

1 Seminaras

Justas Mundeikis*

2019 m. vasario 20 d.

1 Programų instaliavimas

Instaliuokite šias programas savo asmeniniame kompiuteryje. Jeigu naudojotės VU kompiuteriais, gali prireikti instaliuoti nešiojamą (portable) versiją Sublime.

1. Instaliuokite R
2. Instaliuokite RStudio
3. Instaliuokite GitBash
4. Instaliuokite Sublime
5. Nepamirškite pagrindinių nustatymų

```
1 $ git config --global user.name "Vardas èPavard"
2 $ git config --global user.email vardas.pavarde@evaf.vu.lt
3 $ git config --global core.pager cat
```

2 Paskyrų susikūrimas

1. Susikurkite savo paskyrą, naudodami bet kokią pseudonimą @Github, tačiau naudodami savo universitetinį email vardas.pavarde@stud.vu.lt

3 Nuosavo tinklapio sukūrimo projektas

Šioje dalyje naudosimės Git, Sublime ir Github, jog sukurti savo nuosavą internetinį puslapį. Pastaba: Tarkime, kad mano studento nr yra "S175", Jūsų gali būti kitoks, tada naudojate savo vietoj mano "S175"

1. Su Git nueiname and Desktopo (arba tiesiog paleidžiama "Git Bash Here" [su dešiniu pelės mygtuku])
2. Sukuriame folderį webpage `mkdir -p S175/webpage`
3. Perėjus į patį webpage folderį, inicijuojame jį su git, taigi prima `cd S175/webpage`, o tada `git init`
4. Pirmas dalykas, kurį mums reiktų sukurti, tai bazinis html dokumentas kurio pavadinimas būtų *index* `touch index.html`
5. Jeigu tiesiog atsidarysite index.html failą su kuria nors naršykle, pamatysite, kad viskas yra absoliučiai balta! Neįdomu :D

*justas.mundeikis@evaf.vu.lt

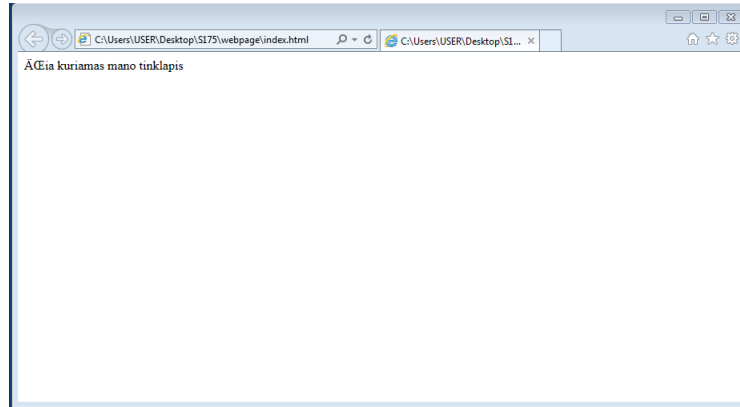
6. Atsidarome *index.html* su *Sublime* editoriumi

7. Editoriuję įrašome tekstą

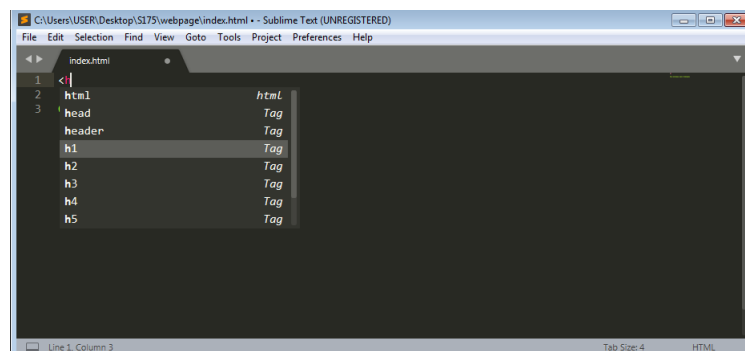
```
1 Č
2 ia kuriamas mano tinklapis
```

išsaugome (ctrl+s)

