**Tutorial GIT Dasar – Cara Menggunakan GIT**

**Pengenalan**

Version control system sangat bermanfaat untuk mencegah developer mengalami kesulitan dalam menganalisa perubahan & bekerja dengan kode yang diakses bersama-sama. Sederhananya, VCS adalah prinsip penting dalam sistem manajemen konfigurasi software yang berkaitan dengan kebutuhan manajemen perubahan dalam project. Perubahan/revisi/update yang dibuat bisa dikenali melalui kode atau angka. Informasi seperti catatan waktu dan identitas yang membuat perubahan juga tetap ada. Dalam tutorial ini, kita akan membahas GIT, salah satu VCS yang paling sering digunakan. Selain cara menggunakan GIT, Anda juga akan mempelajari dasar GIT seperti cara install GIT di sistem yang berbeda dan bagaimana cara menggunakannya

**Apa itu GIT?**

Pada tahun 2005, Linus Torvalds (orang yang membuat kernel Linux) membuat GIT dan sejak itu GIT secara aktif dikelola oleh Junio Hamano, programmer Jepang. Saat ini, GIT menjadi salah satu version control system yang paling terkenal dan ada jutaan project di seluruh dunia yang menggunakan GIT untuk version control nya (termasuk project komersil dan open source). GIT sepenuhnya gratis dan bisa di-download di Mac, Linux, Windows, dan Solaris, daRI [**website resminya**](https://git-scm.com/downloads). Fitur-fitur unggulan dari GIT antara lain:

1. Version control system yang terdistribusi, GIT menggunakan pendekatan peer to peer, tidak seperti yang lainnya seperti Subversion (SVN) yang menggunakan model **client-server**.  
2. GIT memungkinkan developer untuk memiliki branch kode yang independent dan massive. Membuat, menghapus dan menggabungkan branch tersebut menjadi lebih cepat, lancar dan tidak butuh waktu yang lama.  
3. dalam GIT, semua operasinya bersifat atomic, artinya sebuah tindakan akan benar-benar diselesaikan dengan lengkap atau sama sekali gagal. Ini sangat penting, karena di beberapa version control system seperti CVS, operasinya bersifat non-atomic. Jika ada operasi yang ‘gantung’ dan terkait dengan repository, kondisi repository menjadi tidak stabil.  
4. Dalam GIT, semuanya disimpan dalam folder .git. Berbeda dengan VCS lain seperti SVN atau CVS, dimana metadata file disimpan dalam folder tersembunyi seperti .cvs, .svn, .etc.  
5. GIT menggunakan data model yang membanti memastikan integritas cryptographic apapun ada dalam repository. Setiap kali sebuah file ditambahkan atau di-commit, checksum-nya akan diciptakan; sama hal nya, juga di-retrieve lewat checksum-nya juga.  
6. Fitur menarik lainnya yang ada di GIT adalah staging area atau index. Dengan stagin area, developer bisa memformat commit dan membuatnya bisa di-review sebelum benar-benar diterapkan.

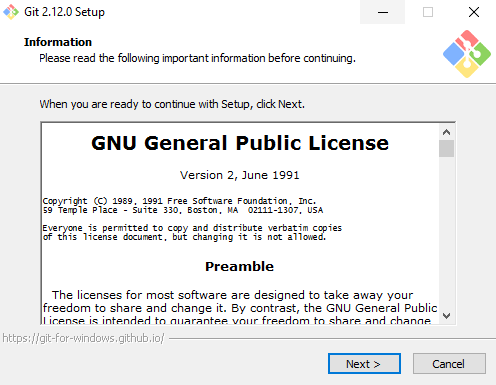
GIT sangat sederhana dalam penggunannya. Untuk memulai, Anda bisa membuat repository atau men-checkout yang sudah ada. Setelah instalasi, perintah git-init akan men-setup semuanya. Selain itu, perintah git clone bisa membuat salinan repository lokal untuk user.

