

M05 PRACA ZDALNA

L03 Pobranie repozytorium

(Początek owocnej współpracy)

Maciej Aniserowicz

DOWNLOAD

Clone

- ściąga repozytorium (w całości, z historią) w wybrane miejsce
- w uproszczeniu: kopiuje katalog `.git` i robi *checkout*
 - (choć nie do końca tak to działa pod spodem)
- wykonywane tylko RAZ dla nowego repozytorium

Git server

Git nie potrzebuje serwera.

Można sklonować dowolne repozytorium lokalnie.

(To tylko jeden z obsługiwanych przez Gita protokołów komunikacji)

Hosting na dysku

"Biedna" alternatywa dla *full-blown repo-hosting*, ale przydatne nawet przy pracy w pojedynkę.

Do czego?

01 Hosting na dysku

Na przykład jedno repo - traktowane jako "zdalne" -
synchronizowane **na Dropbox** jako "backup".

(pamiętasz? mówiliśmy dopiero co, że o backup trzeba dbać we własnym zakresie!)

02 Hosting na dysku

Praca nad kilkoma aspektami jednego repo jednocześnie.

Na przykład

- kod na jednym branchu
- dokumentacja / powerpoint na innym

Inne Working Copy w różnych katalogach.

03 Hosting na dysku

W firmie: repo Gita na *network share* dostępnym dla wszystkich.

Bez dedykowanego hostingu, bez wysyłania kodu poza firmę.

04 Hosting na dysku

Nauka i eksperymenty!

(idealny scenariusz dla nas, teraz)

Clone: demo



[repo 2]

1. `mkdir my_clone`
2. `cd my_clone`
3. `git clone ../my_project .`

(kropka: aktualny katalog)

Co otrzymaliśmy?



[repo 2]

1. gitk

wszystkie commity, wszystkie tagi

co to *remotes/origin/master*?

2. git status

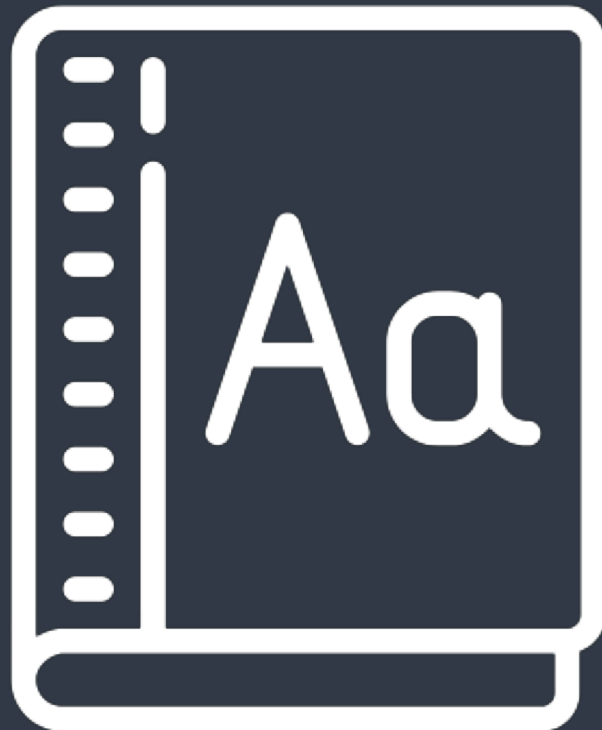
up to date with origin/master

3. git branch -vv

Remote

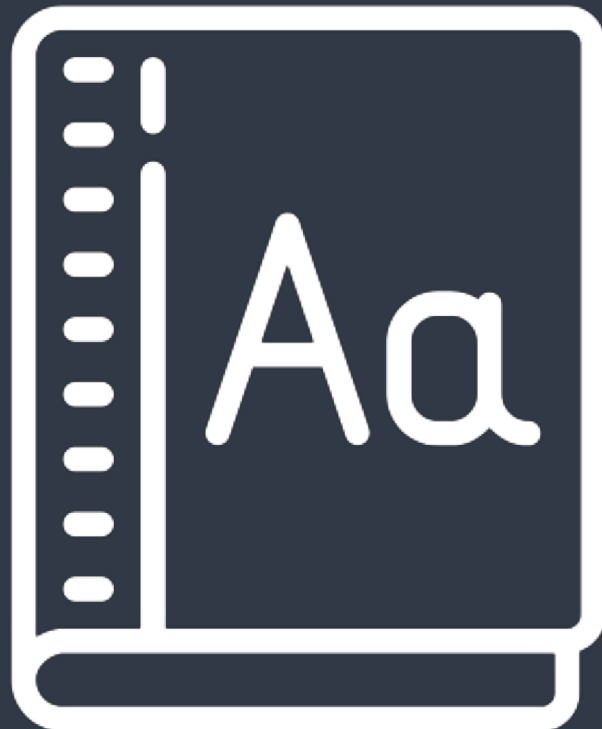
Zdalne
repozytorium.

(po prostu)



origin

Domyślna nazwa
zdalnego
repozytorium
będącego źródłem
CLONE.



Później...

...nauczymy się jak (i PO CO) tę nazwę **zmienić**,
ale w większości przypadków...

BEST PRACTICE

Miej w repozytorium
remote o nazwie "*origin*"

("bo taka konwencja")




Rozkodujmy "refs/..."

refs/remotes/origin/master:

- refs - **referencje** do commitów (wszystkie)
- remotes - commity w **zdalnych** repozytoriach
- origin - zdalne **repo** o nazwie "*origin*"
- master - **gałąź** "*master*" w repo "*origin*"

Proste? **Proste!**

Więcej info o **origin**

- 
1. `git remote` - wszystkie zdalne repo
 2. `git remote show origin` - szczegóły o "*origin*"

Gdzie Git to trzyma?

W plikach **tekstowych**! Katalog `".git"`, jak wszystko.

1. `cat .git/config`

Co z "oryginalnym" repo?

Nic!

Ono nawet nie wie, że zostało sklonowane.

Zmieńmy je!

UPLOAD?

("sneak peek")

Wysyłamy zmiany



[repo 2]

1. `git commit --allow-empty -m "ping remote"`
2. `gitk`
3. `git status`
4. `git branch -vv`

ahead 1

use "git push" to publish your local commits

OKEj!

Git push?

Nowa komenda.

Zaraz poznamy ją lepiej. Póki co...

push



Nowa komenda, zaraz poznamy ją lepiej. Póki co...

[repo 2]

1. git push

ALE GROŻNY KOMUNIKAT!

Nie można wysłać zmian!

Dlaczego? Bo klonowane repo jest w użyciu. **Służy do pracy!**

Wysyłając zmiany na gałęzi "master",
zmodyfikujemy komuś gałąź, na której pracuje!

Jak to rozwiązać? Jak podzielić się kodem?

Do wymiany kodu służy **specjalny tryb pracy** repozytorium.

Już za chwilę...

Podsumowanie

- git clone
- git branch -vv
- git remote show



ZADANIE

- ✓ **Sklonuj** swoje repozytorium do innego katalogu
- ✓ **Poobserwuj relacje** między repozytoriami używając nowopoznanych komend

