



AKADEMIJA.IT

INFOBALT IR TECH CITY

ĮVADAS. GIT

Andrius Stašauskas

andrius@stasauskas.lt

<http://stasauskas.lt/itpro2018/>

INFORMACIJA

- 14 paskaitų - 13 teorija/praktika + 1 konsultacija
- Pirmadieniais, Antradieniais ir Ketvirtadieniais 8:55 - 12:45
- 5 savaitės:
 - Lapkričio 19, 20, 22 d. - 3 paskaitos
 - Lapkričio 26, 27, 29 d. - 3 paskaitos
 - Gruodžio 3, 4, 6 d. - 3 paskaitos
 - Gruodžio 10, 11, 13 d. - 3 paskaitos
 - Gruodžio 17, 18, 19, 20 d.
 - 1 paskaita, 1 konsultacija, 1 testas, 1 kontrolinis



KURSO PROGRAMA

- Versijų kontrolės sistemos: Git
- Web UI technologijos, jų istorija
- Javascript moduliai plačiau: scoping, unit testai, paketų/modulių administratorius, transpiliavimas
- ReactJs biblioteka
- Verslo sistemų kūrimas
 - maven, tomcat, spring boot, istorija, mvc, request/reponse, DI, layers, rest api, unit testing
- Persistence (JPA)
 - orm, persistence api, jpql, transactions, integracija su spring, entities, relationships (multi), CRUD



KĄ GAMINSIME

- Gaminsime supaprastintą el. parduotuvės variantą
- Vartotojo sąsaja - Javascript + ReactJs + Bootstrap CSS
- Serveris - REST Api (Spring Boot + Java 8)
- Duomenų bazė - H2 reliacinė duomenų bazė + JPA (Java Persistence Api)
- Tikslai:
 - kurso programos techninis tikslas - išmokti ir sukurti pilnai veikiančią aplikaciją
 - aukštesnis tikslas - suprasti modernaus Web kūrimo kontekstą, mokėti atpažinti naudojamą technologijas



EL. PARDUOTUVĖS FUNKCIJOS

- Peržiūrėti prekių pasiūlą
- Peržiūrėti prekės detales
- Įsidėti prekę į krepšelį
- Peržiūrėti krepšelio turinį
- Pašalinti iš krepšelio
- Pirkti prekes krepšelyje
- Administruoti prekes: pridėti, išmesti, atnaujinti



GIT PRADMENYS



AKADEMIJA.IT

INFOBALT IR TECH CITY

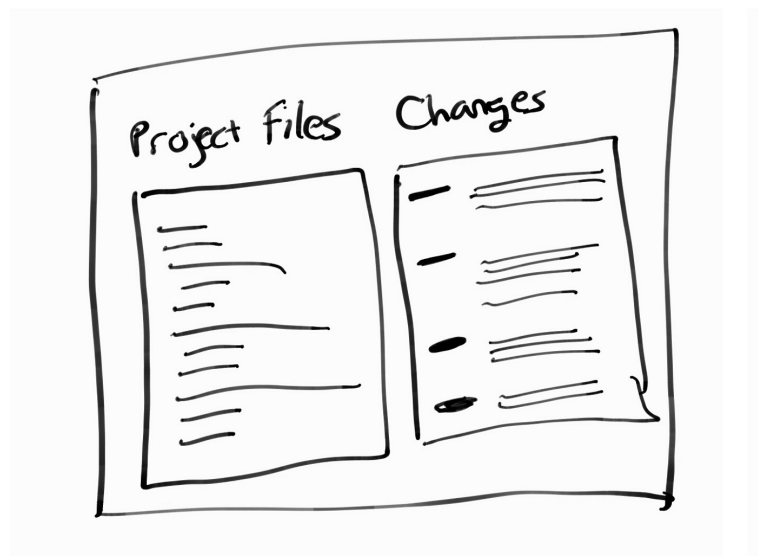
TURINYS

- Versijų kontrolės istorija
- Git versijų kontrolė
- Git šakos
- Git nutolusios repozitorijos
- Užduotys tarp teorijos



VERSIJŲ KONTROLĖ

- Angliškai - version control, revision control, source control
- Skirta suvaldyti dokumentų pakeitimus, kuriuos atlieka keli vartotojai
- Įsivaizduokite, jog visi turite įrašyti savo vardą į tą patį tekstinį failą

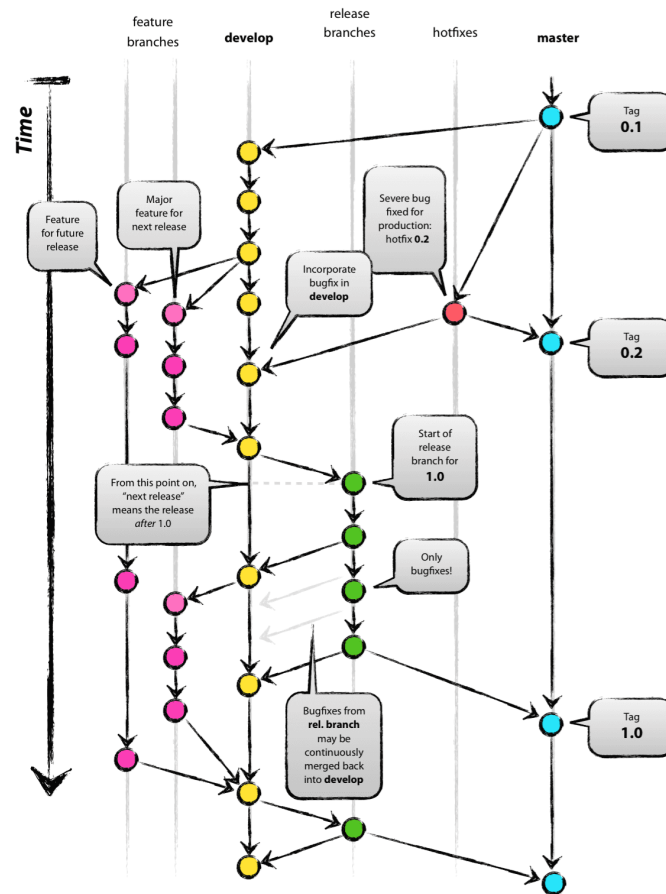


VERSIJŲ KONTROLĖS KARTOS

Karta	Tinklas	Pvz
Pirma	Jokio	RCS, SCCS
Antra	Centralizuotas	CVS, Subversion, Team Foundation Server
Trečia	Išskirstytas	Bazaar, Git, Mercurial



