurie klaidų tipai gali būti sudėtingi atskirti ir suprasti, ypač pradedantiesiems programuotojams. Tai gali sukelti neaiškumų, jei programa neteisingai apdoroja atsakymus arba susiduria su problemomis, komunikujant su interneto serveriais.

**Asinchroninis pobūdis:** Kartais asinchroninis kodas gali būti painus suprasti ir sudėtingesnis valdyti. Jis reikalauja tinkamo klaidų valdymo ir kodo organizavimo, kad būtų išvengta nenumatytyų rezultatų.

# Kodas



## Kodo eilučių paaiškinimas:

```
1 // Adresas, iš kurio norime gauti duomenis
2 const apiUrl = 'https://jsonplaceholder.typicode.com/users/1'
3
4 // Atliksim GET užklausą į nurodytą adresą
5 fetch(apiUrl)
6   .then(response => response.json()) // pavertimui į JSON formatą
7   .then(data => {
8     // Čia mes gausime atsakymo duomenis
9     console.log(data)
10  })
11 .catch((error) => {
12   // Jei įvyksta klaida, ją apdorojame čia
13   console.error('Įvyko klaida:', error)
14 })
15
```

**const apiUrl = 'https://jsonplaceholder.typicode.com/users/1';**: Sukuriame konstantą, kuri nurodo interneto adresą (URL), iš kurio norime gauti duomenis. Šiuo atveju, mes norime gauti duomenis apie vartotoją iš API.

**fetch(apiUrl):** Šis kodas pradeda GET užklausą į nurodytą interneto adresą. Fetch API grąžina pažadą (Promise) - tai yra atsakymo į užklausą pažadas.

**.then(response => response.json()):** Su .then() metodu apdorojame gautą pažadą. Kai užklausa yra sėkminga, response.json() paverčia atsakymo duomenis į JSON formatą.

**.then(data => { console.log(data); }):** Kai duomenys yra paverčiami į JSON, juos išvedame į konsolę, kad galėtume pamatyti, ką gavome.

**.catch(error => { console.error('Įvyko klaida:', error); }):** Jei užklausa nepavyksta, klaidą apdorojame šiame bloke.

# Kodas

```
1 // Adresas, į kurį norime siųsti duomenis
2 const apiUrl = 'https://jsonplaceholder.typicode.com/users';
3
4 // Duomenys, kuriuos siūsime (pavyzdys)
5 const userData = {
6   name: 'Jonas',
7   email: 'jonas@example.com',
8 };
9
10 // Atliksim POST užklausą į nurodytą adresą su duomenimis
11 fetch(apiUrl, {
12   method: 'POST',
13   headers: {
14     'Content-Type': 'application/json',
15   },
16   body: JSON.stringify(userData),
17 })
18   .then(response => response.json()) // pavertimui į JSON formatą
19   .then(data => {
20     // Čia mes gausime atsakymo duomenis (pavyzdžiu, ID priskirtą naujam vartotojui)
21     console.log(data);
22   })
23   .catch(error => {
24     // Jei įvyksta klaida, ją apdorojame čia
25     console.error('Įvyko klaida:', error);
26   });
27});
```

## Kodo eilučių paaškinimas:

**const apiUrl = 'https://jsonplaceholder.typicode.com/users';**: Sukuriame konstantą, kuri nurodo interneto adresą (URL), į kurį norime siųsti duomenis. Šiuo atveju, mes norime siųsti duomenis apie naują vartotoją į API.

**const userData = { name: 'Jonas', email: 'jonas@example.com' };**: Sukuriame objektą su duomenimis, kuriuos siūsime į serverį. Tai gali būti, pavyzdžiu, duomenys apie naują vartotoją.

**fetch(apiUrl, { method: 'POST', headers: { 'Content-Type': 'application/json', }, body: JSON.stringify(userData), })**: Su fetch() funkcija pradedame POST užklausą į nurodytą interneto adresą. Naudojame method: 'POST', kad nurodytume, jog tai yra POST užklausa. headers nustatome, kad siunčiame duomenis JSON formato, o body su JSON.stringify(userData) pavercia userData objekta į JSON formatą ir siunčia į serverį.

**.then(response => response.json()):** Su .then() metodu apdorojame gautą pažadą. Kai užklausa yra sėkminga, response.json() pavercia atsakymo duomenis į JSON formatą.

**.then(data => { console.log(data); })**: Kai duomenys yra pavertiami į JSON, juos išvedame į konsolę, kad galėtume pamatyti, ką gavome (pavyzdžiu, naujo vartotojo ID).

**.catch(error => { console.error('Įvyko klaida:', error); })**: Jei užklausa nepavyksta, klaidą apdorojame šiame bloke.

Šis kodas atliks POST užklausą į nurodytą interneto adresą ir sius **userData** objekta su naujo vartotojo duomenimis.

# Top 85 JavaScript Interview Q&A

Top 85 JavaScript Interview Questions & Answers

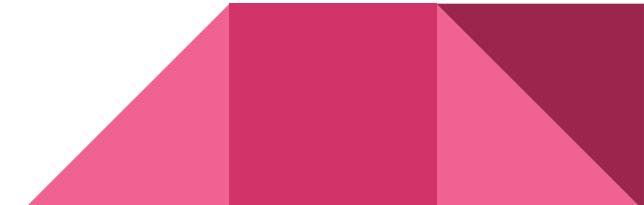
# Rikiavimo algoritmai



JS



CS-Playground-React



# Paieškos algoritmai rikiuotame masyve

**Binary search**      steps: 0

The diagram illustrates the binary search algorithm on a sorted array of 16 integers. The array elements are: 1, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47, 53, 59. A target value, 37, is highlighted in orange at the top left. The search process is shown with three pointers: 'Low' (0), 'mid' (8), and 'high' (16). The 'mid' pointer is currently pointing to the element 23.