8. Atnaujiname interneto naršyklę (F5), rezultatas turėtų atrodyti daugmaž taip

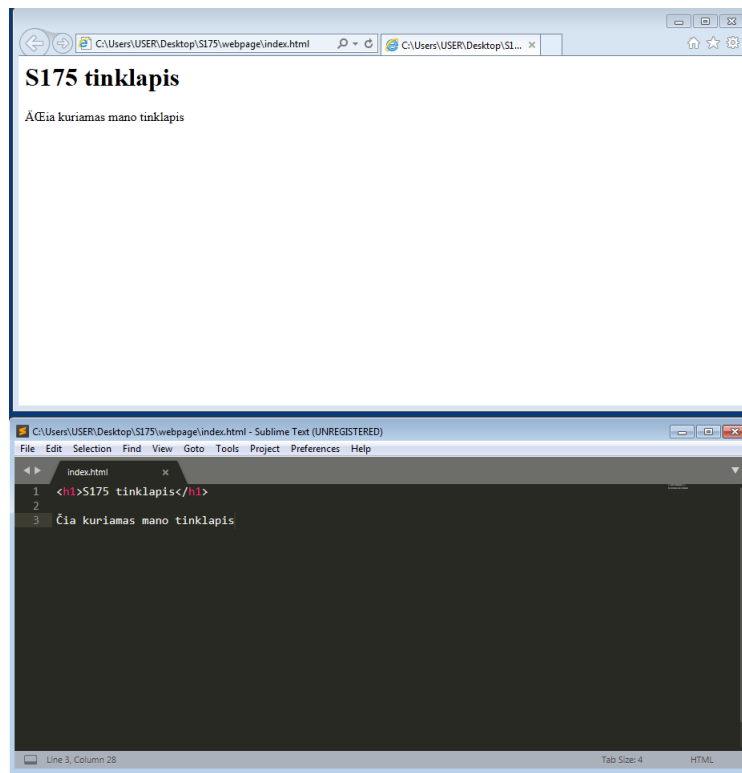


9. Tam kad dokumentas turėtų antraštę, į pirmą eilutę parašom `<h1>SI75 tinklapis </h1>`. HTML kode `<h1>` pažymį pimro lygio antraštės *tag*. Tiesa, *Sublime* yra pakankamai protingas ir pradėjus rašyti `<h` išmeta pasirinkimą, pasirinkus `h1` pats supildo viską, bereikia įrašyti tik savo tekstą



10. Išsaugokite pakeitimus editoriuję ir atnaujinkite naršyklę

11. `git add . && git commit -m "Sukuriamas index.html failas"`



12. Dabar prikurkime šiek tiek teksto mūsų tinklapiui pvz., "Šio tinklapio kūrimas yra 1.Seminaro darbas" ir išsaugome editoriaus pakeitimus

```
1 <h1>S175 tinklapis</h1>Č
2
3 ia kuriamas mano tinklapisŠ
4
5 io tinklapio ūkrimas yra 1.Seminaro darbas
```

13. Pažiūrime ką rodo `git status`

```
1 USER@PC MINGW64 ~/Desktop/s175/webpage (master)
2 $ git status
3 On branch master
4 Changes not staged for commit:
5   (use "git add <file>..." to update what will be committed)
6   (use "git checkout -- <file>..." to discard changes in working directory)
7
8       modified:   index.html
```

14. Pažiūrime ką rodo `git diff`, kuris parodo skirtumus tarp failų

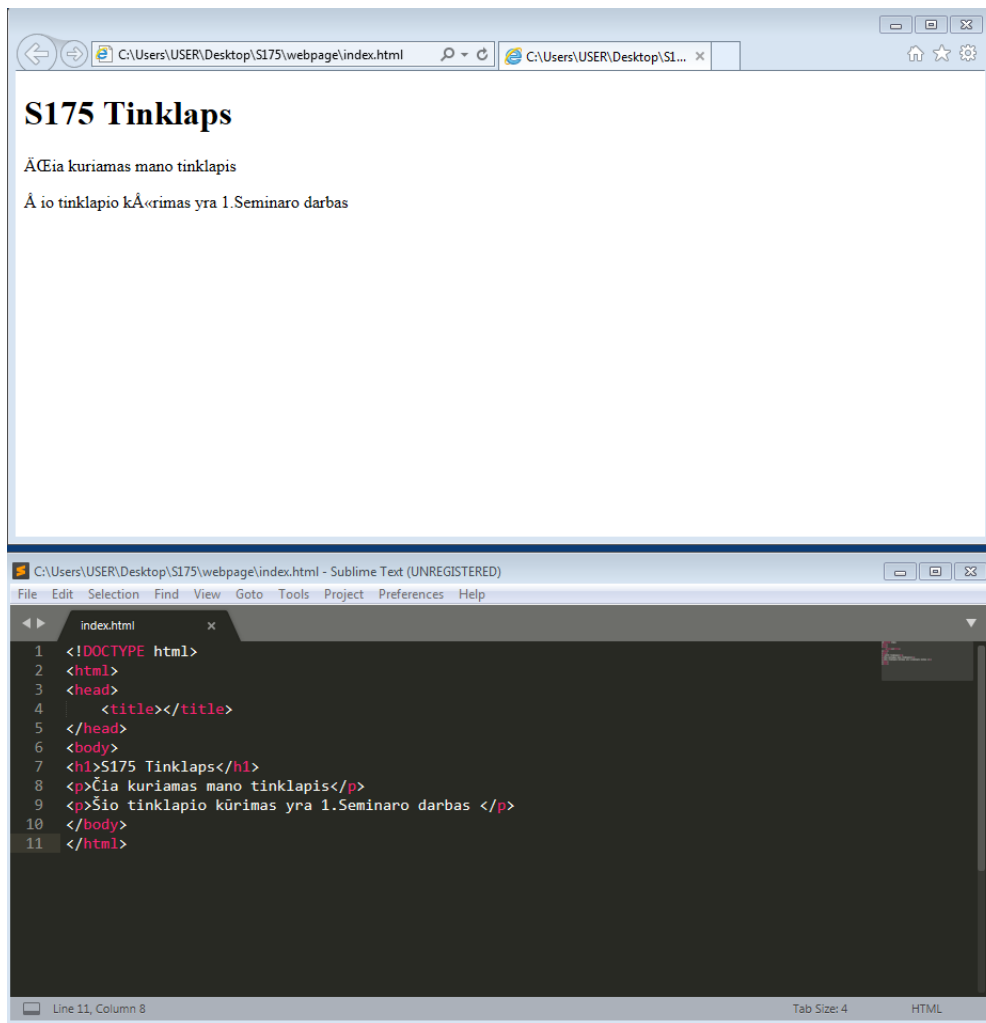
```
USER@PC MINGW64 ~/Desktop/s175/webpage (master)
$ git diff
diff --git a/index.html b/index.html
index 4d8137a..47c1b4b 100644
--- a/index.html
+++ b/index.html
@@ -1,3 +1,5 @@
<h1>S175 tinklapis</h1>
-Čia kuriamas mano tinklapis
\ No newline at end of file
+Čia kuriamas mano tinklapis
+Šio tinklapio kūrimas yra 1.Seminaro darbas
\ No newline at end of file
USER@PC MINGW64 ~/Desktop/s175/webpage (master)
$
```

