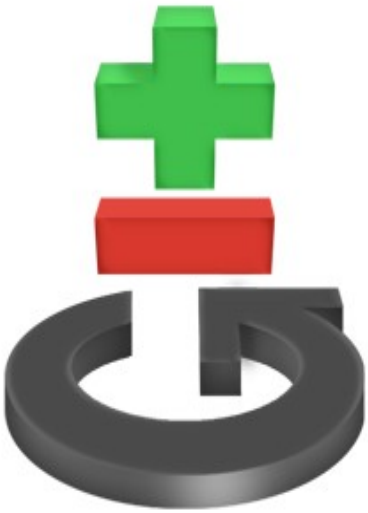
Οδηγός για…

Git και Github



Κατάλογος περιεχομένων

Εισαγωγή

Όλα τα παρακάτω σχηματίζουν έναν βασικό οδηγό για το Git και το GitHub. Σημειώστε πως αυτός ο οδηγός δεν είναι πλήρης, ούτε παρέχει λεπτομέρειες για το πως λειτουργεί το συγκεκριμένο εργαλείο. Στο τέλος, υπάρχουν χρήσιμα link που πιστεύω θα βρείτε ότι χρειάζεστε για την λειτουργία του git. Καλή ανάγνωση…

A Git και DVCS

Το git είναι ένα κατανεμημένο (distributed) σύστημα ελέγχου εκδόσεων κώδικα (DVCS) και δίνει την δυνατότητα να δουλεύετε πάνω σε ένα έργο έχοντας τον πλήρη έλεγχο του κώδικα που αναπτύσετε χωρίς να είσαστε συνδεδεμένοι στο διαδίκτυο. Η γενική ιδέα είναι ότι μπορείτε να αναπτύσετε κώδικα τοπικά στον υπολογιστή σας και κάθε στιγμή να ανανεώνετε το δημόσιο αποθετήριο ώστε να το βλέπουν οι συνεργάτες σας.

A.1 Εγκατάσταση του git σε Ubuntu/Debian

Για διανομές βασισμένες σε Debian(όπως το Ubuntu) εκτελούμε μέσω του τερματικού

```
sudo apt-get install git
```

Σημείωση 1: Με την εντολή `sudo aptitude search git` εμφανίζονται πακέτα που σχετίζονται με το git.

A.2 Αποθετήριο του Git

Το git είναι ένα εργαλείο όπως θα δείτε και παρακάτω που είναι υλοποιημένο σε κώδικα. Μπορείτε να βρείτε τον κώδικα αυτό στο GitHub και να τον κλωνοποιήσετε στον υπολογιστή σας(θα δείτε παρακάτω περισσότερα για το clone) με:

```
$ git clone git://git.kernel.org/pub/scm/git/git.git
```

A.3 Βασικές εντολές

Ο λόγος που προτρέχω και γράφω τις παρακάτω εντολές είναι για όσους δεν θέλουν να διαβάσουν ολόκληρο τον οδηγό και χρειάζονται μόνο αυτές. Προφανώς αν δεν ανήκετε σε αυτή τη κατηγορία πηδήξτε την τρέχων παράγραφο.

`git init`	Αρχικοποίηση του τρέχων φακέλου ώς αποθετήριο
`git pull`	Ανανέωση του public αποθετηρίου με το local
`git add <filenames>`	Προσθήκη των αρχείων στο INDEX
`git push`	Συγχώνευση brances με το local
`git clone <url>`	Αντιγραφή public αποθετηρίου στον υπολογιστή μας
`git commit -m <message>`	Δημιουργία στιγμιοτύπου με δείκτη HEAD

A.4 Ssh ρύθμιση για συχρονισμό με το GitHub

Για να δουλέψετε με το git και ταυτόχρονα με το github θα πρέπει να υπάρχει ένας συχρονισμός ώστε κάθε φορά που θέλετε να κάνετε push/pull κώδικα, το git να το καταλαβαίνει αυτόματα και να δρά σωστα. Αυτό που θα κάνουμε είναι να δημιουργήουμε ένα rsa κλειδί και να το μεταφέρουμε στον λογαριασμό μας στο github ώστε κάθε φορά να μπορείτε να ανεβάζετε(push) και να κατεβάζετε(pull) αρχεία χωρίς να γράφετε εντολές και κωδικούς και άλλα διάφορα. Δεν θα γίνω πολύ αναλυτικός σε αυτό το βήμα γιατί ο σκοπός του οδηγού είναι άλλος.