**Langkah 1 – Install GIT di berbagai jenis sistem**

**Opsi 1 – Install GIT di Windows**

Menginstall GIT di Windows sangat mudah, cukup dengan mendownload dan menjalankan instalasinya. Ikuti langkah berikut ini untuk meng-install GIT di Windows:

1. Buka [**website ini**](https://git-for-windows.github.io/) dan download installer GIT untuk Windows.  
2. Setelah download, buka file tersebut untuk menjalankan proses instalasi. Ikuti semua instruksi, klik **Next** dan **Finish** hingga semua proses instalasi selesai.

  
3. Jalankan perintah berikut ini di terminal:

git config --global user.name "John Smith"

git config --global user.email "example@email.com"

Catatan: Jangan lupa mengganti John Smith dan example@email.com dengan detail login Anda sendiri.

Setelah itu, GIT di Windows Anda siap digunakan.

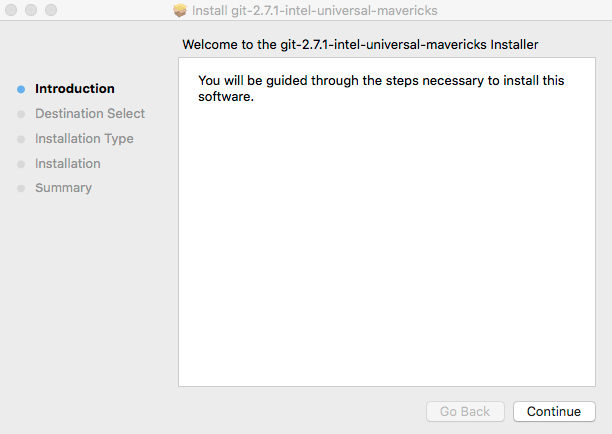
**Opsi 2 – Menginstall GIT di MacOS**

Ada banyak cara meng-install GIT di perangkat Mac. Bisa jadi GIT sudah terinstall di komputer Anda jika Anda telah memiliki aplikasi XCode sebelumnya. Jalankan perintah berikut ini di terminal untuk memastikan:

git --version

Jika Anda mendapatkan output seperti git version 2.7.0 (Apple Git-66), Anda beruntung. Namun jika tidak, ikuti langkah berikut ini:

1. Buka [**website ini**](https://sourceforge.net/projects/git-osx-installer/files/) dan download installer terbaru untuk Mac.  
2. Ikuti instruksi di layar dan selesaikan proses instalasi.

  
3. Sekali lagi, ketikkan perintah git --version  
untuk memastikan bahwa instalasi telah berhasil.  
4. Jalankan perintah berikut ini di terminal Anda untuk mengkonfigurasi email Anda dan username yang digunakan untuk akun GIT:

git config --global user.name "John Smith"

git config --global user.email "example@email.com"

Catatan: Jangan lupa mengganti John Smith dan example@email.com dengan detail login Anda sendiri.

**Opsi 3 – Install GIT di Linux**

Jika Anda adalah pengguna Linux, mungkin Anda telah terbiasa meng-install software dan paket di komputer Anda menggunakan perintah apt-get atau yum install. Begitu juga dengan instalasi GIT:

**Untuk pengguna Debian/Ubuntu (apt-get):**  
1. Buka terminal dan jalankan perintah berikut ini:

Sudo apt-get update

Sudo apt-get install git

2. Cek apakah instalasi sudah berhasil dengan menggunakan git --version

3. Jalankan perintah berikut di terminal untuk mengkonfigurasikan email Anda dan username yang digunakan dengan akun GIT Anda:

git config --global user.name "John Smith"

git config --global user.email "example@email.com"

Catatan: Jangan lupa mengganti John Smith dan example@email.com dengan detail login Anda sendiri.

**Fedora (yum/dnf)**

Anda bisa mengekstrak paket GIT menggunakan yum atau dnf.