15. Kadangi pakeitimai teisingi, galima `git commit -am "Sukurta index.html šėantrat."` Kadangi jau trackinam visus failus tai galima naudoti sutrumpintą komandą, kur -am reiškia -all ir -m

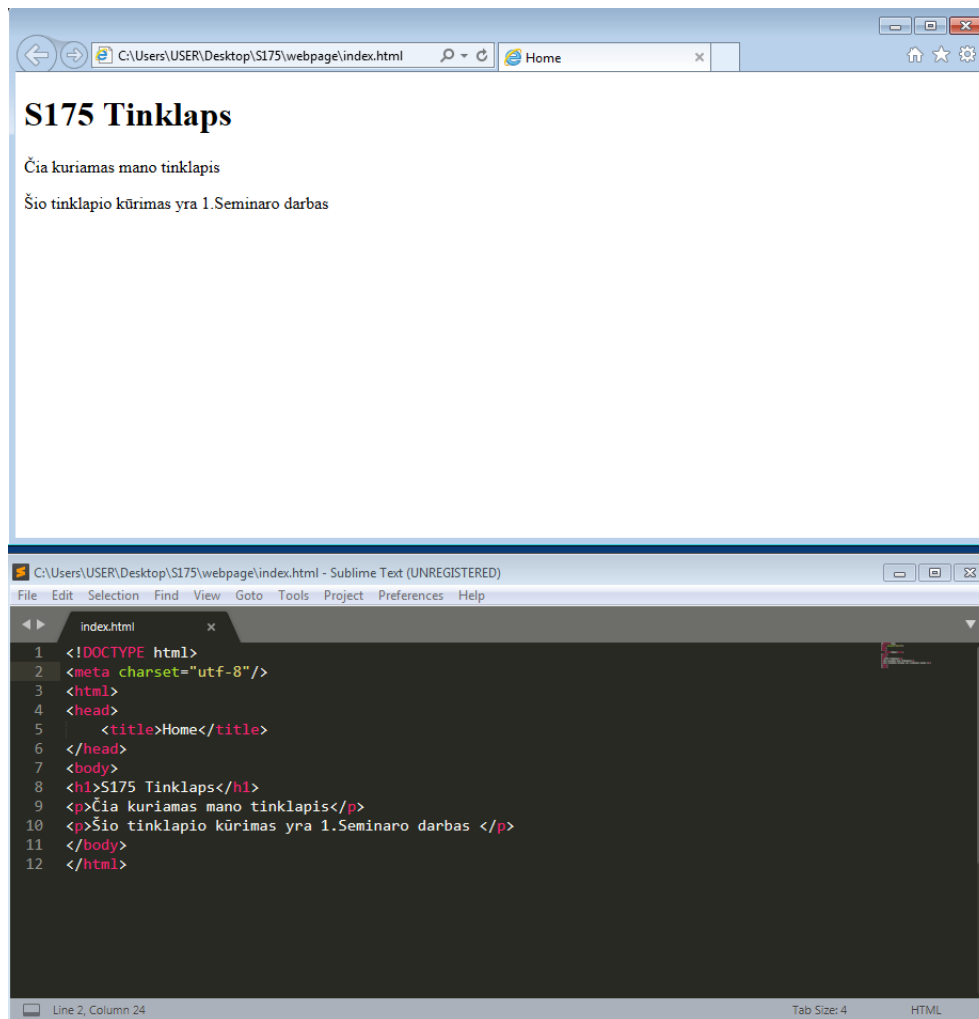
16. Na bet į internetinį puslapį toli gražu tai dar nepanašu... Todėl į editorių perrašome šį bloką" (arba pradėdame typinti <h ir pasirenkame html ir užpildome trūkstama informacija)

