



Workshop about Git SCM v. 2024

Lukáš Kotecký

Senior Software Quality Engineer

✉ lkotek@redhat.com

🌐 github.com/lkotek

O čem to dnes bude?



1. K čemu je git a jeho konfigurace

- a. K čemu slouží git a jeho **základní principy**
- b. **Prerekvizity** a nastavení gitu pro **první použití**

2. První commit a pull request

- a. Začínáme aneb **repozitář, branch a commit**
- b. **Klonujeme repozitář** a tvoříme **první commit!**

3. Verzováním to nekončí!

- a. Jak spolu souvisí **git a CI**
- b. **Github Actions**



Kdo jsem

- **Senior Software Quality Engineer** ve firmě [Red Hat](#)
- Člen **KVM QE** týmu zaměřeného na **virtualizaci**
 - **automatizované testování** [virtualizace](#) ([KVM](#)/[QEMU](#))
 - **konfiguraci CI** a úpravu s tím spojených nástrojů



Řeším, co všechno se musí stát, když vyjde nová verze balíku X pro operační systém Y, aby se spustila sada automatizovaných testů Z to ve správné konfiguraci :)

- **Nadšenec do Linuxu** už od střední školy
- **Fanoušek elektroniky** a nějakého toho domácího pájení
- [Příležitostný](#) autor [článků](#) a přednášející na konferencích
- Bývalý **aplikační admin bankovních IT** systémů a středoškolský učitel



Red Hat

- Firma **založena v roce 1993** v Raleigh
- ~20 tisíc zaměstnanců
- Produkty:
 - **Red Hat Enterprise Linux** ([RHEL](#))
 - Red Hat **OpenShift** (kontejnerizace)
 - Red Hat **Ansible** Automation Platform a mnoho [dalších](#)
- V roce 2019 [odkoupena](#) firmou [IBM](#) za 34 miliard USD
- v ČR od roku 2004, **pobočka v Brně** (od 2006)
 - největší pobočka v rámci Red Hatu zaměřená na **vývoj software**
 - ~1700 zaměstnanců



Staré logo – školní návštěva firmy v roce 2016



Cíl workshopu



Verzování je věc, se kterou se setkáte v **každé softwarové firmě** (a zdaleka nejen tam), ať už jste vývojáři, administrátoři nebo budete třeba “jen” upravovat dokumentaci.

- **Co je cílem workshopu?**

- **Situace:** Chci *přispět do existujícího* projektu
(nebo vytvořit *svůj vlastní* gitový projekt)
 - Co k tomu potřebuji? ⇒ **základy práce s gitem**
 - Jak svoje změny mohu začlenit? ⇒ **vytvoření pull/merge* requestu**
 - A co automatizace spojená s gitem? ⇒ **propojení s dalšími nástroji**

- **Co není cílem workshopu?**

- objasnit, jak přesně git funguje uvnitř a detailní zvládnutí práce s ním
 - sám se i po letech stále učím :)

* Výraz “pull request” je poplatný GitHubu, můžete se setkat také se spojením “merge request”

K čemu slouží verzovací systémy



Verzovací systémy (Source Code Management → **SCM**) se používají pro verzování změn zdrojového kódu, typicky pro **software**, ale nejen tam; např. u **dokumentace** a pro **konfigurační** soubory (configuration management).

- **Spolupráce více lidí na jednom projektu**
 - ⇒ **kdo** a **kdy** provedl **jaké** změny (+ jejich snadné revertování)
 - ⇒ práce na vlastní větvi před začleněním změn
 - ⇒ nutnost sloučit různé (i konfliktní) změny různých přispěvatelů
- **Identifikace toho jaká změna byla začleněna a kdy**
 - ⇒ jaká funkcionálita je součástí které verze vyvíjeného projektu
 - ⇒ proč nefunguje X po vydání verze Y
- **Navázání dalšího workflow při (nebo před) publikování změn**

