Csapatmunka támogató eszköz: Git

GitLab és Git elérés

- http://git-okt.sed.inf.szte.hu/users/sign_in
  - Belépés h-s azonosító/jelszó párral (LDAP fül)

- A létrehozott projekt azonosítói:
- http://gitlab-okt.sed.hu/2022_kurzuskod-gyakorlat_projektnev/projektnev.git
- git@git-okt.sed.inf.szte.hu:2022_kurzuskod-gyakorlat_projektnev/projektnev.git

Git használat

- Videó
  - https://www.youtube.com/watch?v=6yfBsmJJ4Fs
- Lokális repozitory kezelés
  - git init, git add, git commit, git status, git diff, git log
- Branchelés
  - git branch –r, git checkout –b nev, git merge

Git használat

- Remote repozitory kezelés
  - git remote, git clone, git fetch, git push, git pull

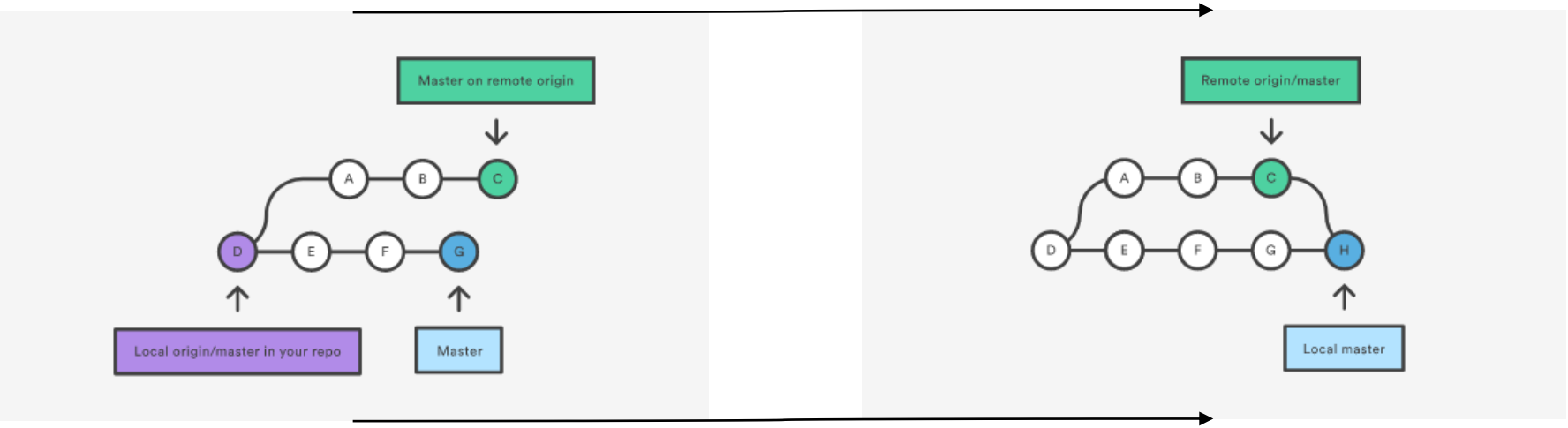
Git használat

- Távoli repozitory lemásolása (origin):
  - git clone
- Távoli kapcsolatok listázása:
  - git remote (-v, add nev url, rm nev)
- Feladat: másoljuk fel egy fájlt a git reponkba!
  - lásd: gitlab.txt

Git használat

- Távoli repozitory lemásolása (origin):

git fetch; git merge



git pull

- Lokális repo (módosításainak) feltöltése távolira:
  - git push remote-nev branch-nev
  - (git push origin master)

Git használat

- Konfliktus kezelés példa:
  - https://www.atlassian.com/git/tutorials/comparing-workflows#centralized-workflow
- Teljes példa – gitlab-os git használat:
  - rf_git_basics.pdf

Git cheatsheet

Git Cheat Sheet

For more awesome cheat sheets visit rebellabs.org!

REBELLABS by ZEROTURNAROUND

Create a Repository

From scratch -- Create a new local repository
```
$ git init [project name]
```

Download from an existing repository
```
$ git clone my_url
```

Observe your Repository

List new or modified files not yet committed
```
$ git status
```

Show the changes to files not yet staged
```
$ git diff
```

Show the changes to staged files
```
$ git diff --cached
```

Show all staged and unstaged file changes
```
$ git diff HEAD
```

Show the changes between two commit ids
```
$ git diff commit1 commit2
```

List the change dates and authors for a file
```
$ git blame [file]
```

Show the file changes for a commit id and/or file
```
$ git show [commit]:[file]
```

Show full change history
```
$ git log
```

Show change history for file/directory including diffs
```
$ git log -p [file/directory]
```

Working with Branches

List all local branches
```
$ git branch
```

List all branches, local and remote
```
$ git branch -av
```

Switch to a branch, my_branch, and update working directory
```
$ git checkout my_branch
```

Create a new branch called new_branch
```
$ git branch new_branch
```

Delete the branch called my_branch
```
$ git branch -d my_branch
```

Merge branch_a into branch_b
```
$ git checkout branch_b
$ git merge branch_a
```

Tag the current commit
```
$ git tag my_tag
```

Make a change

Stages the file, ready for commit
```
$ git add [file]
```

Stage all changed files, ready for commit
```
$ git add .
```

Commit all staged files to versioned history
```
$ git commit -m "commit message"
```

Commit all your tracked files to versioned history
```
$ git commit -am "commit message"
```

Unstages file, keeping the file changes
```
$ git reset [file]
```

Revert everything to the last commit
```
$ git reset --hard
```

Synchronize

Get the latest changes from origin (no merge)
```
$ git fetch
```

Fetch the latest changes from origin and merge
```
$ git pull
```

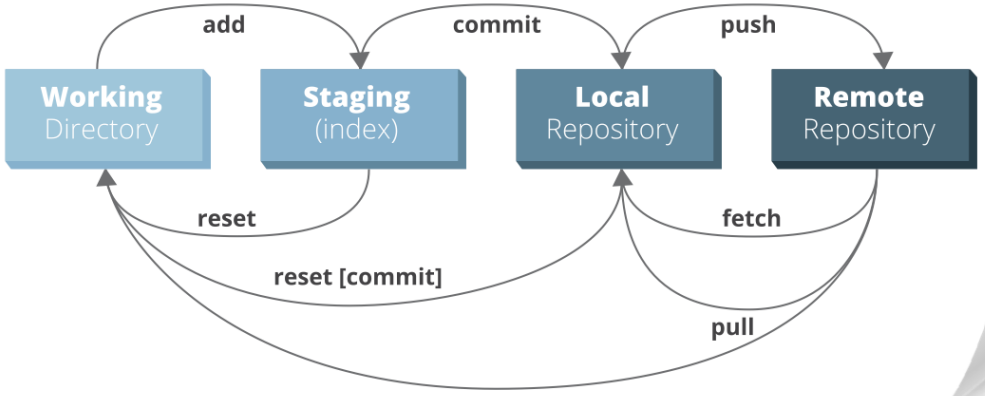
Fetch the latest changes from origin and rebase
```
$ git pull --rebase
```

Push local changes to the origin
```
$ git push
```

Finally!

When in doubt, use git help
```
$ git command --help
```

Or visit https://training.github.com/ for official GitHub training.



BROUGHT TO YOU BY JRebel

Teendők

- Git gyakorlás
- Projektterv készítés
- Jövő óra: UML tervezés