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <title>Home</title>
5 </head>
6 <body>
7   <h1>S175 Tinklapis</h1>
8   <p>Čia kuriamas mano tinklapis</p>
9   <p>Šio tinklapio ūkimas yra 1.Seminaro darbas </p>
10 </body>
11 </html>
```

- `<!DOCTYPE html>` nusako koks tai dokumento tipas, šiuo atveju tai html dokumentas
- `<html>` atidaro html tag
- `<head>` atidaro head tag
- `<title></title>` atidaro ir uždaro title tag
- `</head>` uždaro head tag
- `<body>` atidaro body tag (t.y. puslapio turinys)
- `<h1></h1>` atidaro ir uždaro antraštės tipo tag
- `<p></p>` atidaro ir uždaro paragrafo tag
- `</body>` uždaro body tag
- `</html>` uždaro html tag



17. Matyti, jog naršyklėje esantis Tab rodo ne visai tai, ką norėtume matyti, pvz., "Home", todėl turime editoriaus kodą 4 eilutėje pakeist į `<title>Home</title>`
18. Dar vienas nemalonus dalykas, naršyklė nesupranta lietuviškų simbolių, todėl įterpiame po pirmos eilutės: `<meta charset="utf-8"/>`, taigi priskiriame meta informaciją puslapiui ir pasakome, jog naršyklė skaitydama šį tekstą naudotų utf-8 kodavimą, kuris turi ir lietuviškus simbolius
19. Išsaugome editoriaus pakeitimus, atnaujiname naršyklę ir commitinam pakeitimus `git commit -am "papildytas i`
Dabartinis tinklapis jau daug maž panašus į tinklapį.



20. Github sukuriame naują repo "webpage", Praleidžiame aprašymą ir paliekame be "README".

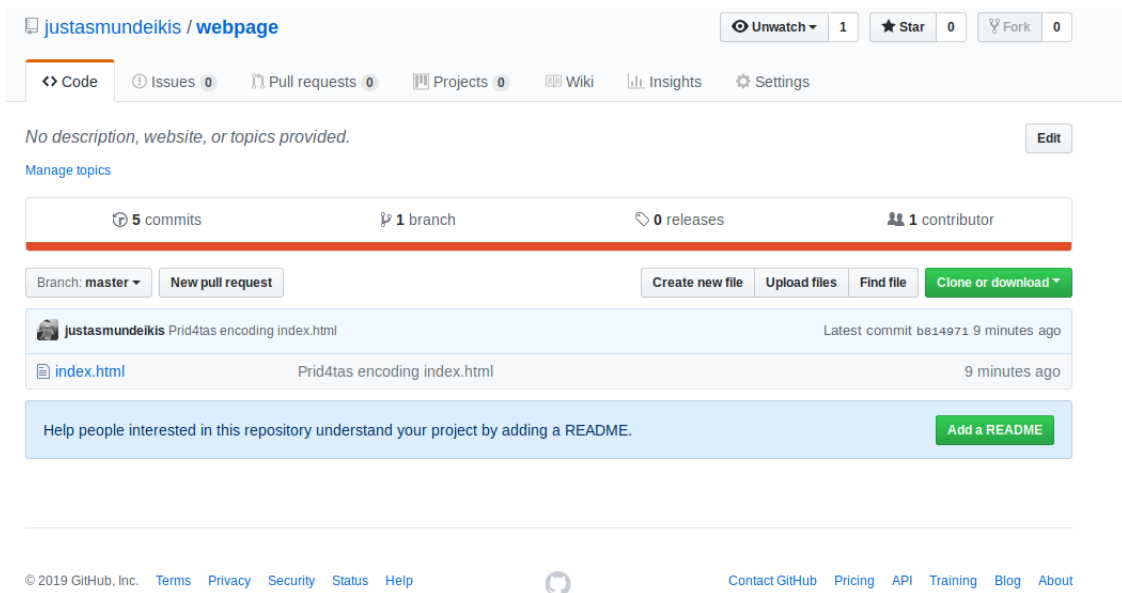
21. Keliaujame į Git Bash ir įrašome

```
git remote add origin https://github.com/vartotojovardas/webpage.git".
```

tada shipinam savo darbą į internetą:

```
git push -u origin master".
```

-u reiškia, jog vietinė repo išmoksta, kad *Github* yra *upstream repository*, o tai leis ateityje tiesiog parsisiųsti naudojant *pull* komandą. Ataujiname Github psl ir pamatysim



22. Kadangi nėra blogai turėti README failą, tai Git Bash sukuriame jį.