K čemu slouží verzovací systémy



```
lkotek@t14:~/git/tp-qemu — git log
commit 2a4b5261ea51706b05a9c518a56ab896323b4fe4 (HEAD -> ensure-sshpas...
for-ansible, origin/ensure-sshpas-installs-for-ansible)
Author: Lukas Kotek <lkotek@redhat.com>
Date:   Fri Dec 2 15:19:40 2022 +0100

    ansible: add checks for additional dependencies like `sshpas`

    In case ansible is installed via pip before avocado based ansible tests
    are executed, some key dependencies don't have to be satisfied - the most
    important `sshpas`. This commit adds check for `sshpas` command and
    ensures package providing this command is installed.

    Signed-off-by: Lukas Kotek <lkotek@redhat.com>

commit 04529c20fd40b188b3e1893cf7bc7505731d9a7b (upstream/master, master)
Merge: 2de951c3 e8bab627
Author: YongxueHong <yhong@redhat.com>
Date:   Fri Dec 2 13:54:18 2022 +0800

    Merge pull request #3542 from qingwangrh/2141964_fix_serial_generator

    physical_resources_check:fix serial generator

commit 2de951c367b4f9020143efd4848e6ff99171d5d1
Merge: 27883294 afad0844
Author: YongxueHong <yhong@redhat.com>
Date:   Fri Dec 2 11:14:20 2022 +0800

    Merge pull request #3552 from qingwangrh/2149204_enhance_qsd

    qsd:enhance add_vubp_into_boot and set default image format
```

← Log s **commity** v repozitáři tp-qemu (testy)

↓ Automatické **testy pro pull request** (GitHub)

ansible: update provider with optional custom dnf repository #3550

Merged yanan-fu merged 1 commit into autotest:master from lkotek:ansible-installation-repo last week

Conversation 17 Commits 1 Checks 2 Files changed 3

ansible: update provider with optional custom dnf repository 5f24517

CI on: pull_request 2

Python 3.6 succeeded 3 weeks ago in 1m 27s

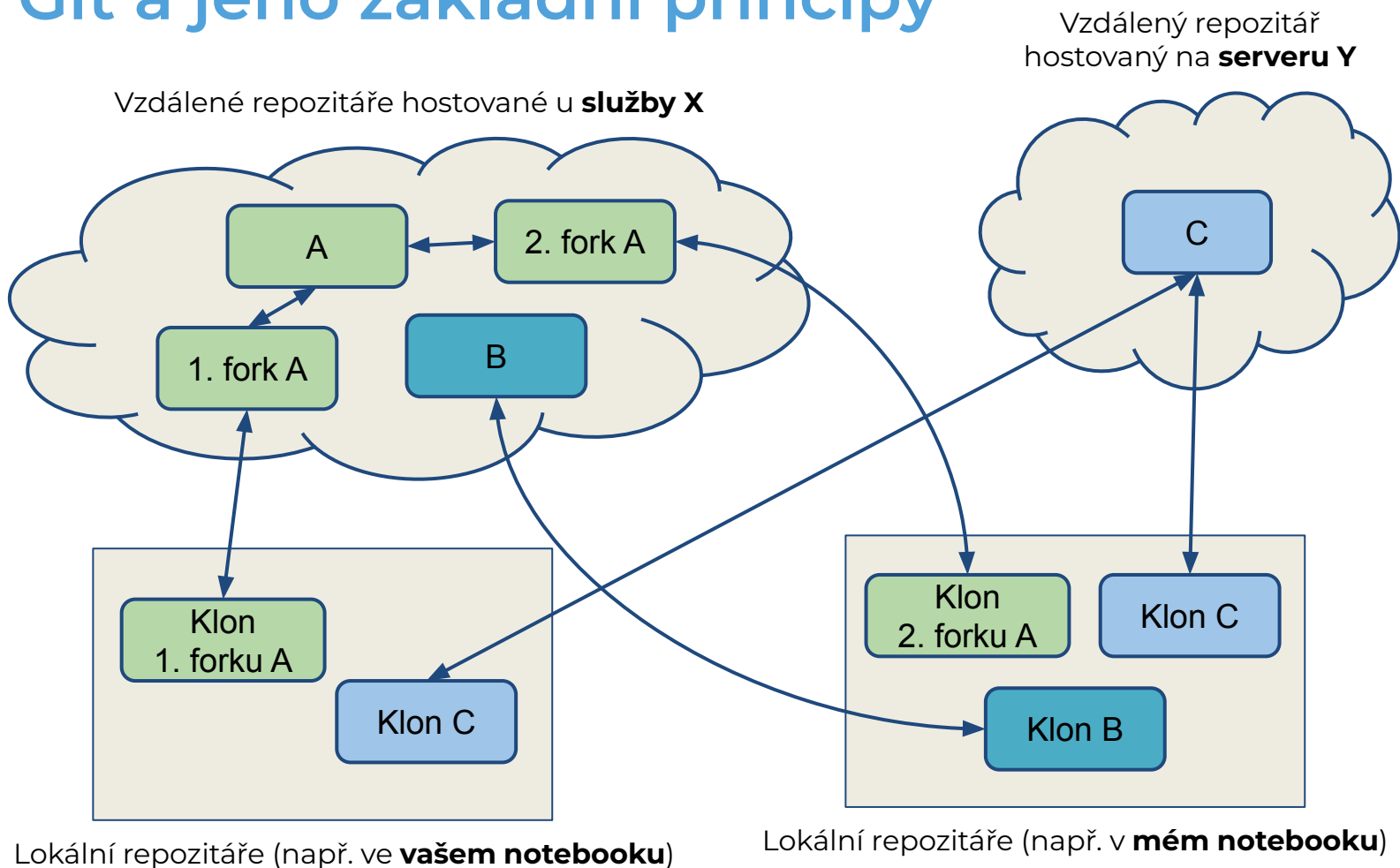
- Set up job
- Run echo "Job triggered by a pull_request event on branch is refs/pull/3550/merge in repository"
- Check out repository code
- Set up Python 3.6
- Install dependencies
- Run inspekt
- Run echo "This job's status is success."
- Post Set up Python 3.6
- Post Check out repository code
- Complete job