1. Buka terminal dan jalan perintah ini:

Sudo dnf install git

Sudo yum install git

2. Cek instalasi dengan menggunakan git --version  
3. Jalankan perintah berikut di terminal untuk mengkonfigurasikan email Anda dan username yang digunakan dengan akun GIT Anda:

git config --global user.name "John Smith"

git config --global user.email "example@email.com"

Catatan: Jangan lupa mengganti John Smith dan example@email.com dengan detail login Anda sendiri.

**Langkah 2 – Menggunakan GIT**

Sekarang, setelah GIT di-install di komputer Anda, mari kita pelajari dasar GIT dan bagaimana cara menggunakan GIT.

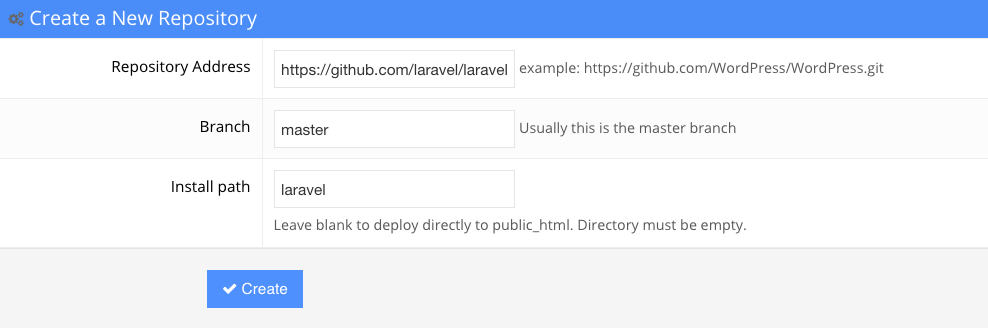
* **Membuat/mengatur/mengambil repository**: Repository adalah asset yang paling kuat dari semua project versioning. Untuk mengubah direktori menjadi repository GIT, jalankan perintah git init <directory>. Folder bernama .gt akan muncul di direktori bila perintah tersebut telah dijalankan. Kemudian, jika Anda telah memiliki direktori sebelumnya dan ingin mengambilnya, Anda bisa menggunakan perintah git clone. Jika Anda ingin mencoba mengambil repository lokal, gunakan perintah ini:

git clone /path/to/local/repository

Jika Anda ingin memeriksa repository yang tersimpan, gunakan:

git clone user.name@host:/path/to/remote/repository

Jika Anda memiliki akun di Hostinger, Anda bisa dengan mudah menyalin dan mengatur repository melalui **Member Area** -> **GIT**. Sebagai contoh, jika Anda ingin menyalin repository GIT, masukkan alamatnya, pilih branch dan lokasi install, lalu klik tombol.



Setelah selesai, Anda bisa mengatur repository Anda di bagian tersebut.

* **Alur kerja**: Setelah repository selesai dibuat, kita akan membahas tentang struktur pada GIT. Setiap repository lokal memiliki 3 cabang, yaitu: direktori kerja yang mengandung file; index berperan sebagai staging area dan HEAD yang menjadi penunjuk commit terakhir yang dilakukan oleh user. Jadi, seperti inilah cara kerjanya: user menambahkan file atau melakukan perubahan dari direktori kerja ke index (staging area) dan setelah di-review, file atau perubahan di-commit ke HEAD.
* **Perintah ADD dan COMMIT**: Perubahan yang diajukan atau file tambahan akan ditambahkan ke index dengan menggunakan perintah add. Untuk menambahkan file, perintah sederhananya adalah:

git add <file\_name>

Jika Anda cukup yakin untuk melakukan perubahan langsung di HEAD, Anda langsung gunakan perintah commit. Contohnya:

git commit –m “Tambahkan catatan untuk commit Anda”

**Catatan: Bila perintah commit telah dijalankan (dari direktori kerja), file akan di-commit ke HEAD, namun belum terkirim ke remote repository.**

* **Mengirim perubahan:** Setelah Anda melakukan commit perubahan (dan sudah yakin siap dikirim ke repository asli), Anda bisa menggunakan perintah push.Setelah git push origin master dijalankan dari direktori kerja, perubahan yang ada di HEAD akan dikirim ke remote repository. Seperti yang telah disebutkan sebelumnya, master bisa diganti  menjadi nama branch dimana perubahan ingin Anda commit.