Εκτελείτε την παρακάτω εντολή:

```
$ ssh-keygen -t rsa
```

και αυτόματα θα σας ζητήσει για ένα όνομα του κλειδιου. Μπορείτε να δώσετε ότι θέλετε όπως, “github_key” και συνεχίζετε παρακάτω όπου θα σας ζητήσει να δώσετε δυο φορές έναν κωδικό για αυτό το κλειδί. Μπορείτε να το αφήσετε και κενό αλλά καλύτερα δώστε κάποιον απλό. Μόλις γίνει η σύνδεση, δεν θα χρειαστεί να ξανα-πληκτρολογήσετε τον κωδικό. Με άλλα λόγια θα πρέπει να κάνετε restart το σύστημά σας για να σας ξανα-ζητηθεί ο κωδικός.

Ωραία, το κλειδί μας ολοκληρώθηκε και θα πρέπει να έχετε δυό αρχεία τώρα με ονόματα, <όνομα_κλειδιού> και <όνομα_κλειδιού>.pub. Το πρώτο δεν θα πρέπει να το πειράξετε ούτε να το δώσετε πουθενά. Είναι καθαρά προσωπικό! Το .pub όπως καταλάβένετε είναι το public κλειδί οπότε θα πρέπει να το μεταφέρετε στον host που στην προκειμένη περίπτωση είναι το github. Οπότε, ανοίγουμε τον λογαριασμό μας στο github και πηγαίνουμε στο, "Account Settings/SSH keys/Add SSH key". Εκεί θα πρέπει να κάνουμε copy/paste ότι ακριβώς περιέχει το <όνομα_κλειδιού>.pub αρχείο μας. Προσέξτε να μην προσθέσετε κάποιο space, enter etc. Πρέπει να είναι όπως ακριβώς φαίνεται στο αρχείο μας.

A.5 Ρύθμιση της εγκατάστασης του Git

Το εργαλείο git θα πρέπει να δουλεύει μια χαρά τώρα! Παρακάτω θα δώσουμε κάποιες ρυθμίσεις ώστε αν τυχόν κάνετε κάποιο push/pull να ξέρει κάθε ένας στην ομάδα ποιος το έκανε. Αρχικά θα δώσετε ένα όνομα και ένα email όπως φαίνετε και παρακάτω. Έπειτα μπορείτε για να είναι πιο όμορφο να δώσετε κάποια χρώματα στις εντολές του git.

α Πληροφορίες για τον author/commiter

```
git config --global user.name "type your name here"
git config --global user.email “Type your email here”
```

β Χρωματισμένη έξοδος εντολών

```
git config --global color.ui auto
git config --global alias.st status
```

Κάθε στιγμή μπορείτε να ενημερώνεστε για τις ρυθμίσεις που έχετε δώσει εκτελώντας:

```
git config -l
```

A.6 Εγκατάσταση του gitg

Είναι ένα εργαλείο με γραφικό περιβάλλον που προσφέρει ευκολία για να βλέπουμε τις εκδόσεις που κώδικά μας. Το αποθετήριο αναγνωρίζεται αυτόματα και θα μπορούμε να βλέπουμε λεπτομέρειες για τα commit που έχουν γίνει. Η εγκατάσταση γίνεται εκτελώντας:

```
apt-get install gitg
```

Β Δημιουργία αποθετηρίων στον υπολογιστή μας

Αποθετήριο? Για κάποιον που δεν έχει ξαναασχοληθεί με DVCS η έννοια του αποθετήριου είναι κινέζικη(τουλάχιστον έτσι ήταν για μένα στηνα αρχή :P). Αποθετήριο λοιπόν, δεν είναι τίποτα άλλο απο έναν 'φάκελο'. Μια τοποθεσία που αποθηκεύονται αρχεία. Θέλοντας να καλύψω όσες περισσότερες περιπτώσεις μπορώ θα δούμε 3 περιπτώσεις δημιουργίας αποθετηρίου.