Git a jeho základní principy



- Jde o Open Source **distribučovaný verzovací systém**
 - **open source** ⇒ git je software šířený pod licencí **GNU GPL**, lze ho volně používat, upravovat a šířit
 - **distribučovaný** ⇒ každý vývojář může mít pracovat lokálně na *rovnocenné kopii* projektu obsahující veškeré commity a větve
- Autorem je **Linus Torvalds** (vyvinul a použil git pro Linux, viz kernel.org)
- Vhodný primárně pro obsah ve formě **textu**
- Práce s gitem probíhá v terminálu (ale lze jej integrovat do většiny IDE)
- Repozitáře mohou být **lokální** i **vzdálené**
 - GitHub a GitLab jsou příklady služeb poskytujících **vzdálené repozitáře** na bázi gitu (a mnoho další navázané funkcionality)
 - Lokální repozitář, tzn. vytvořený např. na vašem notebooku

Git a jeho základní principy



Prerekvizity pro workshop



- **Nainstalovaný git** a zřízený účet na <http://github.com> (registrace i používání pro Open Source projekty je bezplatné).
- Pro práci je doporučen notebook s konektivitou do internetu s **nainstalovaným Linuxem** (Fedora Workstation)
- Optimálně je vhodné na daném notebooku vygenerovat **pár ssh klíčů**** (viz návod v [článku](#) na root.cz) a **veřejný klíč** nahrát přes webové rozhraní do GitHubu.
 - Na Linuxu v souboru `~/.ssh/[vas_klic].pub`
 - Zpříjemní to práci s gitem (např. není nutné zadávat heslo při pushování změn na vzdálený repozitář).

Generujeme ssh klíče na Fedora Workstation

- Máme nainstalovaný balík **openssh**?
 - V terminálu (bash) zadáme příkaz **ssh-keygen**:
 - Zobrazí se příkaz nebyl nalezen?
 - Pokud není nainstalovaný, zadáme (jako root):
rpm -q openssh
dnf install openssh
- Je v adresáři **.ssh** přítomen soubor s příponou **.pub**? Zobrazíme ho:
cat .ssh/id_ed25519.pub