ŠAKOS



ŠAKOS



@ashk31

Repo arba Repository - centralizuotas katalogas



AKADEMIJA.IT

INFOBALT IR TECH CITY

KOMANDŲ PAGALBA

```
$ git help <command>
```

Komandos ir jos parametrų aprašymas

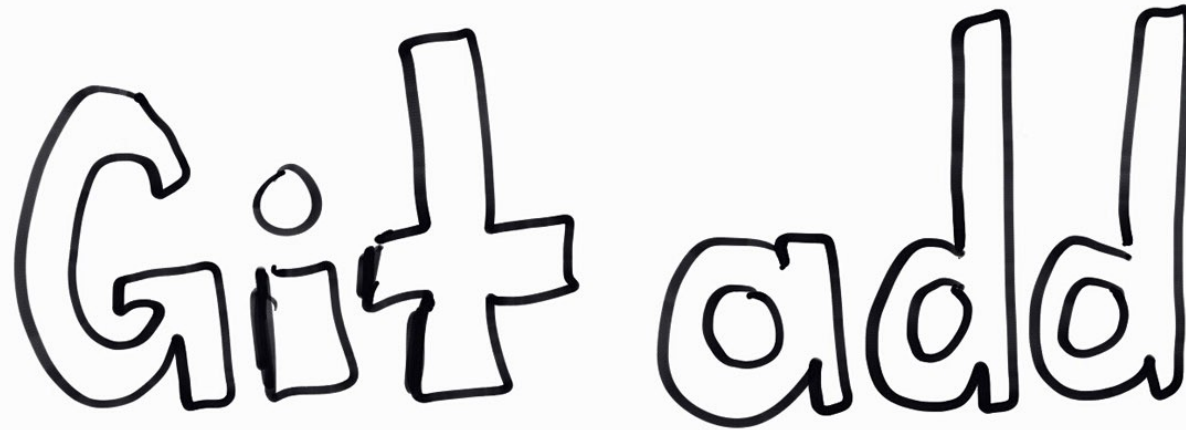


REPOZITORIJOS PARUOŠIMAS

`$ git init` - paruošia dabartinį katalogą kaip git
repozitoriją



FAILŲ PRIDĖJIMAS



Git add

@ashk31

```
$ git add <path>
```

Pridėti viską iš dabartinio katalogo:

```
$ git add .
```



FAILŲ PRIDĒJIMAS



`$ git status` - dabartinē repozitorijas būsena.

- Po pakeitumu `git status + git log`



FAILŲ PRIDĖJIMAS

```
→ testproject git:(master) X git status
On branch master
Your branch is ahead of 'origin/master' by 1 commit.
  (use "git push" to publish your local commits)
Changes to be committed:
  (use "git reset HEAD <file>..." to unstage)

        modified:   README.md
        renamed:     oldname.txt -> newname.txt

Changes not staged for commit:
  (use "git add/rm <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

        modified:   README.md
        deleted:    test.txt

Untracked files:
  (use "git add <file>..." to include in what will be committed)

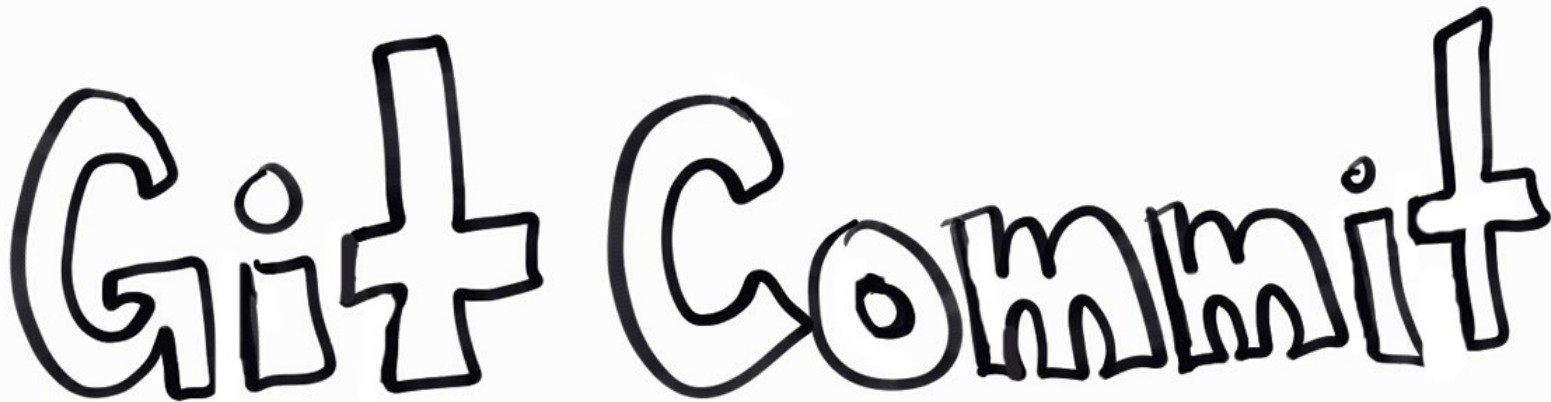
        newfile.txt

→ testproject git:(master) X
```

Bus "sucommitinti" žalių failų pakeitimai



SUPAKUOTI Į COMMIT'Ą

A large, hand-drawn illustration of the words "Git Commit" in a thick, black, outlined font. The letters are slightly irregular and stylized, giving it a sketchy, informal appearance. The text is centered horizontally and occupies the middle portion of the slide.