Β.1 Δημιουργία κενού αποθετηρίου στον υπολογιστή μας

Αρχίζουμε με το github! Ανοίγουμε την σελίδα https://github.com/ και αφού κάνουμε login στον λαγαριασμό μας, πάνω δεξιά επιλέγουμε το “create new repo”, δίνουμε ένα όνομα στο αποθετήριο και το δημιουργούμε. Όπως θα δείτε θα πρέπει να σας εμφανίσει τις εντολές με τις οποίες θα δουλέψετε. Το αποθετήριο έχει δημιουργηθεί μόνο στο github τώρα. Θα πρέπει να πάμε στον υπολογιστή μας και απο ένα τερματικό να δημιουργήσουμε έναν φάκελο με όνομα το ίδιο όνομα του αποθετηρίου στο github.

```
mkdir <όνομα αποθετηρίου>
```
Έπειτα, αφού μπούμε στον φάκελο εκτελούμε:
```
git init
```
και έτσι αρχικοποιούμε τον φάκελο αυτό σε αποθετήριο!

Εκτελώντας:
```
ls -a
```
μπορείτε να δείτε τα κρυφά αρχεία του φακέλου-αποθετηρίου. Ο .git φάκελος είναι αρχεία που χρειάζονται για το αποθετήριο. Δημιουργείται αυτόματα για κάθε ένα αποθετήριο που φτιάχνετε και περιέχει πληροφορίες για τα add και τα commit που θα δούμε παρακάτω.

Β.2 Δημιουργία αποθετηρίου έχοντας υλοποιημένο κάποιο project

Προυπόθεση εδώ, είναι να έχουμε ήδη υλοποιημένο κάποιο project τοποθετημένο σε κάποιον φάκελο. Γιατί να κάνουμε local αποθετήριο; Γιατί μπορεί να θέλουμε να δουλέψουμε κι άλλο πάνω σε αυτό το project και να βγάλουμε ένα version 2 με κάποιον συνεργάτη. Αφού πάμε πάλι στο github και κάνουμε την γνωστή διαδικασία στον φάκελο απλά εκτελούμε:
```
git init
```
αρχικοποιείται όπως και παραπάνω σε αποθετήριο και είμαστε έτοιμοι!

Β.3 Δημιουργία local αποθετηρίου “αντιγράφοντας” κάποιο απο τον Github

Σε αυτή τη περίπτωση, υπάρχει κάποιο αποθετήριο στο github που μας ενδιαφέρει(είτε είναι δικό μας είτε όχι). Αν δεν είναι δικό μας θα πρέπει να είναι public για να συνεχίσουμε. Θέλουμε λοιπόν, να το κάνουμε clone στον υπολογιστή μας και να επεξεργαστούμε τον κώδικα χωρίς να έχει επίπτωση στο σημείο απο όπου τον πήραμε, χωρίς να επηρεάσει τον προγραμματιστή δηλαδή. Για να αντιγράψουμε το project απο το github στον local δίσκο μας εκτελούμε:
```
$git clone <url> myproject
```
το <url> αναγράφεται πάνω πάνω στον λογαριασμό μας στο github. Δεν είναι το URL πλοήγησης του browser!