Nastavení gitu pro první použití

- Globálně platná nastavení pro **všechny vaše lokální repozitáře**
 - Vaše změny (commity) budou identifikovány zadanými údaji:
git config --global user.name "Vase Jmeno"
git config --global user.email "vasejmeno@domena.cz"
- Volitelně lze také uložit heslo (doporučuji raději použít ssh klíč):
git config --global user.password [vase_heslo]
- Záznam se na Linuxu zapíše do souboru **~/.gitconfig**
 - skrytý soubor v domovském adresáři uživatele

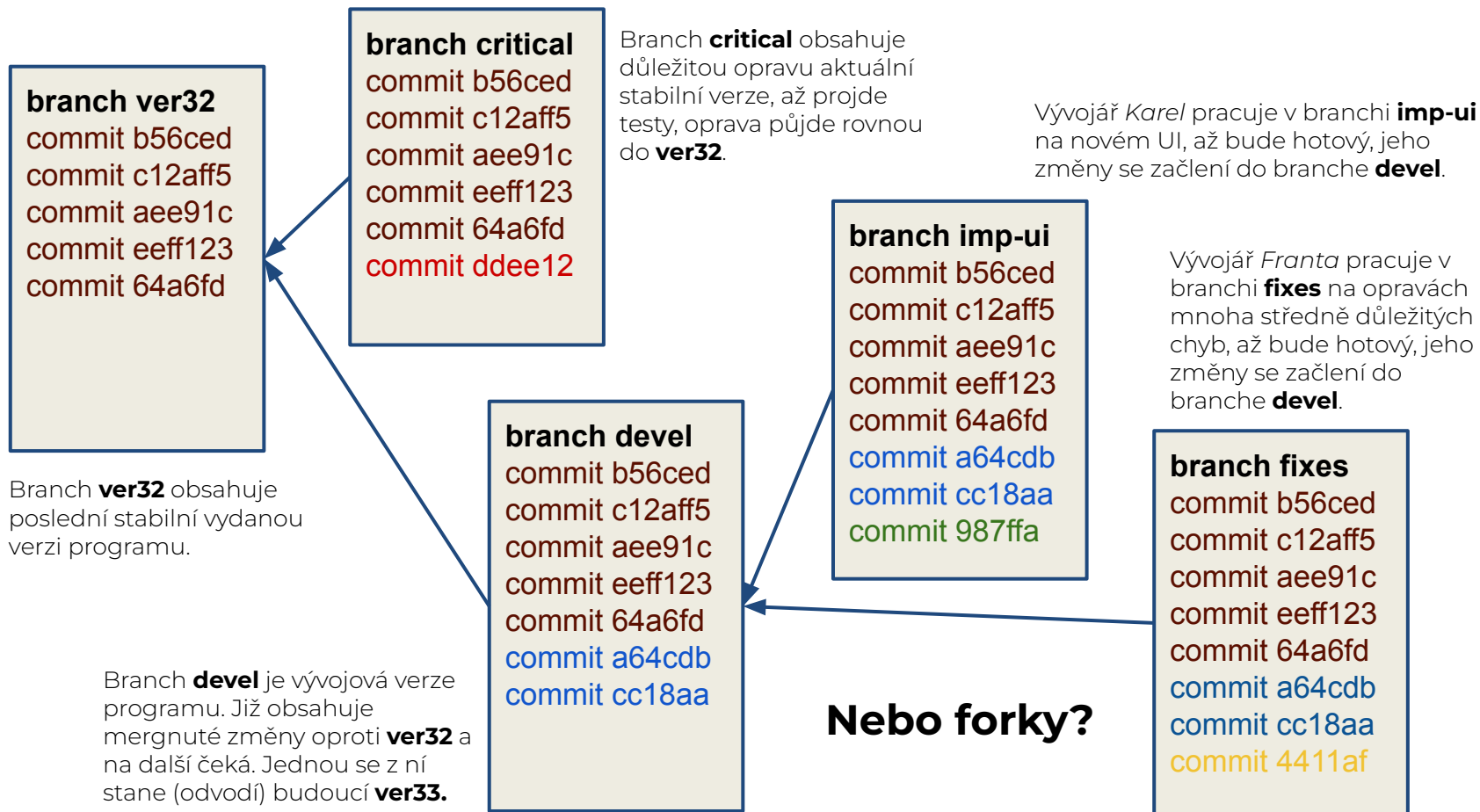
Repozitář, branch a commit



- **repository** = místo, které obsahuje samotný projekt
 - Celou **historii projektu** (větve, commity) od *počátku* věků :)
 - změny mezi vzdáleným a lokálním repozitářem lze snadno synchronizovat
- **fork** = duplikát existujícího repozitáře
- **branch** = větev, obsahující stav projektu, např.:
 - master/main → pro průběžné začleňování změn
 - devel → vývojová větev připravovaná na vydání
 - Jde o příklady ⇒ názvy mohou být libovolné
- **merge** = proces zapsání změn z jedné větve do druhé
- **commit** = jednotlivá zapsaná změna

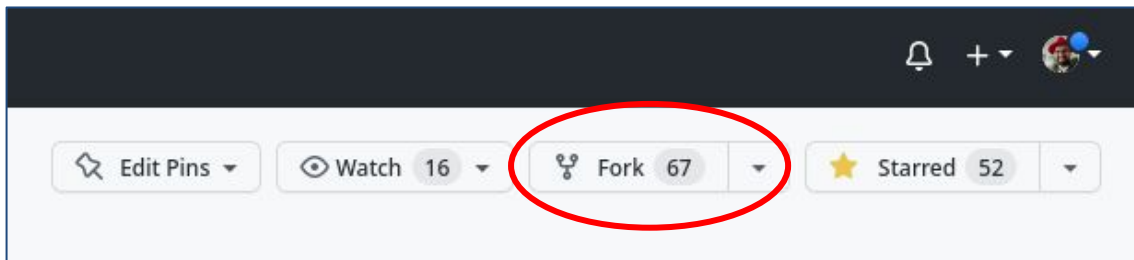
Repozitář, branch a commit

Jeden repozitář?



Vytváříme fork projektu

- Na svých změnách typicky pracujeme v rámci **forku** repozitáře
 - (dalším možným přístupem je **vlastní branch** v rámci **původního repozitáře**, oba přístupy se hodí v jiných situacích)
- Náš pracovní projekt: <https://github.com/lkotek/trebesin-workshop/>



Klonujeme repozitář

- Pracovní repozitář s vaším **forkem***:
 - [https://github.com/\[account\]/trebesin-workshop/](https://github.com/[account]/trebesin-workshop/)
 - Pokud máte vygenerovaný pár ssh klíčů, naklonujeme pomocí:
git clone git@github.com:[account]/trebesin-workshop.git
 - Pokud ne, nevadí, provedeme pomocí:
git clone https://github.com/[account]/trebesin-workshop.git

