### 3. LEKCIJA

##### 3.1. Darbs komandrindā

Kā standarta rīks programmētājiem ir komandrinda ar komandrindas interfeisu. Tālākās komandas strādās uz GIT komandu interfeisa, LINUX komandu interfeisa un arī citiem UNIX bāzētiem produktiem, piemēram, MacBook komandrindas.

3.1.1. Visu failu un mapīšu saraksts

Lai redzētu kāds ir saraksts visiem failiem un direktorijām uz kuriem mēs varam aiziet/izmantot ir komanda ls.

3.1.2. Pārvietošanās pa mapēm

Lai komandu interfeisā (komandrindā) pārvietotos par mapēm ir sekojoša komanda:

cd – jebšu change directory, kas nozīmē direktoriju (jebšu mapīšu ceļu mainīšanas adrese)

Standartā, atverot komandrindu, būsim nokļuvuši lietotāja mājas mapē (Kornetā lietotāja ar kuru esat ielogojies tajā brīdī datorā). Windows sistēmās tas jau ir šis ceļš (manā gadījumā): C:\Users\Martins Kruze\

Lai pārvietotos „uz augšu”, tiek lietots divu punktiņu pieraksts „..”, šai gadījumā, ja es esmu sava lietotāja mapē: C:\Users\Martins Kruze\ un veicu komandu cd .., tad es nokļūstu mapē C:\Users\

STAIGĀŠANAS PA MAPĒM PIEMĒRS

Lai veiktu staigāšanu pa mapēm, nepieciešams lietot cd un mapes nosaukums komandu, piemēram: esam mapē: C:\Users\Martins Kruze\, un viss saraksts šai mapei ir sekojošs:

Martins Kruze@MARTINSKRUZE-PC ~

$ ls

AppData

Application Data

Contacts

Cookies

Desktop

Documents

Downloads

Saved Games

ntuser.ini

putty.exe

repos

mums vajag mapē repos => iespeju\_tilts, tad komanda būs:

Martins Kruze@MARTINSKRUZE-PC ~

$ cd repos/iespeju\_tilts/

Martins Kruze@MARTINSKRUZE-PC ~/repos/iespeju\_tilts (master)

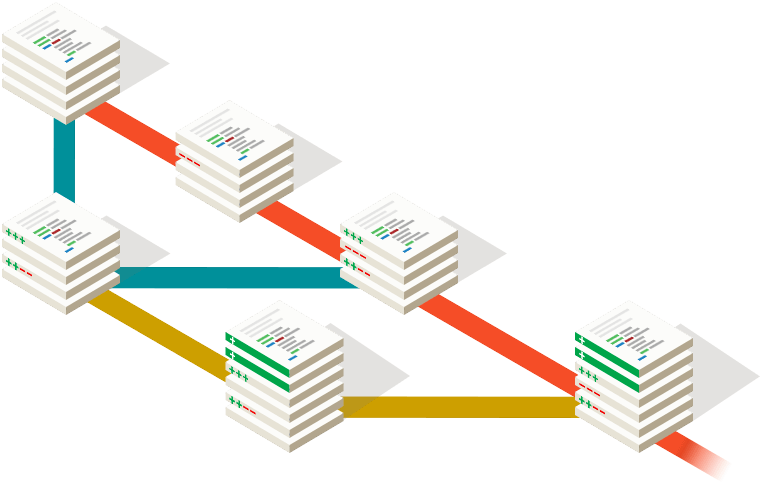
$ ls

01\_lekcija\_30\_09\_2014 02\_lekcija\_02\_10\_2014 03\_lekcija\_07\_10\_2014

##### 3.2.Kas ir GIT?

Kas ir Versiju Kontroles Sistēma – tā ir sistēma, kas ļauj saglabāt kodu, saglabāt, labot un atgriezties pie iepriekšējiem papildinājumiem un ļauj veiksmīgi vairākiem cilvēkiem strādāt pie viena un tā paša projekta, bez konstantas visa koda kopēšanas turpu, šurpu.

Eksistē vairākas versiju kontroles sistēmas, bet mēs lietosim Git (<http://git-scm.com/>) un GitHub konkrētai versiju saglabāšanai.



##### 3.3. Kā lieto GIT?

Jāsaprot, ka Git gadījumā ir viens *repozitorijs* – vieta kur visi faili un izmaiņas tajos tiek saglabātas. Piemēram, šīs lekcijas dokumentācijas git *repozitorija* adrese ir git@github.com:martinskruze/iespeju\_tilts.git

Git komandas (tās, kuras mēs visbiežāk lietosim (pilns saraksts: <https://training.github.com/kit/downloads/github-git-cheat-sheet.pdf>)):

git init – izveido jaunu repozitoriju

git add . – pievieno visu jauno, kas ir repozitorijā

git commit -am „papildinajuma nosaukums” – papildinājuma nosaukuma piešķiršana – šis ir ļoti nepieciešams, lai varētu pēc tam orientēties lielos projektos.

git remote add origin [repozitārija adrese] – pievienojam centrālo atrašanās vietu repozitorijam

git push -u origin master – sākotnējā uzstumšana uz repozitorija

git push – standarta uzstumšana uz repozitorija

Standarta gadījums

Standarta gadījums, ja nepieciešams pievienot jaunākās izmaiņas. Atrodamies repozitorija mapē:

git add .

git commit -am „izmainu nosaukums”

git push

jaunu izmaiņu novilkšanai:

git pull

##### 3.4. github un ssh atslēgas (konta piesaistīšana datoram)

Šis ir jādara vienu reizi, katram datoram:

Adrese, kurā uzstādīt GitHubam SSH atslēgas: <https://github.com/settings/ssh>

Adrese ar pamācībām SSH atslēgu uzstādīšanā: <https://help.github.com/articles/generating-ssh-keys/>

##### 3.5.ruby valodas palaišana

Lai palaistu kodu ruby valodā, nepieciešams lietot komandu ruby:

ruby [faila nosaukums]

Martins Kruze@MARTINSKRUZE-PC ~/repos/iespeju\_tilts/01\_lekcija\_30\_09\_2014/kods (master)

$ ruby pitagors.rb

hipotenuza ir: 5.830951894845301 cm