```
touch README.md" .
```

o tada atsidarome failą su *Sublime* ir perrašome šį Markdown formatuotą tekstą:

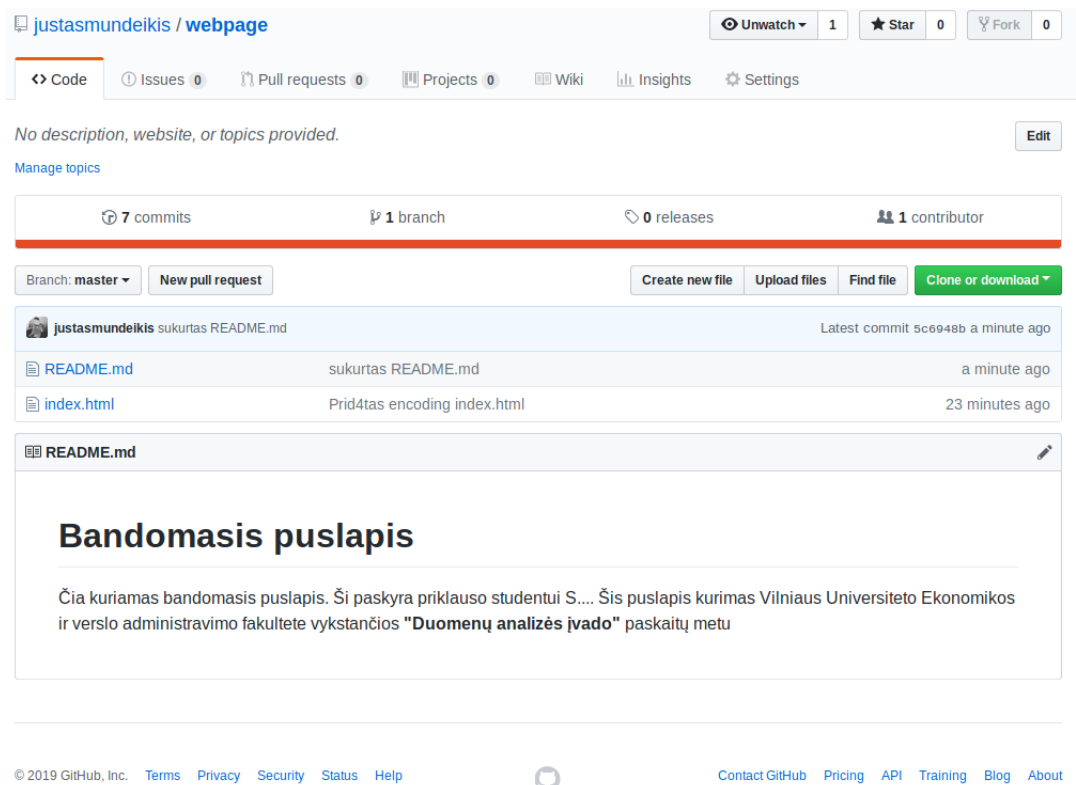
```
1 # Bandomasis puslapisČ
2
3 ia kuriamas bandomasis puslapis. Ši paskyra priklauso studentui S....Š
4 is puslapis kurimas Vilniaus Universiteto Ekonomikos ir verslo administravimo
   ↳ fakultete čvykstanios **"ųDuomenė analizė įvado"** ūpaskait metu
```

23. dabar reikia stag'inti ir commit'inti README.md fail, bei ship'inti jį į Github

```
git add . && git commit -m "sukurtas README.md" bei
```

```
git push . Kadangi jau pasakėme kas yra upstream repository, galime trumpinti komandą.
```

24. Atnaujinius GitHub puslapį, matome:



25. Čia galima rasti daugiau markdown formatavimo subtilybių: [Markdown cheatsheet link](#)

26. Kadangi puslapis yra gana nykus, pabandykim įdėti kokį linksną paveiksluką

27. Git Bash sukuriamo naują aplanką `mkdir images`

28. Git Bash yra stiprus tuo, jog gali netgi atsisiųsti failus iš interneto! Tam naudojame šią komandą:

```
curl -o images/github.png -OL https://upload.wikimedia.org/wikipedia/commons/thumb/b/b3/GitHub.svg/567px-GitHub.svg.png
```

Folderyje images turėjo atsirasti šis paveikslukas



29. Dabar reikia dar jį įterpti į `index.html` failą. Tai galime padaryti html body įrašę tokią eilutę ``. Kadangi paveikslukas yra `images` tai įrašome atitinkamai `src=""` (source). Na bet kad būtų linksmiau, pridėdame dar ir vieną paveiksluką iš interneto

