NÁPOVĚDA

D:\Projekt>git help [prikaz]	zobrazí nápovědu pro uvedený příkaz
D:\Projekt>git config --help	vypíše seznam parametrů, které lze zadat do konfiguračního nastavení Git-u

ZÁKLADNÍ NASTAVENÍ GIT

D:\Projekt>git config --global user.name "Jméno Příjmení"	
D:\Projekt>git config --global email@example.com	
D:\Projekt>git config --global core.editor scite	(např. pspad, emacs, vi, notepad, ...)
D:\Projekt>git config --list	vypíše aktuální konfiguraci

ZALOŽENÍ PROJEKTU

Inicializace repozitáře - existující adresář

D:\Projekt>git init	

Klonování - existující repozitář

D:\Projekt>git clone [URL]	
D:\Projekt>git clone [URL] [NazevPracovnihoAdresare]	

PŘIDÁNÍ SOUBORŮ DO REPOZITÁŘE (ZAČÁTEK SLEDOVÁNÍ SOUBORŮ)

D:\Projekt>git add [JmenoSouboru \ JmenoAdresare]	

PŘIDÁNÍ SOUBORŮ DO OBLASTI PŘIPRAVENÝCH ZMĚN

D:\Projekt>git add [JmenoSouboru \ JmenoAdresare]	

ZÁPIS ZMĚN

D:\Projekt>git commit	systém vyzve k zadání komentáře
D:\Projekt>git commit -m "Komentář k revizi"	implicitní zadání komentáře
D:\Projekt>git commit -a -m "Komentář k revizi"	implicitní zadání komentáře a přeskočení oblasti připravených změn (není nutné provádět příkaz git add [JmenoSouboru], commit je automaticky proveden pro každý sledovaný soubor)

IGNOROVÁNÍ SOUBORŮ

Do souboru .gitignore vložit masku pro soubory, které nebudeme chtít sledovat

Příklad:

*.[oa]	ignorovat všechny soubory končící na .o nebo .a
*~	ignorovat všechny soubory končící vlnovkou
# komentář, je automaticky ignorován	
*.a	ignorovat soubory s příponou .a
!lib.a	kromě souboru lib.a
/TODO	ignorovat soubor TODO v kořenovém adresáři, ne v podadresářích
build/	ignorovat všechny soubory v adresáři build/
doc/*.txt	doc/notes.txt ale ne doc/server/notes.txt

Hvězdička (*) zastupuje žádný nebo více znaků, [abc] označuje jakýkoliv znak uvedený v závorkách, otazník (?) označuje jeden znak, znaky v hranatých závorkách oddělené pomlčkou ([0-9]) označují libovolný znak v daném intervalu.

ODSTRANĚNÍ A PŘEJMENOVÁNÍ (PŘESOUVÁNÍ) SOUBORŮ

D:\Projekt>git rm [JmenoSouboru]	odstranění souboru bude připraveno k zapsání, při příštím commit-u bude soubor odstraněn ze sledování a zmizí z pracovního adresáře
D:\Projekt>git rm --cached [JmenoSouboru]	odstranění souboru bude připraveno k zapsání, při příštím commit-u bude soubor odstraněn ze sledování, ale nedojde k jeho odstranění z pracovního adresáře
D:\Projekt>git mv [PuvodniJmenoSouboru] [NoveJmenoSouboru]	přejmenování souboru

RUŠENÍ ZMĚN

D:\Projekt>git commit --amend	umožní provést změny v poslední revizi (např. opravení komentáře, přidání zapomenutého souboru apod.)

Příklad: třetí příkaz nahradí výsledek prvního příkazu

```
D:\Projekt>git commit -m "První revize"
D:\Projekt>git add [JmenoZapomenutehoSouboru]
D:\Projekt>git commit --amend
```

D:\Projekt>git reset HEAD [JmenoSouboru]	vyjme soubor z oblasti připravených změn
D:\Projekt>git checkout -- [JmenoSouboru]	zahození změn v souboru a navrácení do podoby v poslední revizi. **Pozor, změna je nevratná!!**

KONTROLA STAVU PROJEKTU

D:\Projekt>git status	

D:\Projekt>git diff	zobrazí změny, které ještě nebyly připraveny k zapsání (srovná pracovní adresář s oblastí připravených změn)
D:\Projekt>git diff --staged (starší příkaz je git diff --cached)	zobrazí změny, které již byly připraveny k zapsání a které budou obsahem příští revize (srovná připravené změny s poslední revizí)

D:\Projekt>git log	zobrazení historie revizí
D:\Projekt>git log -p	zobrazení historie revizí i s rozdíly (diff)
D:\Projekt>git log -3	zobrazení historie posledních tří revizí
D:\Projekt>git log --stat	zobrazí statistiku změn
D:\Projekt>git log --pretty=[oneline,short,full,fuller]	zobrazí historii změn v různých formátech a s různou úrovní detailů