Git Commit

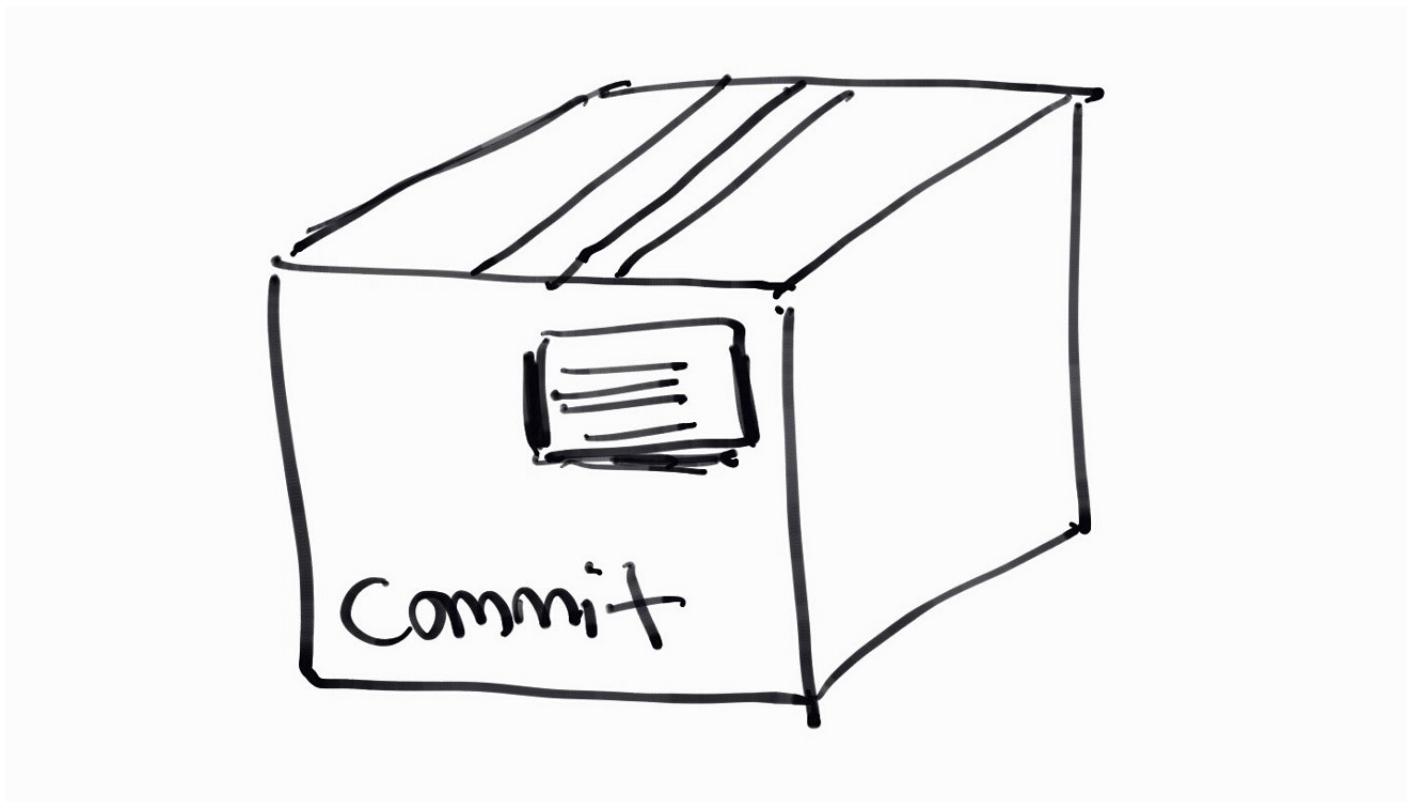
@ashk31

```
$ git commit -m <message>
```

Komanda, atliekama pabaigus darbą ar jo etapą su failais.
Susikuria changeset'as, turintis informaciją apie
pasikeitusius failus.



SUPAKUOTI Į COMMIT'Ą



@ashk31

Pakeitimai supakuoti į commit'ą kuris nebemodifikuojamas



AKADEMIJA.IT

INFOBALT IR TECH CITY

COMMIT'Ų ISTORIJA

\$ git log - commit'ų istorija dabartinėje šakoje

```
commit c0326d2386dd1227f35f46f1c75a8f87e2e93076
Author: Hudson @ build.clojure.org <build@clojure.com>
Date:   Fri Oct 28 14:24:47 2016 -0500

    [maven-release-plugin] prepare for next development iteration

commit e3c4d2e8c7538cfda40accd5c410a584495cb357
Author: Hudson @ build.clojure.org <build@clojure.com>
Date:   Fri Oct 28 14:24:47 2016 -0500

    [maven-release-plugin] prepare release clojure-1.9.0-alpha14

commit b80e1fe4b14654d943e2f8b060b0bc56e18b4757
Author: Nicola Mometto <brobronsa@gmail.com>
Date:   Fri Oct 7 21:23:39 2016 +0100

    CLJ-1242: equals doesn't throw on sorted collections

    Signed-off-by: Stuart Halloway <stu@cognitect.com>

commit 2ff700ede3866f97d7b1f53342e201df94384aee
Author: Nicola Mometto <brobronsa@gmail.com>
Date:   Sat Nov 7 02:58:40 2015 +0100

    CLJ-1790: emit a cast to the interface during procol callsite

    Signed-off-by: Stuart Halloway <stu@cognitect.com>

commit c6b76fadb4750c8f73d842cfdcf882b4a05683cae
Author: Alex Miller <alex.miller@cognitect.com>
Date:   Fri Oct 28 08:42:26 2016 -0500

    CLJ-2024 stest/check should resolve function spec

    Signed-off-by: Stuart Halloway <stu@cognitect.com>
```