```
1 <!DOCTYPE html>
2 <meta charset="utf-8" />
3 <html>
4 <head>
5   <title>Home</title>
6 </head>
7 <body>
8 <h1>S175 Tinklapis</h1>
9 <p>Čia kuriamas mano tinklapis</p>
10 <p>Šio tinklapio ūkrimas yra 1.Seminaro darbas </p>
11 <p></p>
12 <p>
13 </p>
```

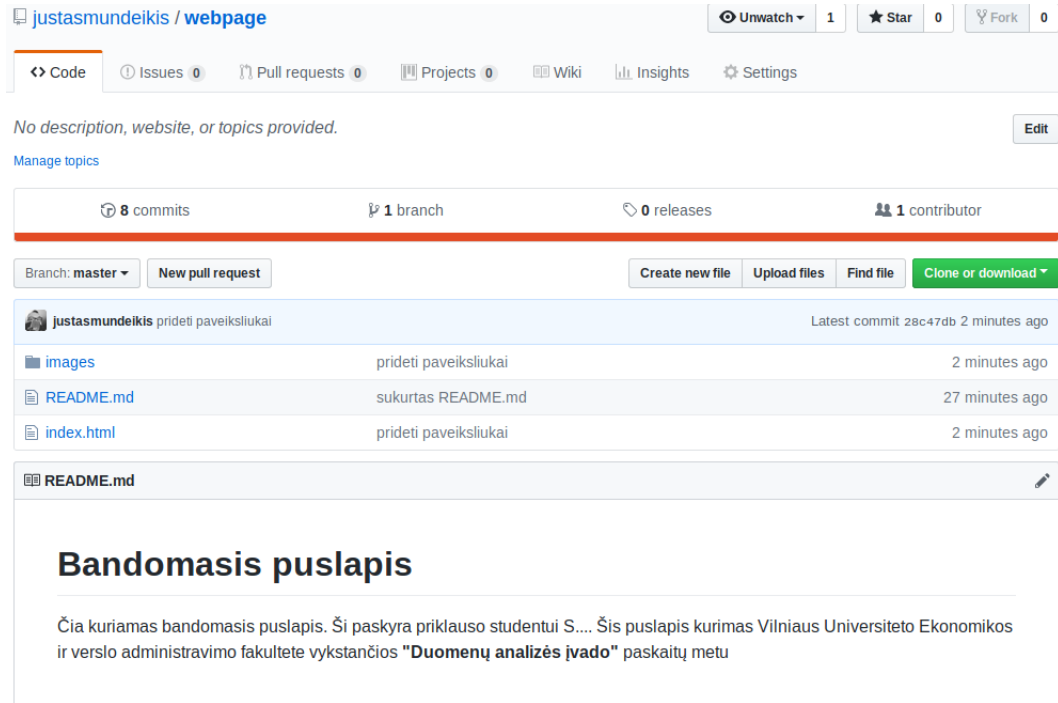


```
14 </body>
15 </html>
```

```
git add . && git commit -m "ėpridti paveiksliukai" bei
```

```
git push
```

Ataujinus Github puslapį, matome, jog atsirado folderis "images"



30. GitBash sukuriamo naują atšaką ir einame į ją `git checkout -b about-puslapis` [-b sukuria atšaką]. Alternatyviai `git brancha about-puslapis` ir `git checkout about-puslapis`

31. Nukopijuojame index.html į failą pavadinimu about.html naudodami komandą `cp index.html about.html`

32. Atsidarome about.html su Sublime ir pakeičiame turinį, supildant savo duomenis

```
1 <!DOCTYPE html>
2 <meta charset="utf-8" />
3 <html>
4 <head>
5   <title>About</title>
6 </head>
7 <body>
8 <p><strong>Vardas:</strong> Vardas ėPavard</p>
9 <p><strong>Stud. nr. :</strong> S175</p>
10 <p><strong>Kursas</strong>: Ekonomika</p>
11 <p><strong>ėGrup:</strong> xx</p>
12 </body>
13 </html>
```

33. Kadangi norime, jog ir pagrindiniame puslapyje būtų nuorodą į puslapį apie mus, turime pakeisti index.html į

