10 корисних Git команд, які полегшать роботу

23.04.2019 Cinex Linux, Windows, Веб-розробка 0

ПОШУК

ПОШУК …

ПОЗНАЧКИ

APACHE AUTORUN BASH BITCOIN BLOCKCHAIN COMPOSER CRYPTOCURRENCY CSS DATABASE DJANGO DOCKER ELOQUENT ENGLISH ENUM GAMES GIMP GIT HTML JAVA JAVASCRIPT JQUERY LARAVEL MERCURIAL PARSING PHOTOSHOP PHP PHPSTORM

Провідні розробники поділилися топом Git команд, які незамінні в роботі з найпопулярнішою системою контролю версій.

За останні кілька років популярність git зросла, що дозволило цій системі стати найбільш поширеною. Вона використовується різними програмістами і командами розробників, починаючи невеликими опен-сорс проектами і закінчуючи linux kernel. Існує велика кількість Git команд, які важливо знати.

Давайте ж розберемо топ Git команд.

git log --no-merges

PLAYONLINUX PLUGINS
PROBLEM SOLVING PYTHON REGEXP
SEO SOFTWARE ENGINEERING SPRING
SQL TELEGRAM TIPS AND TRICKS
TIPS AND TRICKS UNIT TESTING
WORDPRESS ВІРШІ ШПАРГАЛКИ

ОСТАННІ НОТАТКИ

20 Laravel Eloquent порад і трюків
15.12.2019

Як зробити скріншот сайту по URL на PHP
04.08.2019

Docker Cheat Sheet
05.07.2019

Гарячі клавіші Ubuntu Linux
01.07.2019

Git happens! 6 типових помилок Git і як їх виправити
22.06.2019

НЕДАВНІ КОМЕНТАРІ

Ця команда показує всю історію комітів, але пропускає ті, в яких відбулися злиття двох гілок, або де було вирішено конфлікт фіксації. Вона дозволяє швидко переглянути всі зміни, зроблені в проекті без скупчення злиттів в історії git.

```
$git log --no-merges

commit e75fe8bf2c5c46dbd9e1bc20d2f8b2ede81f2d93
Author:  John
Date:   Mon Jul 10 18:04:50 2017 +0300

    Add new branch.

commit 080dfd342ab0dbdf69858e3b01e18584d4eade34
Author:  John
Date:   Mon Jul 11 15:40:56 2017 +0300

    Added index.php.

commit 2965803c0deeac1f2427ec2f5394493ed4211655
Author:  John
Date:   Mon Jul 13 12:14:50 2017 +0300

    Added css files.
```

git revert --no-commit [commit]

Git revert створює новий коміт з вмістом, отриманим з усіх існуючих комітів, які були їм скасовані. Якщо ви хочете завернути названий коміт і уникнути автоматичних помилок, можете використовувати --no-commit або скорочення -n.

git diff -w

`Git diff` показує зміни між двома комітами, робочими деревами або файлами на диску. Коли кілька людей працює над одним і тим же проектом, часто відбуваються зміни через tab текстового редактора. Щоб ігнорувати відмінності, викликані прогалинами при порівнянні рядків, можна використовувати команду з -w.

git diff --stat

Показує, як кожен файл був змінений за певний час. Ви можете додати 3 параметра: **width** для визначення ширини виведення за замовчуванням, **name-width** для установки ширини імені файлу і **count** для обмеження виведення на перше число рядків.

```
$ git diff --stat
 index.php | 83 ++++++++++++++++++++++++++++++----------------------------
 1 file changed, 43 insertions(+), 40 deletions(-)
```

```
$ git diff --stat-width=10
 index.php | 83 +++---
 1 file changed, 43 insertions(+), 40 deletions(-)
```

git reset --soft HEAD^

Скиньте head до певного коміта, не торкаючись індексного файлу і робочого дерева. Всі зміни, зроблені після цієї фіксації, переносяться на етап "поставлені для комітів". Далі вам просто потрібно запустити git commit, щоб додати їх назад.

git stash branch [branch-name] [stash]

Ця команда створює нову гілку з ім'ям branch-name і перевіряє її, а потім застосовує до неї зміни від заданого stash і скидає його. Якщо жоден stash не вказано, використовується останній. Це дозволяє застосовувати будь-які заховані зміни в більш безпечному середовищі, яка згодом може бути об'єднана з майстром.

git branch -a

Команда показує список всіх віддалених і локальних гілок. Ви можете використовувати прапор --merged, щоб бачити тільки ті гілки, які повністю об'єднані з головною. Таким чином, ви можете

відстежувати свої гілки і дізнаватися, які з них більше не використовуються, і можуть бути видалені.

```
$ git branch -a

  dev
* master
  remotes/origin/HEAD -> origin/master
  remotes/origin/dev
```

git commit --amend

За допомогою **git commit --amend** ви можете змінити свій попередній коміт, замість того щоб створювати новий. Якщо ви не внесли свої зміни в віддалену гілку, можете використовувати цю команду для внесення змін в останній коміт, додавання останніх змін і навіть зміни повідомлення про коміт.

git pull --rebase

Git pull --rebase змушує git спочатку витягнути зміни, а потім переустановити розблоковані коміти поверх останньої версії віддаленої гілки. Параметр --rebase може використовуватися для створення лінійної історії, уникаючи непотрібних фіксацій.

git add -p

Коли ви використовуєте цю команду замість негайного додавання всіх скоєних змін, система запитує, що зробити з кожним з них. Таким чином, ви зможете самі вибрати, що саме хочете закомітити.

```
index db78332..a814f7e 100644
--- a/package.json
+++ b/package.json
@@ -6,7 +6,6 @@
```

```
5     },
6     "devDependencies": {
7       "bootstrap-sass": "^3.3.7",
8 -     "gulp": "^3.9.1",
9       "jquery": "^3.1.0",
10      "laravel-elixir": "^6.0.0-11",
11      "laravel-elixir-vue-2": "^0.2.0",
12 Stage this hunk [y,n,q,a,d,/,e,?]?
```

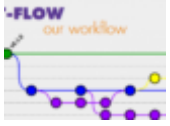
git log --graph

З цією опцією ви зможете побачити невеликий граф в форматі ASCII, який показує поточну гілку і історію злиттів:

```
1  $ git log --pretty=format:"%h %s" --graph
2  * 2d3acf9 ignore errors from SIGCHLD on trap
3  *  5e3ee11 Merge branch 'master' of git://github.com/dustin/grit
4  |\
5  | * 420eac9 Added a method for getting the current branch.
6  * | 30e367c timeout code and tests
7  * | 5a09431 add timeout protection to grit
8  * | e1193f8 support for heads with slashes in them
9  |/
10 * d6016bc require time for xmlschema
11 *  11d191e Merge branch 'defunkt' into local
```

Джерело

GIT TIPS AND TRICKS



« ПОПЕРЕДНІЙ
Git Flow

ДАЛІ »
Система контролю версій Git: розширена шпаргалка

ЗАЛИШТЕ ПЕРШИЙ КОМЕНТАР

Залишити коментар

Вашу адресу електронної пошти не буде опубліковано

Коментувати

Ім'я*

Email *

Збережіть моє ім'я, електронну пошту у цьому веб-переглядачі наступного разу, коли я коментуватиму.

ОПУБЛІКУВАТИ КОМЕНТАР