Jeśli chcesz…

- **ściągnąć kod z repozytorium**
  a. upewni się że masz aktywne konto w https://portal.plgrid.pl
  b. napisz do kamich@agh.edu.pl żebym mógł Cię dołączyć do zespołu
  c. zainstaluj git'a w wersji 1.8+ (Linux) lub GitExtensions (Windows)
  d. wejdź na stronę https://git.plgrid.pl/projects/MODFEM/repos/modfem2015
  e. wejdź w pozycję menu 

  f. skopiuj adres url, który powinien wyglądać jakoś tak.:
  https://<twój_username>@git.plgrid.pl/scm/modfem/modfem2015.git
  g. zrób
  `git clone skopiowany_url`
  h. w folderze `modfem2015` znajduje się twoje repozytorium git modfem'a
  i. jeżeli interesują Cię stare przykłady, to popatrz na repozytorium
  https://git.plgrid.pl/projects/MODFEM/repos/modfem2015_old_examples

- **przejść na wersje kodu ze stabilnym programem do prowadzenia obliczeń**
  `git checkout master`

- **przejść na najnowszą gałąź kodu**
  `git checkout develop`

- **uaktualnić swoje repozytorium do wersji na serwerze**
  `git pull`

- **rozpocząć pracę nad nową funkcjonalnością w modfem'ie**
  = założyć nową gałąź (brancha)

- **założyć nową gałąź i przejść na nią**
  a. zdecyduj jakiego rodzaju ma to być gałąź
    - nowa funkcjonalność (feature branch)
    - poprawka błędu (hotfix)
  b. przejdź na najnowszą gałąź kodu
  c. uaktualnij swoje repozytroium do wersji na serwerze
  d. załóż gałąź

```
git checkout -b <nazwa nowej gałęzi>
```
albo
```
git branch <nazwa nowej gałęzi>
git checkout <nazwa nowej gałęzi>
```
albo dla brancha, którego od razu chcemy stworzyć na serwerze
```
git branch --set-upstream-to=origin/<nazwa> <nazwa>
git checkout <nazwa>
```

- **wysłać (nową) gałąź na serwer**

```
git checkout <nazwa_gałęzi>
git push origin --set-upstream <nazwa_gałęzi>:<nazwa_gałęzi>
```

- **zobaczyć jakie zmiany zostały zrobione od ostatniego commita w aktualnej gałęzi**
  ```
  git status
  ```

- **dodać pliki**
  ```
  git add <nazwy_plików>
  ```
  (git nie może dodawać pustych folderów)

- **skopiować pliki z innej gałęzi**
  ```
  git checkout <gałąź NA którą chcesz skopiować>
  git checkout <gałąź Z której kopiujesz> ścieżka/do/pliku
  ```
  (działa też dla całych folderów)

- **usunąć pliki**
  ```
  git rm <nazwa_pliku>
  ```

- **przesunąć pliki**
  git mv <nazwa_pliku>

- **zapisać wprowadzone zmiany**
  a. wszystkie zmiany od ostatniego commita
  ```
  git commit -a -m "Jedno zdanie co i jak"
  ```
  b. konkretne pliki
  ```
  git commit <nazwa_pliku> -m "Jedno zdanie co i jak"
  ```

- **wysłać wprowadzone zmiany na serwer**
  git push

- **porównać dwie gałęzie**
  ```
  git diff <gałąź1>..<gałąź2>
  ```
  lub by zobaczyć to samo, ale w odniesieniu do "wspólnego przodka"
  ```
  git diff <gałąź1>...<gałąź2>
  ```

- **wprowadzić do <aktualnej gałęzi> zmiany z <innej gałęzi>**
  = patrz **dołączyć gałąź <nazwa**=inna gałąź**> do gałęzi <nazwa docelowa**=aktualna gałąź**>**

- **dołączyć swoją gałąź <nazwa> do aktualnej wersji kodu (tej testowanej)**
  = dołączyć gałąź <nazwa> do gałęzi *develop*

- **dołączyć gałąź <nazwa> do gałęzi <nazwa docelowa>**
  ```
  git checkout <nazwa>
  ```

```
git pull
git push
git checkout <nazwa_docelowa>
git pull
git merge --no-ff <nazwa>
```
(parametr `--no-ff` jest istotny)
```
git push origin <nazwa_docelowa>
```
miejsce łączenia oznaczamy tagiem, żeby można było w razie czego wrócić do tego co się stało
```
git tag -a branch-<nazwa> -m "Merge <nazwa> into
<nazwa_docelowa>"
git push --tags
```

- **zamknąć gałąź**
```
git branch -d <nazwa>
git push origin :<nazwa>
```

- **tymczasowo schować zmiany, do których chcesz później przywrócić**
```
git stash save
```

- **przywrócić wcześniej schowane zmiany**
```
git stash apply
```