

Git – Nastavení porovnávání & slučování přes Meld

 Praktické rady pro konfiguraci nástroje Meld jako diff/merge tool v Git na Windows i Linuxu.



Co je Meld?

- **Meld** je vizuální nástroj pro porovnávání a slučování souborů.
 - Umožňuje přehledné zobrazení rozdílů a snadné řešení konfliktů.
-

Nastavení Meld jako diff/merge tool

- ▶  [Windows – Kompletní postup](#)
 - ▶  [Linux – Kompletní postup](#)
-

Použití v praxi

- ▶  [Porovnání změn](#)
- ▶  [Řešení konfliktů při slučování](#)



Git – Práce s úložištěm



Praktické rady pro vytvoření a použití Git úložiště na lokálním i online prostředí.



Vytvoření úložiště

►  [Kompletní postup](#)



Klonování úložiště

►  [Použití v pracovním prostředí](#)



Git – Submoduly: Přehled & použití



Praktické rady pro správu externích repozitářů pomocí submodulů v Git.



Co jsou submoduly?

- Umožňují vložit jeden Git repozitář do jiného jako podadresář.
 - Zachovávají nezávislost obou repozitářů.
 - Vhodné pro správu závislostí, sdílený kód nebo více projektů najednou.
-






Struktura projektu se submodulem

- ▶  Ukázková struktura
-



Základní příkazy

- ▶  Přidání submodulu
 - ▶  Klonování projektu se submodule
 - ▶  Aktualizace submodulů
-






Praktický příklad v Unity projektu

- ▶  Struktura & přidání submodulů
-





Tipy pro práci se submoduly

- ▶  Přepínání verzí submodulu
 - ▶  Úpravy v submodulu
 - ▶  Odstranění submodulu
-



Řešení častých problémů

- ▶  Submodul v "detached HEAD" stavu
- ▶  Submodul ukazuje změny i když žádné nemáte

- ▶  Změna URL submodulu
 - ▶  Kontrola stavu submodulů
-

Výhody & nevýhody submodulů

- ▶  Výhody
- ▶  Nevýhody



Git Flow – Strategie větvení & workflow



Praktické rady pro efektivní správu větví v týmu pomocí Git Flow.





Co je Git Flow?

- **Git Flow** je osvědčená strategie pro řízení verzí a vývoj v týmech.
 - Umožňuje jasně oddělit vývoj, přípravu vydání a opravy chyb.
-






Hlavní větve

- ▶  Základní větve
 - ▶  Pomocné větve
-



Typické workflow

- ▶  Vývoj nové funkce
 - ▶  Příprava vydání
 - ▶  Oprava chyby v produkci
-



Pravidla pro práci s Git Flow

- ▶  Doporučené postupy
-



Vizualizace workflow

- ▶  Schéma větvení



Git – Vytvoření & push nové větve na remote

🚀 Praktické rady pro založení a umístění nové větve (`develop`) na Git server (např. GitHub, GitLab).



Vytvoření nové větve

- ▶ 🛠️ [Krok 1: Založení větve](#)
-



Nastavení vzdáleného repozitáře

- ▶ 🔗 [Krok 2: Ověření remote](#)
-



Push větve na server

- ▶ 📦 [Krok 3: Push větve na remote](#)
-



Ověření online

- ▶ 🔍 [Krok 4: Kontrola na webu](#)



Git – Smazání vzdálené větve



Praktické rady pro bezpečné odstranění větve z Git serveru (např. GitHub, GitLab).



Upozornění



Warning

Smazání vzdálené větve je **nevratná operace**.

Ujisti se, že větev už nepotřebuješ a všechny důležité změny jsou začleněny jinde.



Postup krok za krokem

- ▶ 🔍 Krok 1: Zobrazení všech větví
- ▶ 🗑️ Krok 2: Smazání vzdálené větve
- ▶ 🧹 Krok 3: Vyčištění lokálních referencí



Git – Pull Request (PR)



Praktické rady, jak funguje **Pull Request**, kdy a proč ho použít.







Co je Pull Request?

- **Pull Request (PR)** je žádost o začlenění změn z jedné větve do jiné (typicky z `feature` do `main` nebo `develop`).
 - Umožňuje týmovou kontrolu, diskusi, testování a schválení změn před sloučením.
-



Typický workflow PR

- ▶  Krok 1: Vytvoření nové větve
 - ▶  Krok 2: Vytvoření PR na serveru
 - ▶  Krok 3: Code review & testy
 - ▶  Krok 4: Schválení & merge PR
-



Výhody Pull Requestu

- ▶  Proč používat PR?



Git – Aktualizace `.gitignore` & odstranění mezipaměti



Praktické rady, jak správně aktualizovat `.gitignore` a odstranit již sledované soubory z Git mezipaměti.






Proč aktualizovat `.gitignore`?

- `.gitignore` určuje, které soubory Git nemá sledovat.
 - Po změně je nutné odstranit již sledované soubory z mezipaměti, aby se ignorovaly.
-






Postup krok za krokem

- ▶  Krok 1: Odstranění mezipaměti sledovaných souborů
- ▶  Krok 2: Přidání změn do stage
- ▶  Krok 3: Commit změn




Git – Přesun commitů do nové nebo existující větve

 Praktické rady, jak přesunout poslední commity ze jedné větve do nové nebo existující větve.

Přesun commitů do nové větve

- ▶  Krok 1: Vytvoření nové větve z aktuální
- ▶  Krok 2: Odstranění commitů ze zdrojové větve
- ▶  Krok 3: Přepnutí do nové větve

Přesun commitů do existující větve

- ▶  Krok 1: Merge commitů do cílové větve
- ▶  Krok 2: Odstranění commitů ze zdrojové větve
- ▶  Krok 3: Přepnutí do cílové větve

Note

Více informací najdeš v [diskuzi na Stack Overflow](#).



Git – **fixup!** & **squash!** commity

🚀 Praktické rady, jak efektivně opravovat a slučovat commity pomocí **fixup!** a **squash!** v Gitu.



Co znamená **fixup!** a **squash!**?

- **fixup!** – vytvoří commit, který opravuje předchozí commit bez změny jeho zprávy.
 - **squash!** – vytvoří commit, který sloučí zprávu s původním commitem.
-



Postup krok za krokem

- ▶ 🔧 Krok 1: Vytvoření opravného commitu
- ▶ ↺ Krok 2: Rebase s automatickým sloučením
- ▶ ✎ Krok 3: Úprava v editoru







Git – Nahrazení vzdálené větve z lokální větve

 Praktický návod, jak kompletně nahradit historii vzdálené větve pomocí nové lokální větve.

Kdy použít tento postup?

- Chceš začít s čistou historií commitů (např. po refaktoringu).
 - Potřebuješ odstranit veškerou předchozí historii z hlavní větve (`main` / `master`).
 - Vhodné pro projekty, kde je nutné kompletní "reset" repozitáře.
-

Postup krok za krokem

- ▶  Krok 1: Vytvoření nové větve bez historie
- ▶  Krok 2: Přidání všech souborů
- ▶  Krok 3: První commit
- ▶  Krok 4: Smazání původní hlavní větve
- ▶  Krok 5: Přejmenování nové větve na hlavní
- ▶  Krok 6: Force push do vzdáleného repozitáře