UŽDUOTIS 1

- susikurkite katalogą: `studentai`
- inicijuokite jame git repozitoriją
- sukurkite failą `studentai.dat`
- atverkite failą ir įrašykite į jį savo vardą ir pavardę
- "sucommitinkite" pakeitimus
- peržvelkite, ar istorijoje atsirado commit'as



UŽDUOTIS 1 - SPRENDIMAS

```
$ mkdir studentai  
$ cd studentai  
$ git init  
$ echo "Jonas Petraitis" >> studentai.dat  
$ git add studentai.dat  
$ git commit -m "Added my name to studentai.dat"  
$ git log
```



UŽDUOTIS 1

- Pirmą kartą paleidus git

```
git config --global user.email "you@example.com"  
git config --global user.name "Your Name"
```

- Pirmas pakeitimas add, commit, bet po antro pakeitimo nėra add ir/ar commit



.GITIGNORE

- dažnai turime failų, kurie neturi būti commit'inami
 - `.eclipse` konfigūracija
 - `target` katalogas atsirandantis build'o metu
 - kiti failai specifiniai konkrečiam vartotojui
- `.gitignore` tekstinis failas leidžia nurodyti šablonus kelių, kurie neturi būti tvarkomi Git'o



.GITIGNORE

```
.gradle  
/build/  
!gradle/wrapper/gradle-wrapper.jar  
h2  
logs  
  
target  
*.iml  
.idea
```



ŠAKOS (BRANCH)



ŠAKOS (BRANCH)

- pagal nutylėjimą sukuriamas "master branch'as"
 - Mercurial default, SVN trunk, CVS HEAD
- naujus branch'us kuria vartotojai
- kiekvienai programavimo užduočiai naudojami atskiri branch'ai. Pabaigtos užduoties pakeitimai perkeliami į "master" branch'ą



ŠAKOS (BRANCH)

```
$ git checkout -b test  
Switched to a new branch 'test'
```

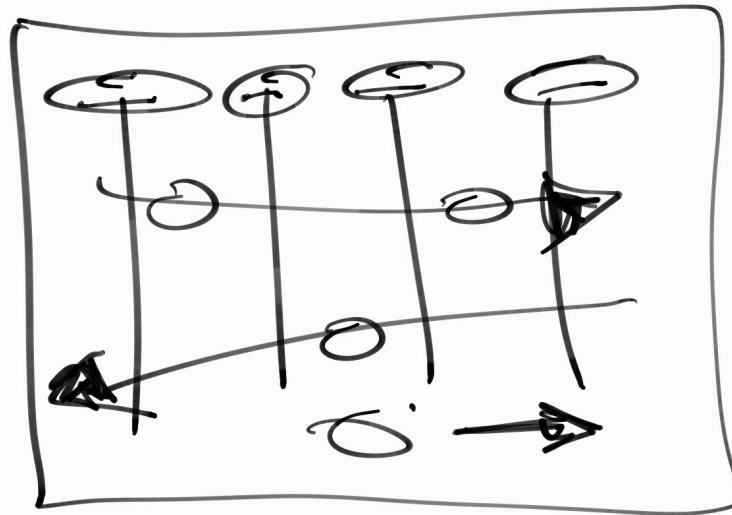
```
$ git branch  
  master  
* test
```

```
$ git checkout master  
Switched to branch 'master'
```

```
$ git branch  
* master  
  test
```



ŠAKOS (BRANCH)



\$ `git merge <branch-name>` - Atlikti merge commit'ą, perkeliant nurodyto branch'o pakeitimus į dabartinį branch'ą



ŠAKOS (BRANCH)

```
$ git branch
* master
  test
$ echo "master branch tekstas" >>
file$ git add .
$ git commit -m "Master commitas"
[master 2a5c8f1] Master commitas
1 file changed, 1 insertion(+)
```



ŠAKOS (BRANCH)

```
$ git checkout -b naujas
Switched to a new branch 'naujas'
$ echo "naujas branch tekstas" >>
file$ git add .
$ git commit -m "Naujas commitas"
[naujas 52f1927] Naujas commitas
 1 file changed, 1 insertion(+)
$ git checkout master
Switched to branch 'master'
```



ŠAKOS (BRANCH)

```
$ cat file
master branch tekstas
$ git merge naujas
Updating 2a5c8f1..52f1927
Fast-forward
  file | 1 +
  1 file changed, 1
insertion(+) $ cat file
master branch tekstas
naujas branch tekstas
```



GIT ISTORIJS NARŠYMAS

- Dažnai labai patogiu tai daryti grafiniais įrankiais
 - gitg, gitk, git gui (git-gui)
- Jei git status kažkas yra, patogiau git gui

```
$ sudo apt-get install gitg gitk git-gui  
$ gitg
```



GITG

The screenshot displays the Git GUI interface for a repository named 'genesis'. The top bar shows the repository name and the current branch, 'origin/GENESIS-2981_Quote_expired_batch_2'. The left pane lists the commit history, with the most recent commit being 'GENESIS-2981_Quote_expired_batch_4'. The middle pane shows a graphical representation of the commit history, with branches and merges visualized as lines and dots. The right pane shows the diff for the selected commit, highlighting changes in the file 'core/policy/core/policy-lifecycle/src/main/java/com/eisgroup/genesis/policy/core/lifecycle/commands/quote/BatchExpireHandler.java'. The diff shows several lines of code being added and modified. The bottom pane shows the file 'core/pom.xml' with a diff showing changes to the 'core/test-product/test-product-its/bpm-its/test/resources/local.properties' file.