**Sequential search**      steps: 0

The diagram illustrates the sequential search algorithm on a sorted array of 16 integers. The array elements are: 1, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47, 53, 59. A target value, 37, is highlighted in orange at the top left. The search process is shown with an orange arrow pointing sequentially through each element of the array.

[www.penjee.com](http://www.penjee.com)

# The best front-end hacking cheatsheets



A practical guide to learning front end development  
for beginners



# PHP - PDO Tutorial for MySQL Developers

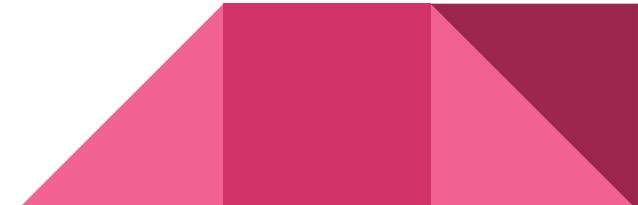
JS



PDO Tutorial for MySQL Developers

# To Designers With Love

## (A Letter From a Front-end Developer)

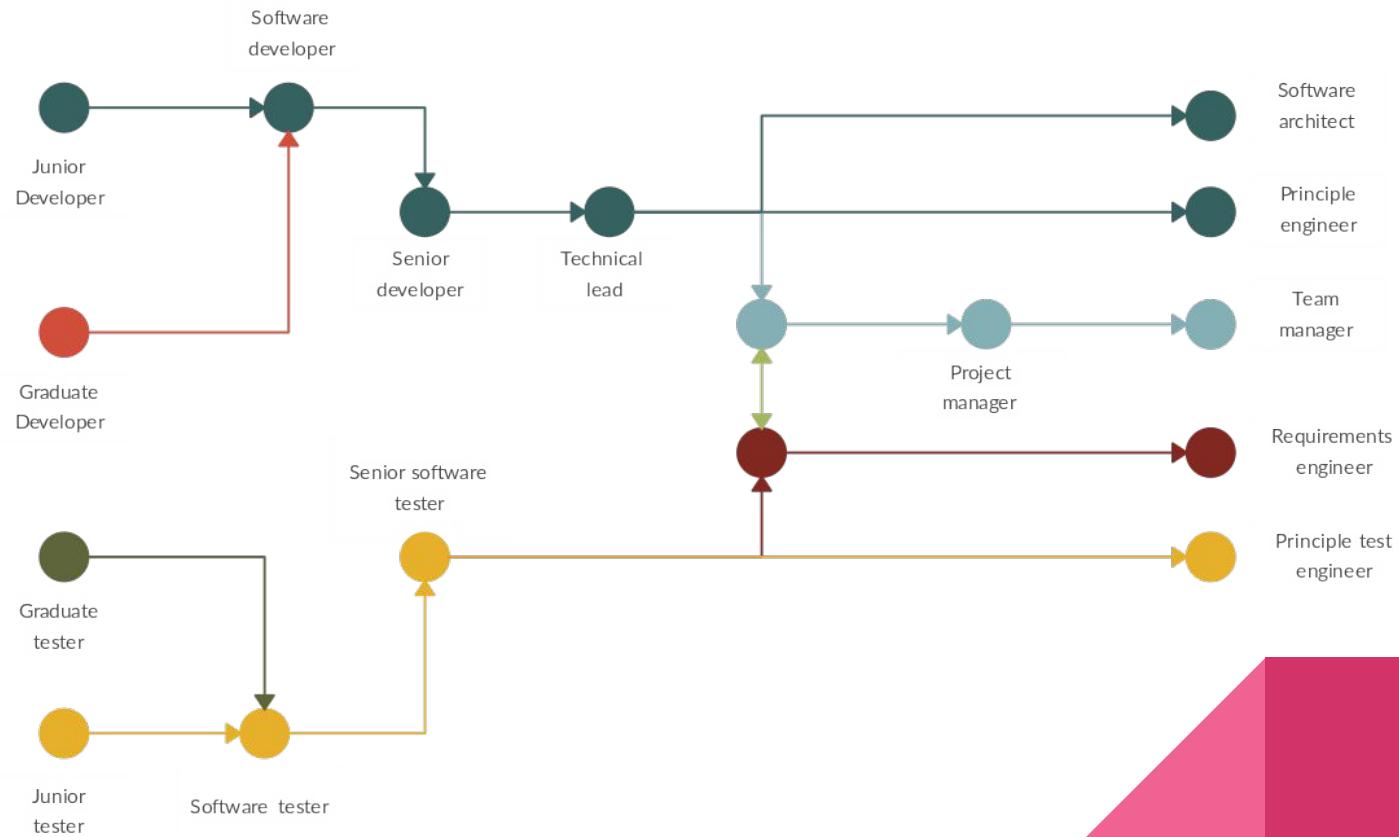


# What's next ?



<https://roadmap.sh/frontend>

## CAREER PATH OF A SOFTWARE ENGINEER/DEVELOPER at ABC Co.



THE END ...