Vzdálené repozitáře

- Jak pracovat s původním repozitářem (upstream) a forkem (origin)?*
 - S jakým vzdálenými repozitáři vlastně pracujete:
git remote -v
 - Další repozitář přidáme pomocí:
git remote add [navez] [url_repozitáře]
 - Nebo odebere:
git remote remove [navez]

* Ve výsledku můžete jednoduše **synchronizovat změny** z upstream repozitáře do vašeho forku.

Tvoříme první commit

1. Vytvořte novou branch založenou na stávající pomocí ⇒ **git checkout -b devel**
2. Vytvořte **nový textový soubor** soubor a napište do něj jméno vašeho oblíbeného programovacího jazyka a **změny uložte**. Poté se použije příkaz ⇒ **git diff**
3. Proveďte “stage” změn ⇒ **git add [název souboru]**
4. Vytvořte **commit** (zapište změny do repozitáře) ⇒ **git commit -s**
5. Odešlete změny do **vzdáleného repozitáře** ⇒ **git push**
+ zkontrolujte obsah repozitáře

modified	staged	committed
změny, o kterých git “neví”	změny připraveny k zapsání	změny zapsány
git diff	git add [název souboru]	git commit [-s -m]

Vytváříme pull/merge request

The screenshot shows a GitHub Pull Request page. At the top, the 'Pull requests' tab is highlighted with a yellow circle. Below it, the 'Comparing changes' section shows the 'base repository: autotest/tp-qemu' and 'head repository: lkotek/tp-qemu' selected, both circled in red. A red cloud annotation 'Odkud chceme vzít změny' (Where do we want to take the changes) points to the head repository. Another red cloud annotation 'Kam je chceme začlenit' (Where do we want to include it) points to the base repository. The 'compare' dropdown is set to 'ensure-sshpas-installs-for-a...'. A green button 'View pull request' is visible. Below the comparison, a commit message 'ansible: add checks for additional dependencies like `sshpas` #3557' is shown, along with a diff view of the file 'provider/ansible.py'.