genesis (C:\)
GENESIS-2981_Quote_expired_batch_2

GENESIS-2981_Quote_expired_batch_4
GENESIS-2981_Quote_expired_batch_3
GENESIS-2981_Quote_expired_batch_2
GENESIS-2981_Quote_expired_batch_1
GENESIS-2981_negative_solr_query
Genesis-2796_batch_processing_migration
GENESIS-0000-stale-artifact-fix
GENESIS-2570_Schedul...Manager-implementation
GENESIS-2793_GenericDatabaseAPI
GENESIS-2570-column-s...ding-extended-interfaces
GENESIS-2473_error_msgs_dot_fix
GENESIS-2433_crmv0001_message_is_incorrect
GENESIS-2171_not_found_for_banking
GENESIS-580_initial_policy_operations_history
GENESIS-2290_refactor...it_revision_to_use_CRUD
GENESIS-2172_should_n...not_latest_policy_version
GENESIS-2103-patched-transit-revision
GENESIS-2172_validate_latest_revision
GENESIS-1959_corrupted_keys_fix
GENESIS-2159_dev_update
GENESIS-1701_term_reduce
GENESIS-1701_cancel_update
GENESIS-2120_term_datefix
GENESIS-2112_privileges_update
GENESIS-2084-total-premium-clear
GENESIS-1701_cancel_action
GENESIS-2057_space_fix
GENESIS-1982_bulk_fields
GENESIS-1982_copy_handlers_update
GENESIS-569_Cancel_N...ovements_purple_cherry
GENESIS-1508_Create_Policy_Rewrite_action
noop-fix-pink-cherry
noop-fix
GENESIS-569_Cancel_Notice_action_improvements
GENESIS-1747_Genesis_Full_App
allstone-test
▼ Nutolusione saugyklos
▼ origin
GENESIS-3150_operatio...rocessingStage_missed

GENESIS-2981_Quote_expired_batch_2 (master)
Merge branch 'GENESIS-2981_Quote_expired_batch_2' into 'master'
GENESIS-2981: rename state to quoteState
Merge branch 'GENESIS-3630_GHPProduct' into 'master'
origin/GENESIS-3630_GHPProduct GENESIS-3630 "GHP" product is not supported during Opportunity association
Update Jenkinsfile-libs-CG
GENESIS-2981_Quote_expired_batch_2 (origin/GENESIS-2981_Quote_expired_batch_2)
GENESIS-2981: add option to configure cassandra port
Merge branch 'GENESIS-3429_Organization_DSL_model' into 'master'
Merge branch 'GENESIS-3636_TA_Regression_Refactoring' into 'master'
Merge branch 'master' of http://vnoeisgengit02.exigengroup.com/git/genesis/eis-genesis into GENESIS-2981_Quote_expired_batch
GENESIS-2981: hostname and port was used in server, so remove them to let cassandra client to decide [update]
GENESIS-3636 Regression refactoring regarding istf migration
Update Jenkinsfile-libs-CG
GENESIS-2981: hostname and port was used in server, so remove them to let cassandra client to decide
Merge branch 'GENESIS-0000_QA_docker_build_fix' into 'master'
Merge branch 'GENESIS-3424_decisionTableEngine' into 'master'
GENESIS-2981: fix CG itest commands to compile itest projects
GENESIS-0000 qa build fix 2
GENESIS-0000 qa build fix
Merge remote-tracking branch 'origin/master' into GENESIS-3424_decisionTableEngine
Merge branch 'GENESIS-3577_Fix_sonar_violations' into 'master'
Merge branch 'GENESIS-2981_Quote_expired_batch' of http://vnoeisgengit02.exigengroup.com/git/genesis/eis-genesis into GENESIS-2981_Quote_expired_batch
Merge branch 'master' of http://vnoeisgengit02.exigengroup.com/git/genesis/eis-genesis into GENESIS-2981_Quote_expired_batch
Merge branch 'GENESIS-3469_provide_link_to_logs' into 'master'
origin/GENESIS-3469_provide_link_to_logs GENESIS-3469 set title fix
origin/GENESIS-3424_decisionTableEngine GENESIS-3424 refactored engine domain and introduced propertyName
Merge branch 'GENESIS-3473_Sonar_optimizations' into 'master'
GENESIS-3469 set title
GENESIS-3469 initial
origin/GENESIS-3473_Sonar_optimizations GENESIS-3473 fix time offset
GENESIS-3473 fix3
GENESIS-3473 fix2
GENESIS-3473 fix1

eis_build <eisbuild@eisgroup.com>
2017-10-31 14:06:26 +0200

Revert "Merge branch 'GENESIS-2981_Quote_expired_batch_2' into 'master'"

134 core/policy/core/policy-lifecycle/src/main/java/com/eisgroup/genesis/policy/core/lifecycle/commands/quote/BatchExpireHandler.java
22 core/policy/core/policy-lifecycle/src/main/java/com/eisgroup/genesis/policy/core/lifecycle/modules/PolicyLifecycleModule.java
9 core/policy/core/policy-repository/policy-repository-core/src/main/java/com/eisgroup/genesis/policy/core/entity/version/impl/QuoteVersionEntity.java
10 core/policy/core/policy-repository/policy-repository-core/src/main/java/com/eisgroup/genesis/policy/core/entity/version/impl/QuoteVersionFactory.java
2 core/pom.xml
19 core/test-product/pom.xml
2 core/test-product/test-product-its/bpm-its/test/resources/local.properties

77f60327839a9b7ab7475ac607704
Dropbox 39.4.9
Up to date



AKADEMIJA.IT
INFOBALT IR TECH CITY

GIT RODYKLĒS (POINTERS)

- HEAD - dabartinės būsenos rodyklė
- Daugiau pavyzdžių
<http://www.paulboxley.com/blog/2011/06/git-caret-and-tilde>



COMMIT ID

- 52f192703d9951f5bbaa55fc41c0c78bba811a49
- unikalčiai nusakyti commit'ą pakanka 7 pirmų simbolių
52f1927
- žinant commit'o id, galima atlikti visokių veiksmų,
pavyzdžiui nukreipti HEAD'ą į šį commit'ą



COMMIT ID

```
$ git checkout 52f1927
```

You are in 'detached HEAD' state. You can look around, make experimental changes and commit them, and you can discard any commits you make in this state without impacting any branches by performing another checkout.

