進階 Git 指令

Git 就像開發者的瑞士軍刀——多功能、強大，但如果你不知道要使用哪個工具，可能會讓人感到困惑。今天，我們將深入探討 Git 的一些最佳功能和工作流程：精選變更、優雅合併、重新基礎以獲得更清晰的歷史記錄、刪除那些你意外提交的大文件，以及在你意識到自己偏離軌道時撤銷提交。讓我們來分解這些內容。

精選變更：只取你需要的　想象一下，你有一個包含十幾個提交的功能分支，但其中有一個你想要提取並應用到主分支的提交。這時候 `git cherry-pick` 就派上用場了。

它非常簡單：找到提交哈希（你可以從 `git log` 中獲取），切換到你想要應用它的分支，然後運行：

```
git cherry-pick <commit-hash>
```

完成，這個提交現在是你當前分支的一部分。如果有衝突，Git 會暫停並讓你解決它，就像合併一樣。解決完後，提交變更，你就完成了。

我經常使用這個功能，當一個修復漏洞的提交混入一個混亂的功能分支時，我需要立即將它應用到 `main` 分支。但要小心——精選變更會複製提交，所以它會獲得一個新的哈希。如果你後來合併原始分支而不進行一些清理，不要期望它會很好地運行。

合併選項：不僅僅是「合併」　合併是 Git 的基本功能，但你知道它有不同的風味嗎？默認的 `git merge` 如果可能會進行「快進」（直線化歷史記錄），或者如果分支已經分歧，則創建一個合併提交。但你有選擇：

- `--no-ff`**（無快進）**：即使快進可能，也強制創建一個合併提交。我喜歡這個來保持一個清晰的記錄，當一個功能分支進入 `main`。運行它：

  ```
  git merge --no-ff feature-branch
  ```

- `--squash`：將分支中的所有變更拉入當前分支的一個提交中。沒有合併提交，只是一個整潔的包裹。完美地將一個混亂的分支壓縮成一個可展示的：

  ```
  git merge --squash feature-branch
  ```

  完成後，你需要手動提交以完成。

每個都有其用途。我傾向於對長期分支使用 `--no-ff`，對於充滿「WIP」提交的分支，我更喜歡 `--squash`。

重新基礎：像專業人士一樣重寫歷史　如果合併看起來太混亂，`git rebase` 可能是你的風格。它將你的提交取出並重新應用到另一個分支上，給你一個線性的歷史記錄，看起來像你從一開始就計劃好了。

切換到你的功能分支並運行：

```
git rebase main
```

Git 會將你的提交取出，更新分支以匹配 `main`，然後將你的變更重新應用到頂部。如果出現衝突，解決它們，然後 `git rebase --continue` 直到完成。

優點？一個完美的時間線。缺點？如果你已經推送了該分支，並且其他人正在使用它，重新基礎會重寫歷史記錄——這會引起同事的憤怒電子郵件。我只在本地分支或獨自工作的項目中進行重新基礎。對於共享的內容，合併更安全。

從歷史中刪除大文件：哦，那個 2GB 的視頻　我們都經歷過：你意外地提交了一個巨大的文件，推送它，現在你的倉庫變得臃腫。Git 不會輕易忘記，但你可以通過一些努力將該文件從歷史記錄中刪除。

這裡的首選工具是 `git filter-branch` 或更新的 `git filter-repo`（我推薦後者——它更快且更少錯誤）。假設你提交了 `bigfile.zip` 並需要將其刪除：1. 安裝 `git-filter-repo`（檢查其文檔以進行設置）。2. 運行：`git filter-repo --path bigfile.zip --invert-paths` 這將從歷史記錄中的每個提交中刪除 `bigfile.zip`。3. 強制推送重寫的歷史記錄：`git push --force`

注意：這會重寫歷史記錄，所以要與你的團隊協調。如果它在某個拉取請求中，你可能需要清理引用。完成後，你的倉庫會在垃圾收集後變得更小（`git gc`）。

撤銷提交：倒退時鐘　提交後立即後悔嗎？Git 會幫助你。根據你走得有多遠，有幾種方法可以撤銷它：

- **如果你還沒有推送**：使用 `git reset`。要撤銷最後一個提交但保留工作目錄中的變更：

  ```
  git reset HEAD^ --soft
  ```

  想完全丟棄變更？

  ```
  git reset HEAD^ --hard
  ```

- **如果你已經推送**：你需要重寫歷史記錄。使用 `git reset HEAD^` 重置本地，然後強制推送：

  ```
  git push --force
  ```

  這會影響共享歷史記錄，所以要小心。

我用 `git reset --soft` 幫助自己更多次——完美地當你提交得太早並需要調整某些內容。

總結　Git 的靈活性使其如此強大，但如果你不知道選項，很容易變得混亂。精選變更以獲得外科手術精度，調整合併以適應你的工作流程，重新基礎以獲得光滑的歷史記錄，並不要慌張當你需要擦除錯誤——無論是巨大的文件還是匆忙的提交。如果你感到緊張，可以在測試倉庫上練習，很快它們就會像第二本性一樣。

你最喜歡的 Git 技巧是什麼？讓我知道——我總是願意學習新東西！