Jika repository yang ada saat ini belum di-clone dan Anda ingin membuat sambungan antara repository dan remote server, maka jalankanlah perintah ini:

git remote add origin <server>

**Catatan: Ganti <server> dengan alamat remote server.**

Setelah di-clone, perubahan yang telah dibuat akan dikirim ke server.

* **Branches:** Fitur keren lainnya dari GIT adalah kemampuannya untuk memungkinkan developer dan project manager untuk membuat branch independen yang berbeda dalam 1 project. Tujuan utama branch adalah untuk membuat fitur namun tetap mengisolasinya dari yang lain. Branche default di project adalah master branch. Karena banyak branch yang dibutuhkan bisa dibuat dan digabungkan dengan master branch.Branch baru bisa dibuat dengan menggunakan perintah berikut ini:

git checkout -b feature\_n \*

**Catatan: feature\_n adalah nama branch-nya.**

Jika Anda ingin kembali ke master branch utama, jalankan perintah ini:

git checkout master

Branch bisa dihapus dengan menggunakan perintah ini:

git checkout -b feature\_n

Untuk membuat branch bisa digunakan oleh pengguna lain, Anda harus mengirimkannya ke remote repository melalui perintah ini:

git push origin feature\_n

* **Update & Merging:** Jika Anda ingin meng-update direktori kerja lokal Anda saat ini ke yang paling baru dari remote repository, jalankan perintah git pull.Untuk menggabungkan branch lain ke yang sudah aktif, gunakan:

git merge feature\_n

Jika Anda melakukan pull atau merge, GIT akan selalu mencoba menangani conflict  sendiri, namun kadang-kadang juga gagal. Jika terjadi kegagalan karena conflict, maka pengguna harus melakukan resolve conflict secara manual. Setelah Anda meng-edit file (untuk menghapus conflict), tandai sebagai merged dengan perintah:

git add <nama\_file>

Jika setelah merging Anda ingin melihat perubahan, jalan perintah:

git diff <nama\_sumber\_branch> <nama\_target\_branch>

* **Tagging:** Sebelum merilis update/perubahan apapun, sangat disarankan untuk membuat tag. Untuk melakukannya di GIT, gunakan perintah:

git tag 1.1.0 1c2d2d56fa

1c2d2d56fa adalah perintah yang terkait dengan 10 karakter pertama dari commit-id, yang dijadikan sebagai tag. Commit ID bisa ditemukan di log.

* Log: Catatan history dari repository bisa dipelajari dari log. Perintah git log yang menampilkannya. Untuk menampilkan commit yang dibuat oleh seorang pengguna, Anda bisa menggunakan perintah:

git log --author =Bambang

Log yang terkompresi (1 commit per baris) bisa dilihat dengan:

git log --pretty=oneline

Untuk hanya melihat file yang berubah:

git log --name-status

Mengganti perubahan di lokal: Jika Anda tidak sengaja membuat kesalahan dan ingin kembali ke perubahan normal yang sebelumnya, gunakan perintah:

git checkout -- <filename>

Dengan melakukan hal tersebut, akan mengganti perubahan di direktori kerja dengan yang terakhir ada di HEAD. Perubahan apapun yang ditambahkan ke index tidak akan terkena dampak.

Sebaliknya, jika semua perubahan/commit di-drop dan master branch lokal dibutuhkan untuk menunjukkan history terakhir dari server, jalankan perintah:

git fetch origin

git reset --hard origin/master