If you want to create a new branch to retain commits you create, you can do so (now or later) by using `-b` with the checkout command again. Example:

```
git checkout -b <new-branch-name>
```

```
HEAD is now at 52f1927... Naujas commitas
```



MERGE KONFLIKTAI

- Įvyksta kai ta pati failo vieta buvo modifikuota kelių vartotojų
- Galimi merge ir pull komandų metu
- Ištaisius konfliktus failuose, juos reikia add'inti ir "sucommitinti" nekeičiant commit'o pranešimo
- Konfliktų sprendimą palengvina grafiniai įrankiai (pvz kdiff, meld, tortoisemerge)

```
$ git mergetool
```



MERGE KONFLIKTAI

Sukonfigūruoti galima .gitconfig

```
[merge]
  tool = kdiff3
[mergetool "kdiff3"]
  path = /usr/bin/kdiff3
```

- Sukonfigūravus merge'inti daug greičiau ir patogiau su `git mergetool` (vietoj teksto)
- Po merge'ų peržiūrai kas atsitiko patogiau naudoti `gitg`
 - `apt-get install gitg`



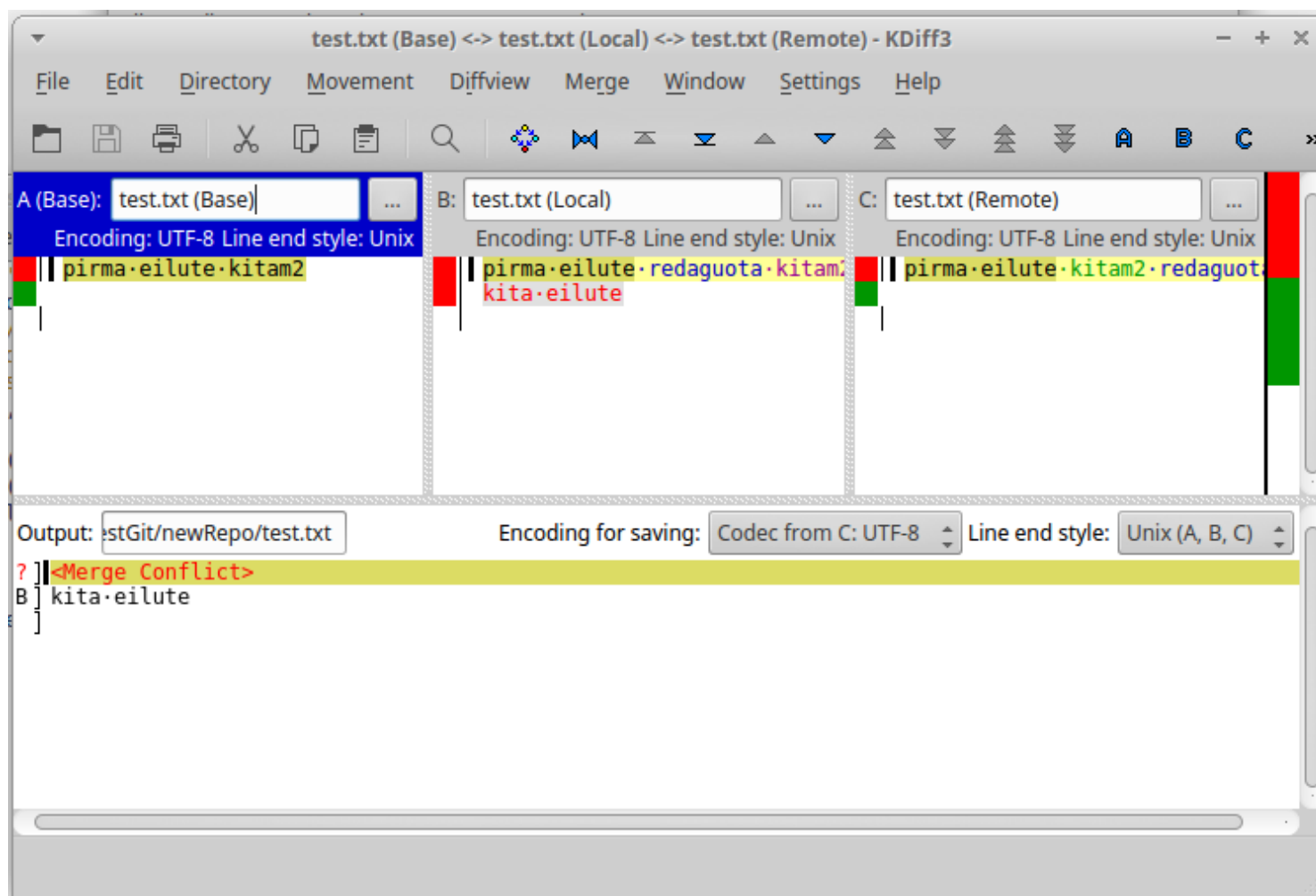
KONFLIKTO PAVYZDYS - TEXT

```
Jonas Petraitis  
<<<<<< HEAD  
1985-01-09  
=====  
Vilnius  
>>>>>> gimimo-vieta
```

- <<<<<< ir >>>>>> žymi konfliktuojančios dalies pradžią ir pabaigą
- <<<<<< HEAD nurodo, jog konfliktuojantis blokas yra iš branch'o, į kurį "merginame"
- ===== atskiria dalis tarp branch'ų
- >>>>>> gimimo-vieta nurodo, jog dalis tarp ===== ir >>>>>> yra iš branch'o gimimo-vieta



KONFLIKTO PAVYZDYS - KDIFF



UŽDUOTIS 2

- Įsitikinkite, kad esate master branch'e
- Sukurkite branch'ą pavadinimu gimimo-data
- Tame branch'e į failą studentai.dat naujoje eilutėje įrašykite savo gimimo datą
- "Sucommitinkite"
- Grįžkite į master branch'ą



UŽDUOTIS 2

- Sukurkite branch'ą gimimo-vieta
- Tame branch'e į failą studentai.dat naujoje eilutėje įrašykite savo gimimo vietą
- "Sucommitinkite"
- Grįžkite į master branch'ą