Odkud chceme vzít změny

Kam je chceme začlenit

Comparing changes

Choose two branches to see what's changed or to start a new pull request. If you need to, you can also create a new branch and commit to it.

base repository: autotest/tp-qemu base: master head repository: lkotek/tp-qemu compare: ensure-sshpas-installs-for-a...

✓ Able to merge. These branches can be automatically merged.

Checks for additional dependencies like `sshpas` #3557

Installed via pip before avocado based ansible tests are executed, some key dependencies don't have to be satisfied – the most im...

View pull request

1 commit 1 file changed 1 contributor

Commits on Dec 2, 2022

ansible: add checks for additional dependencies like `sshpas`

lkotek committed 2 weeks ago ✓

Showing 1 changed file with 14 additions and 7 deletions.

Split Unified

provider/ansible.py

```
@@ -154,10 +154,17 @@ def distro_install():
    policy_map = {"distro_install": distro_install,
                  "python_install": python_install}
```

Verzováním to nekončí!

- Na zapsání commitu lze automatizovaně navázat mnoho **dalších činností**, např.:
 - Kontrolu vašich změn v kódu (např. [pylint](#))
 - Nainstalujeme ho pomocí:

```
pip install pylint --user  
dnf install pylint
```
 - Nástroje pro **Continuous Integration** (CI)
 - **Build nové verze** aplikace
⇒ včetně verze zahrnující změny z vašeho commitu
 - Spuštění **testování nové verze** aplikace



GitHub Actions



- Spust' **vzdáleně*** nějakou akci např. **při commitu**
 - Podívejte se na **YAML soubor s konfigurací**:
trebesin-workshop/.github/workflows/pylint.yaml
- Spustme **pylint** pro skripty v pythonu, viz adresář **src**
- Jak se to celé konfiguruje a jak to funguje?
 - [Začíné s GitHub Actions](#) | [Co jsou to hosted runners?](#)
 - [K čemu slouží pylint?](#) | [Na co si stěžuje pylint](#) :)

* Naopak pro **lokální spuštění** akcí (e.g., test kódu) existuje výborný nástroj [pre-commit](#).

Užitečné příkazy

Jeden odkaz za všechny!
<https://ohshitgit.com/>

- **git init** → založ v aktuálním (nebo zadaném) adresáři nový gitový repozitář
- **git checkout [název_branche]** → přepni se na branch ⇒ s přepínačem **-b** dojde k vytvoření nové branch založené na základě branch aktuální
- **git branch** → ukáže, na které se aktuálně nacházíme větví
- **git status** → ukaž stav stromu (branch? je něco k zapsání? jaké soubory jsou změněny?)
- **git reset [commit_id]** → vyresetuj do stavu dle zadaného commitu
- **git stash** → zahod' provedené změny v pracovním adresáři
- **git merge [název_branche]** → sloučení dvou větví dohromady
- **git log** → ukaž historii (commity) v dané branchi
- **git pull** → stáhni a začleň změny ze vzdáleného do lokálního repozitáře

Kam se podívat dál

- **Kam se podívat na úvod**

[Git SCM](#) | [Git SCM – kniha v češtině](#) | [Git Cheat Sheet](#)
[Seriál o gitu na root.cz](#) | [Kurz od PyLadies](#) | [Tahák](#)

- **Ke stažení**

[GetFedora.org](#) | [Git pro MS Windows](#)

- **Zajímavé nástroje**

[GitHub Actions](#) | [pre-commit](#)

- **Když to chceme pojmut hravě**

[Oh Shit, Git!?! Git Game](#) | [LearnGitBranching](#) (vizualizace)

- **A pojďme ještě dál!**

[GitHub a CI](#) | [GitLab CI](#) | [pre-commit](#) (otestuj, až potom pošli)



Děkuji za pozornost!

QA