CHEAT-SHEET: GITHUB GIT

Git es el sistema de control de versiones distribuido de fuente abierta que facilita las actividades de GitHub en su computadora portátil o de escritorio. Esta hoja de referencia rápida resume las instrucciones de las líneas de comando de Git más comúnmente usadas.

Este software de código abierto se puede descargar tanto para Linux, Windows, Mac y Solaris, y en este tutorial aprenderás los comandos básicos de GIT para sacarle el mejor provecho.

GIT CONFIG

Uno de los comandos más usados en git es git config, que puede ser usado para establecer una configuración específica de usuario, como sería el caso del email, un algoritmo preferido para diff, nombre de usuario y tipo de formato, etc... Por ejemplo, el siguiente comando se usa para establecer un email.

```
git config --global user.email uc@underc0de.org
```

GIT INIT

Se usa para crear un nuevo repertorio GIT.

```
git init
```

GIT ADD

Puede ser usado para agregar archivos al index. Por ejemplo, el siguiente comando agrega un nombre de archivo temp.txt en el directorio local del index.

```
git add temp.txt
```

GIT CLONE

Se usa con el propósito de revisar repertorios. Si el repertorio está en un servidor remoto se tiene que usar el siguiente comando.

```
git clone underc0de@93.188.160.58:/path/to/repository
```

Pero si se desea crear una copia local funcional del repertorio, el comando indicado es

```
git clone /path/to/repository
```

GIT COMMIT

El comando commit es usado para cambiar a la cabecera. Ten en cuenta que cualquier cambio comprometido no afectara al repertorio remoto. Usa el comando.

```
git commit –m “Message to go with the commit here”
```

GIT STATUS

Este comando muestra la lista de los archivos que se han cambiado junto con los archivos que están por ser añadidos o comprometidos.

```
git status
```

GIT PUSH

Este es uno de los comandos más básicos. Un simple push envía los cambios que se han hecho en la rama principal de los repertorios remotos que están asociados con el directorio que está trabajando.

```
git push  origin master
```

GIT CHECKOUT

Se puede usar para crear ramas o cambiar entre ellas. Por ejemplo, el siguiente comando crea una nueva y se cambia a ella.

```
command git checkout -b <banch-name>
```

Para cambiar de una rama a otra solo es necesario usar:

```
git checkout <branch-name>
```

GIT REMOTE

Permite conectar a un repositorio remoto. Muestra los repositorios remotos que están configurados actualmente.

```
git remote -v
```

Este comando te permite conectar al usuario con el repositorio local a un servidor remoto.

```
git remote add origin <93.188.160.58>
```

GIT BRANCH

Sirve para listar, crear o borrar ramas.
Para listar todas las ramas se usa: `git branch`
Para borrar la rama: `git branch -d <branch-name>`

GIT PULL

Para poder fusionar todos los cambios que se han hecho en el repositorio local trabajando: `git pull`

GIT MERGE

Este comando se usa para fusionar una rama con otra rama activa:

```
git merge <branch-name>
```

GIT DIFF

Este comando se usa para hacer una lista de conflictos. Para poder ver conflictos con el archivo base usa: `git diff --base <file-name>`
Para ver los conflictos que hay entre ramas que están por ser fusionadas para poder fusionarlas sin problemas: *`git diff <source-branch> <target-branch>`*
Para solo ver una lista de todos los conflictos presentes usa: *`git diff`*

GIT TAG

Se usa para marcar commits específicos con asas simples:

```
git tag 1.1.0 <instert-commitID-here>
```

GIT LOG

Muestra una lista de commits en una rama con todos los detalles.

```
commit 15f4b6c44b3c8344caasdac9e4be13246e21sadw
Author: Underc0de <Underc0de@gmail.com>
Date:   Mon Feb 10 12:56:29 2020 -0600
```

GIT RESET

Para resetear el index y el directorio que está trabajando al último estado comprometido se usa este comando: `catgit reset - -hard HEAD`

GIT RM

Se puede usar para remover archivos del index y del directorio que está trabajando: `git rm filename.txt`

GIT STASH

Uno de los comandos menos conocidos, ayuda a salvar cambios que no están por ser comprometidos inmediatamente, pero temporalmente:

```
git stash
```

GIT SHOW

Se usa para mostrar información sobre cualquier objeto git: `git show`

GIT FETCH

Busca todos los objetos de un repositorio remoto que actualmente no reside en el directorio local que está trabajando: `git fetch origin`

GIT GREP

Permite buscar en los árboles de contenido cualquier frase o palabra. Por ejemplo, para buscar por www.tupaginaweb.com en todos los archivos:

```
git grep “www.tupaginaweb.com”
```

GITK

Este es la interfaz gráfica para un repositorio local: `gitk`

GIT INSTAWEB

Con este un servidor web puede correr interconectado con el repositorio local. Un navegador web también está automáticamente dirigido a el: `git instaweb –http=webrick`

GIT ARCHIVE

Permite crear archivos zip o tar que contengan los constituyentes de un solo árbol de repositorio: `git archive – -format=tar master`

GIT REBASE

Para la re aplicación de los compromisos en otra rama:

```
git rebase master
```