UŽDUOTIS 2

- "Įmerginkite" gimimo datos pakeitimus į master branch'ą
- "Įmerginkite" gimimo vietos pakeitimus į master branch'ą



```
$ git checkout -b gimimo-data  
Switched to a new branch 'gimimo-data'
```

```
$ echo "1985-01-03" >>  
studentai.dat$ git add .  
$ git commit -m "Gimimo data"  
[gimimo-data 1da96ec] Gimimo data  
1 file changed, 1 insertion(+)
```

```
$ git checkout master  
Switched to branch 'master'
```



```
$ git checkout -b gimimo-vieta  
Switched to a new branch 'gimimo-vieta'
```

```
$ echo "Vilnius" >> studentai.dat  
$ git add .  
$ git commit -m "Gimimo vieta"  
[gimimo-vieta a309462] Gimimo vieta  
1 file changed, 1 insertion(+)
```

```
$ git checkout master  
Switched to branch 'master'
```



UŽDUOTIS 2 - SPRENDIMAS

```
$ git merge gimimo-data
Updating f9f86f9..1da96ec
Fast-forward
 studentai.dat | 1 +
 1 file changed, 1 insertion(+)
```

```
$ git merge gimimo-vieta
Auto-merging studentai.dat
CONFLICT (content): Merge conflict in studentai.dat
Automatic merge failed; fix conflicts and then commit the result.
```



```
$ gedit studentai.dat # pataisome konfliktus faile
```

```
Jonas Petraitis  
<<<<<< HEAD  
1985-01-03  
=====  
Vilnius  
>>>>>> gimimo-vieta
```

```
Jonas Petraitis  
1985-01-03  
Vilnius
```

```
$ git add .  
$ git commit  
[master 5c8fae3] Merge branch 'gimimo-vieta'
```



NUTOLUSIOS REPOZITORIJOS



DECENTRALIZACIJA

- Kiekviena Git'o repozitorija yra kažkurio laiko momento kopija.
- Ji gali nepriklausomai vystytis
- Dažniausiai yra Git Serveris, kuris yra centrinė repozitorija (origin)
- Populiariausi serveriai: <https://github.com>, <https://gitlab.com>, <https://bitbucket.org>



NUTOLUSIOS (REMOTE) REPOZITORIJOS

- Dažniausiai būna viena centrinė remote repozitorija, vadinama 'origin'
- Kuriant naują git repozitoriją (`git init`) ją reikia prisidėti pačiam (`git remote add`)
- Klonuojant (`git clone`) egzistuojančią repozitoriją, ji pridedama automatiškai
- Kad repozitorijos veiktų kartu, jų bazė turi būti ta pati



NUTOLUSIOS (REMOTE) REPOZITORIJOS

`$ git remote add <name> <repository>`

```
$ git remote add origin https://github.com/allstone/itpro.git
$ git remote -v
origin  https://github.com/allstone/itpro.git (fetch)
origin  https://github.com/allstone/itpro.git (push)
```

- Slaptažodžio pateikimas nuorodoje
 - `git clone` [https://username:password@github.com/..](https://username:password@github.com/)



NUTOLUSIOS (REMOTE) REPOZITORIJOS

`$ git clone <repository>`

```
$ git clone https://github.com/allstone/itpro.git
```



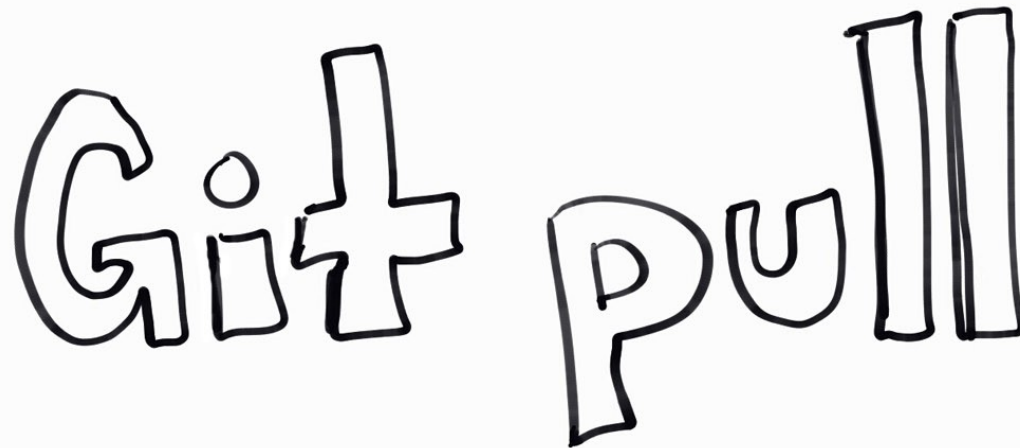
NUTOLUSIOS (REMOTE) REPOZITORIJOS

remote ar clone repozitorijos radimas

The screenshot shows a GitHub repository interface. At the top, there's a navigation bar with tabs: Code, Issues (0), Pull requests (0), Projects (0), Wiki, Insights, and Settings. Below this, a message states "No description, website, or topics provided." with an "Edit" button. A "Manage topics" link is also present. A summary bar shows "1 commit", "1 branch", "0 releases", and "1 contributor". Below the summary, there are buttons for "Branch: master", "New pull request", "Create new file", "Upload files", "Find file", and a green "Clone or download" button. A dropdown menu is open from the "Clone or download" button, showing options to "Clone with HTTPS" (selected), "Use SSH", and "Use Git or checkout with SVN using the web URL". The HTTPS URL is "https://github.com/allstone/itpro.git". At the bottom of the dropdown are buttons for "Open in Desktop" and "Download ZIP". The repository name "allstone test" is visible, along with a file named "test". A blue box at the bottom encourages adding a README.