VZDÁLENÉ REPOZITÁŘE

D:\Projekt>git remote	systém vypíše názvy vzdálených serverů, které jsou nakonfigurovány
D:\Projekt>git remote -v	zobrazí se názvy vzdálených serverů i s URL adresami
D:\Projekt>git remote add [ZkracenyNazevVzdalenehoRepozitare] [URL]	přidá nový vzdálený repozitář, při další práci se lze odvolávat na název repozitáře místo na URL
D:\Projekt>git remote show [NazevVzdalenehoRepozitare]	zobrazí informace o vzdáleném repozitáři

D:\Projekt>git fetch [NazevVzdalenehoRepozitare]	vyzvednutí všech dat ze vzdáleného repozitáře, která ještě nemáme (tj. např. od posledního klonování, clone). Nedochází k automatickému sloučení s lokální kopií

D:\Projekt>git pull [NazevVzdalenehoRepozitáře]	automatické vyzvednutí a sloučení dat ze vzdálené větve do aktuální pracovní větve

D:\Projekt>git push [NazevVzdalenehoRepozitare] [NazevVetve]	odesle na vzdaleny server obsah pracovniho adresare (revizi). Predpokladem je opravneni zapisovat do vzdaleneho repozitare

ZNAČKY (TAGS)

Git používá dva druhy značek, *prosté* (lightweight) a *anotované* (annotated). Prostá značka je pouze ukazatel na konkrétní revizi, anotovaná značka je ukládána jako plnohodnotný objekt v databázi Git (obsahuje kontrolní součet, jméno autora, poznámku apod.) .

D:\Projekt>git tag	výpis značek v repozitory (v abecedním pořadí). Lze zadat masku pro vyhledání pouze určitých značek

Příklad:

D:\Projekt>git tag -l "2.0.*"	vypíše všechny značky začínající na "2.0."

D:\Projekt>git tag v2.0	vytvoření **prosté** značky
D:\Projekt>git tag -a v2.0 -m "Poznamka k verzi 2.0"	vytvoření **anotované** značky
D:\Projekt>git tag v2.0 9cedh2c	dodatečné doplnění značky do konkrétní revize, určené doplněním kontrolního součtu dané revize (nebo jeho části) na konec příkazu
D:\Projekt>git push [Vetev] [NazevZnacky]	přenesení značky do vzdáleného repozitáře (Git nepřenáší značky automaticky)
D:\Projekt>git push --tags	na vzdálený server se přenesou hromadně všechny značky

D:\Projekt>git describe [NavevVetve]	lze použít pro vygenerování „čitelného“ jména revize. Příkaz vrátí název nejbližší značky + počet revizí (commitů) vykonaných od této značky + část kontrolního součtu v aktuální větvi (příp. ve větvi [NazevVetve])

VĚTVE (BRANCHES)

Vytváření větví

D:\Projekt>git branch [NazevVetve]	vytvoří novou větev v projektu
D:\Projekt>git checkout [NazevVetve]	přepnutí se do existující větve projektu
D:\Projekt>git checkout -b [NazevVetve]	vytvoří novou větev a zároveň se do ní přepne

Slučování větví

D:\Projekt>git merge [NazevVetve]	začlenění (sloučení) větve do jiné větve

Příklad:

D:\Projekt>git checkout master	přepnutí se do větve *master*
D:\Projekt>git merge oprava	začlenění větve *oprava* do větve *master*

D:\Projekt>git branch -d [NazevVetve]	smazání již nepotřebné větve

Dojde-li při slučování ke konfliktu, příkazem "git status" lze zjistit, které části větví nebyly sloučeny. V případě výskytu konfliktu při slučování je potřeba na každý soubor, kterého se dotkla úprava kvůli vyřešení konfliktu, aplikovat příkaz "git add [NazevSouboru]". Systému tak dáme na vědomí, že konflikt byl vyřešen.

Přeskládání

D:\Projekt>git rebase [NazevVetve]	všechny změny, provedené v jedné větvi se provedou na větvi jiné. Jedná se v podstatě o linearizaci vývoje

Příklad:

D:\Projekt>git checkout oprava	přepnutí se do větve *oprava*
D:\Projekt>git rebase master	provede změny, které byly udělány ve větvi *oprava*, na větvi *master*

Zobrazení a správa větví

D:\Projekt>git branch	výpis aktuálních větví (větev označená '*' je větev, na které se aktuálně nacházíme).
D:\Projekt>git branch -v	zobrazí poslední revize na každé větvi
D:\Projekt>git branch --merged	zobrazí větve, které již byly začleněny do větve, na které se aktuálně nacházíme
D:\Projekt>git branch --no-merged	zobrazí větve, které doposud nebyly nebyly začleněny do větve, na které se právě nacházíme

Vzdálené větve

D:\Projekt>git fetch origin	aktualizuje lokální kopii databáze ze vzdáleného serveru
D:\Projekt>git push [NazevVzdalenehoServeru] [NazevVetve]	odeslání větve na vzdálený server (je nutné mít práva pro zápis)