```
1 <!DOCTYPE html>
2 <meta charset="utf-8" />
3 <html>
4 <head>
5   <title>Home</title>
6 </head>
7 <body>
8 <h1>S175 Tinklapis</h1>
9 <p>Čia kuriamas mano tinklapis</p>
```

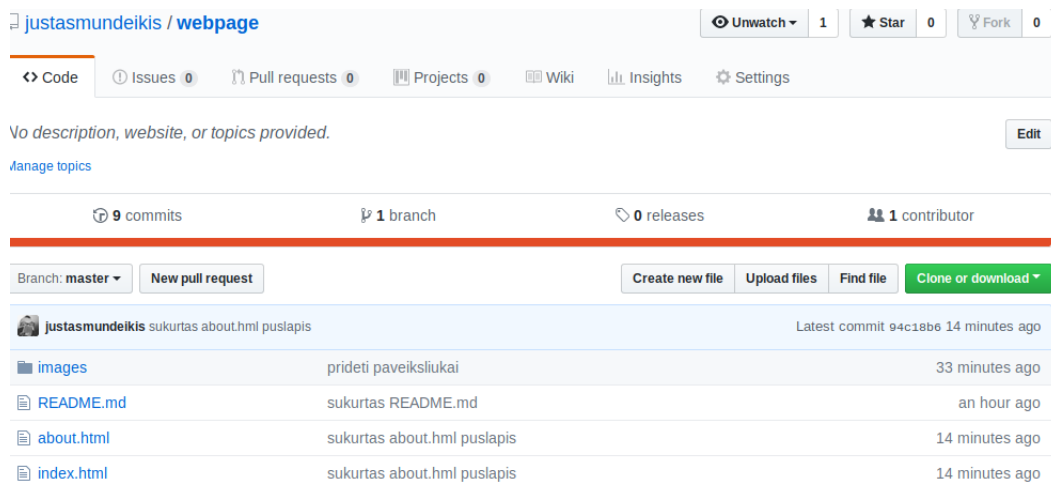
```