LOKALIOS REPO SINCHRONIZACIJA



Git pull

@ashk31

\$ git pull arba \$ git fetch



PAKEITIMŲ ATSISIUNTIMAS



@ashk31

Gauname viską, kas keitėsi, t.y. ką kiti supakavo į commit'us ir yra nusiuntę į remote repository



PULL VS FETCH

- fetch - atsisieničia pakeitimus iš repozitorijos, bet nepritaiko jų lokaliai kopijai
- pull - padaro tą patį, ką ir fetch, bet pritaiko gautus pakeitimus, bet dėl to gali reikėti merge'o
 - techniškai git pull atveju vyksta tokia komandų seka, jei esame master'yje:

```
$ git fetch  
$ git merge remotes/origin/master
```

- dažniausiai visada darome `git pull` ir nesukame galvos
- atsinaujinti visus branch galima su `git pull --all`



LOKALIŲ PAKEITIMŲ IŠSIUNTIMAS Į REMOTE'Ą

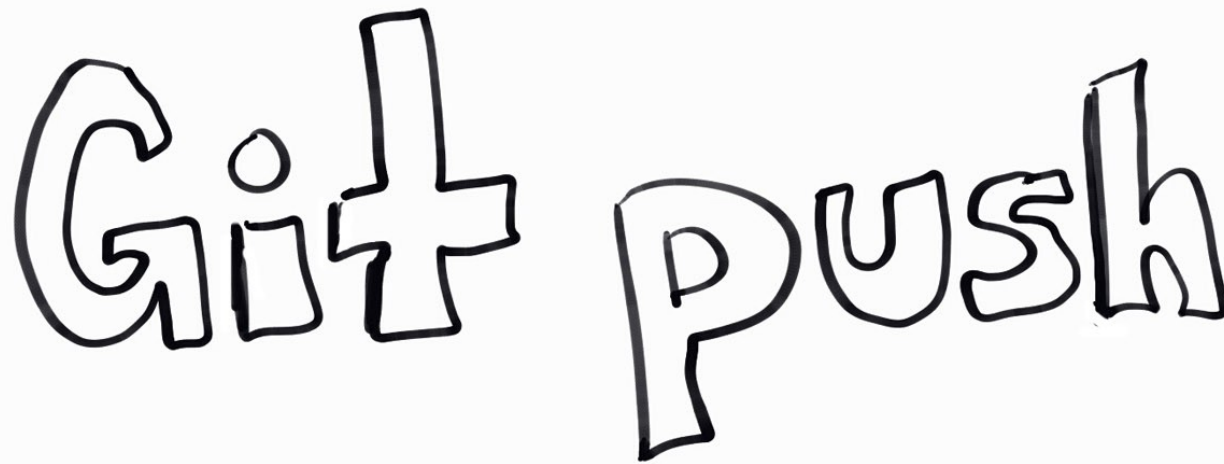


@ashk31

- norime išsiųsti visus savo commit'us į remote repository, kad jie būtų pasiekiami kitiems



LOKALIŲ PAKEITIMŲ IŠSIUNTIMAS Į REMOTE'Ą



Git push

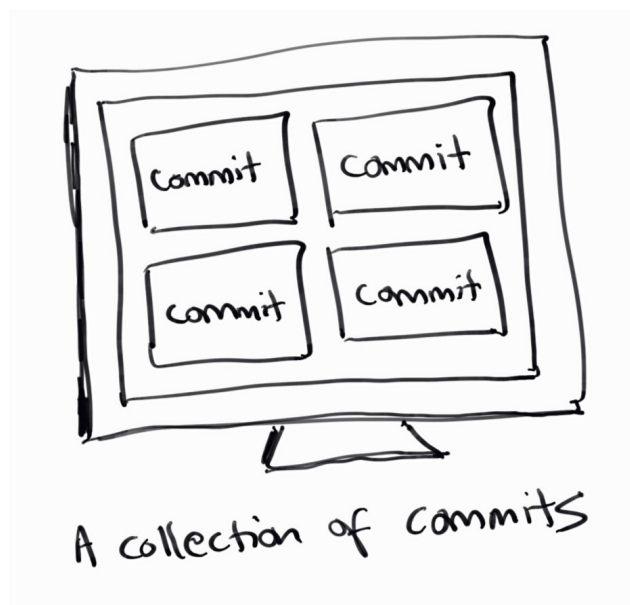
@ashk31

```
$ git push origin <branch-name>
```

- jei remote yra naujų pakeitimų, bus atspausdinta klaida
 - tokiu atveju reikia atlikti `git pull`, išspręsti merge konfliktus ir bandyti dar kartą



KAS BUS SIUNČIAMA



- praktiškai bus išsiunčiami visi commit'ai, esantys branche

UŽDUOTIS 3

- Susikurkite Github'o vartotoją
- Susikurti slaptažodį-token'ą, pirmą kartą atsiradusiame lange jį būtinai nusikopijuoti
 - <https://github.com/settings/tokens>
- Susikurkite Github'e repozitoriją
- Prisidėkite remote'ą
- Perkelkite į Github'ą prieš tai užduotyse sukurtą repozitoriją



UŽDUOTIS 4

- Susigrupuojame į komandas po 3-4 žmones
- Nusprendžiame, kieno repozitoriją "teršime"
- Vienas iš komandos narių susikuria branch'ą su vienu failu ir išsiunčia į nutolusią repozitoriją
- Kiti parsisiunčia pakeitimus - "įeina" į branch'ą
- Iš pradžių įdedame atskirus naujus 3 failus, kiekvienas savo, - išsiunčiame
- Tada darome pakeitimus pagrindiniame faile visi kartu ir bandome išsiųsti bei merge'inti
- Teisių suteikimas:

<https://github.com/<user>/<repo>/settings/collaboration>



ŠALTINIAI

- <https://git-scm.com/book/en/v2>
 - Pirmi 2 skyriai rekomenduojami
 - 3 skyrius smalsiems
- <http://learngitbranching.js.org/>
- dalis paveiksliukų: <https://medium.com/@ashk3l/a-visual-introduction-to-git-9fdca5d3b43a>



KITOJE PASKAITOJE

Web evoliucija. Javascript moduliai. ReactJS įvadas



AKADEMIJA.IT

INFOBALT IR TECH CITY