10 <p>Šio tinklapio ūkimas yra 1.Seminaro darbas </p>
11 <br>
12 <a href="about.html">Daugiau informacijos apie mane</a>
13 <p></p>
14 <p>
15 </p>
16 </body>
17 </html>

```

taigi įterpiame `Daugiau informacijos apie mane`, tagas `<a>` reiškia "an-chor" arba "inkaras" ir yra naudojamas nuorodoms.

34. `git add . && git commit -m "sukurtas about.html puslapis"`
35. Tačiau esame atšakoje "about-puslapis", todėl grįžtame į "master" su `git checkout master`. Jame suprantama nėra pakeitimų, kurios padarėme atšakoje, todėl nusprendžiame sujungti atšaką su master naudodami komandą `git merge about-puslapis`.
36. Jeigu nėra kokių nors jungimo klaidų, kurias reiktų pataisyti ranka, nuostabu!
37. Jeigu nebereikia about-puslapis atšakos, kuri yra sujungta (merge), galime ją ištrinti: `git branch -d about-puslapis`.
38. Jeigu būtume sukūrę atšaką, bet joje nieko prasmingo nesukūrė, flag -d neleistų ištrinti atšakos, tada reiktų naudoti flag -D, pvz., `git branch -D about-puslapis`.
39. patikriname kiek atšakų turime su `git branch`.
40. Dabar galima
41. `git push`. Po sujungimo staginti ir commitinti nebereikia, nes staginimas ir commitinimas įvyko dar about-puslapis atšakoje!
42. dabar reikia nueiti į "Settings"



43. Ties "GitHub Pages" iš "None" pasikeisti į "master branch"

GitHub Pages

GitHub Pages is designed to host your personal, organization, or project pages from a GitHub repository.

Source
GitHub Pages is currently disabled. Select a source below to enable GitHub Pages for this repository. [Learn more.](#)

None **Save**

Select source

- master branch**
Use the master branch for GitHub Pages.
- master branch /docs folder**
Use only the /docs folder for GitHub Pages.
- ☒ **None**
Disable GitHub Pages.

Make this repository private
Hide this repository from the public

Make private

44. Paspaudus nuorodą:

GitHub Pages is designed to host your personal, organization, or project pages from a GitHub repository.

Your site is ready to be published at <https://justasmundeikis.github.io/webpage/>.

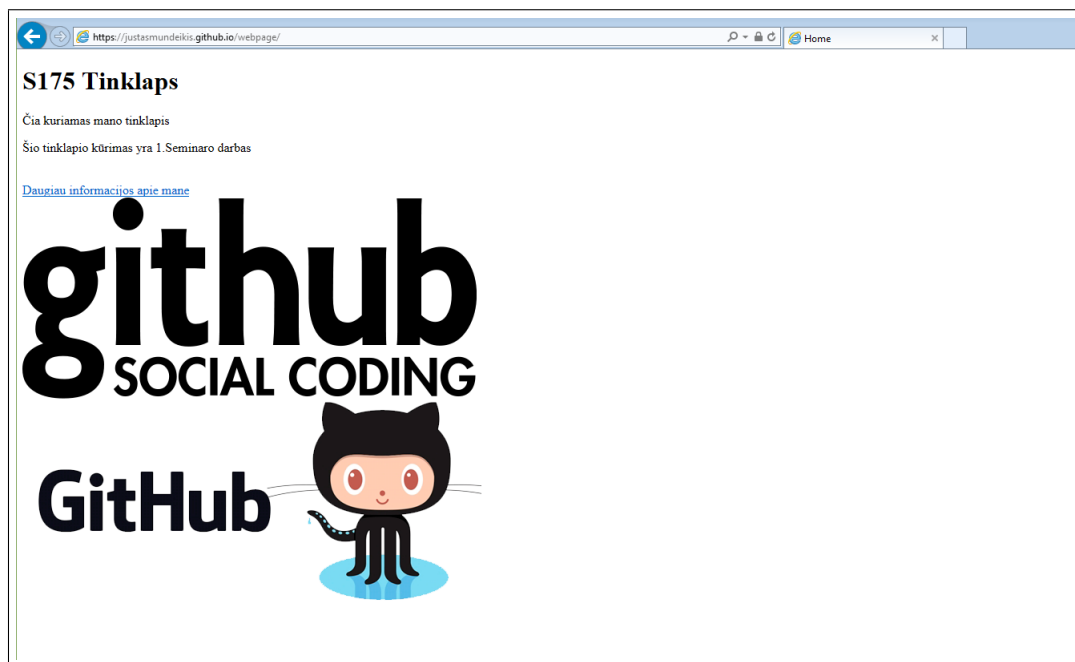
Source
Your GitHub Pages site is currently being built from the master branch. [Learn more.](#)

master branch **Save**

Theme Chooser
Select a theme to publish your site with a Jekyll theme. [Learn more.](#)

Choose a theme

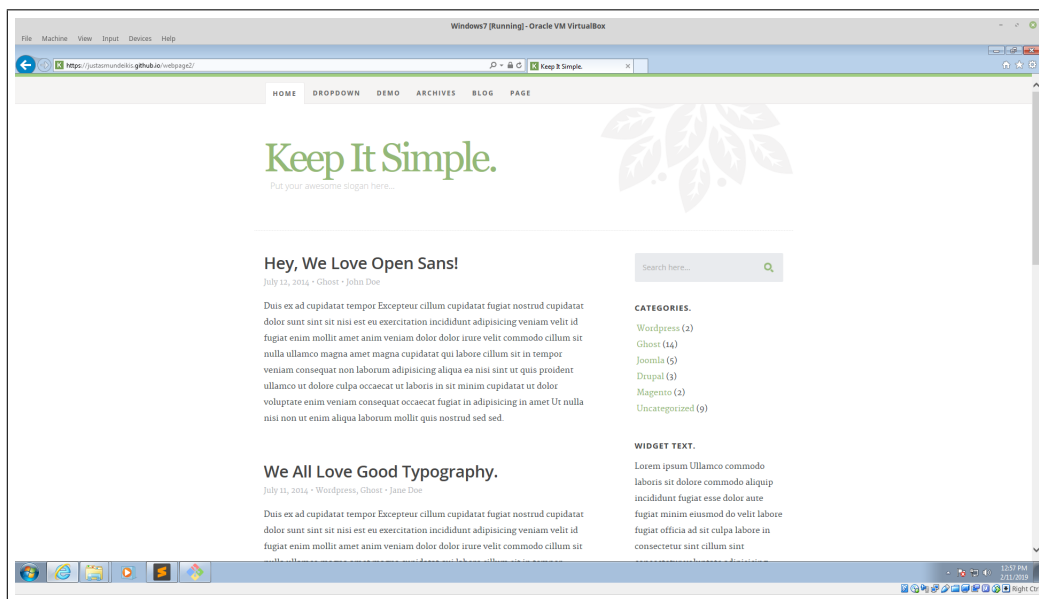
45. Atsidaro pilnai veikiantis tinklapis! Sveikinu!



46. Jeigu norite sukurti labiau sofistikuatą puslapį, galima naudotis html editoriaus pagalba, pvz., <https://html-online.com/editor/>

4 Kaip sukurti nuosavą homepage per 5 min

- Github paskyroje sukuriame naują *repo* pvz., "webpage2"
- Parsisiunčiam iš tinklapio <https://designseer.com/free-blog-html-css-templates/> "KeepSimple" tinklapio paketą išsaugant jį S175 folderyje
- Unzipiniam atsisiųstą failų paketą su GitBash `unzip KeepItSimple20.zip`
- Pereinam į išpakuotą folderį, inicializuojame git, pridame visus failus, ir commitiname naudojant sujungtą komandą:
`cd KeepSimple20 && git init && git add . && git commit -m "sukelti failai"`
- sujungiame su Github `git remote add origin https://github.com/<username>/webpage2.git`
- ir pushinam į Github `git push -u origin master`
- "Settings" esančioje "GitHub pages" dalyje, "Source" pakeičiame į "master"
- ir nuspaudus ant nuorodos turime veikiantį tinklą



- tiesa, jis kiek sudėtingesnis, turi Java Script, CSS ir t.t., bet pati struktūra yra aiški
- Kuo paprastesnį Template susirasite, tuo papračiau galima turėti savo tinklą ir jį keisti
- Taip pat puslapį galima susikurti ir tiesiog pačiame Github