Γ Basic Reference

Έχουμε ένα αποθετήριο έτοιμο για χρήση και κάποια προγράμματα ίσως. Θέλουμε να δουλέψουμε πάνω στο αποθετήριο αυτό χρησιμοποιώνας το git φυσικά. Οι παρακάτω εντολές προφανώς δεν αναλύονται αρκετά καθώς είναι βασισμένες στα manpages αλλα μπορείτε να πάρετε μια ιδέα τι κάνει κάθε μια και να τις έχετε στο μυαλό σας. Εδώ μπορείτε να βρείτε τις πιο ένα pdf με τις πιο βασικές εντολές. Είναι πολύ χρήσιμο αν εκτυπωθεί ! :)

Γ.1 Basic Snapshotting

Οι παρακάτω εντολές μας βοηθάνε ώστε να δουλεύουμε με στιγμιότυπα. Ανατρέξτε στη progit παράγραφο για περισσότερα.

- `add`
- `status`
- `diff`
- `commit`
- `reset`
- `rm`
- `mv`

α git-add

Περιγραφή: Προσθέτει το περιεχόμενο του αρχείου στο index και το ετοιμάζει για να γίνει το επόμενο commit.

Έχοντας αυτό τον χώρο που ονομάζεται index μπορείτε κάθε στιμγή να κάνετε commit. Με το commit στην ουσία, αποθηκεύονται όλα τα περιεχόμενα του INDEX με ένα log message κάπου αλλού. Απο εκεί και πέρα μπορούμε να έχουμε πολλά commit και πάντα ένα INDEX όπως είναι προφανές.

β git-status

Περιγραφή: Εμφανίζει την κατάσταση του Working tree δηλαδή, μονοπάτια που έχουν διαφορά στο index και στο τρέχων HEAD commit. Τα πρώτα αρχεία μπορέιτε να τα κάνετε commit. Τα τρίτα και τέταρτα μπορείτε να τα κάνετε commit αφού γίνει πρώτα add.

γ git-diff

Περιγραφή: Εμφανίζει διαφορές στα commit που έχουν γίνει και στο working tree με τα index.

δ git-commit

Περιγραφή: Καταγράφει αλλαγές που έχουν γίνει στο αποθετήριο. Αποθηκεύει ότι υπάρχει στο index σε ένα νέο commit μαζί με ένα μήνυμα περιγραφής που δίνεται απο τον χρήστη.

ε git-reset

Περιγραφή: Κάνει επαναφορά στην τρέχων κεφαλής(HEAD) σε ένα συγκεκριμένο σημείο.

στ git-rm

Περιγραφή: Απομακρύνει αρχεία απο το working tree και το index. Σημειώστε ότι με git-rm δεν απομακρύνεται κάποιο αρχείο αλλα το working directory

ζ git-mv

Περιγραφή: Μετακινεί η κάνει μετονομασία σε κάποιο αρχείο, φάκελο ή σύνδεσμο

Γ.2 Branching and merging

Οι εντολές εδώ σχετίζονται με τους κλάδους, τις εκδόσεις του project μας και τον κατάληλο χειρισμό τους.

- `branch`
- `checkout`
- `merge`
- `log`
- `stash`
- `tag`

α git-branch

Περιγραφή: Εμφανίζει σε λίστα ή δημιουργεί τα branches που υπάρχουν.

β git-merge

Περιγραφή: Ενώνει δύο η περισότερους κλάδους μαζί. Για παράδειγμα τον master που πιθανώς έχετε μέ κάποιον δέύτερο που δημιουργήσατε.

γ git-log

Περιγραφή: Εμφανίζει πληροφορίες για τα commit που έχουν γίνει.

Γ.3 Sharing and Update projects

Οι παρακάτω αντολές σχετίζονται με τον διαμοιρασμό των projects. Για παράδειγμα το upload και download των δικών σας project σε κάποιο αποθετήριο.

- `fetch`
- `pull`
- `push`
- `remote`

α git-fetch

Περιγραφή: Μας φέρνει-κατεβάζει αντικείμενα απο κάποιο άλλο αποθετήριο.

β git-pull

Περιγραφή: Φέρνουμε στον υπολογιστή μας κάποιο άλλο αποθετήριο ή τοπικό κλάδο(branch).

πχ. `git pull --rebase`

Με το rebase ζητάμε από το git να τροποποιήσει το τοπικό μας αποθετήριο και να κάνει αυτά τα commit που δεν υπάρχουν ακόμα στο δημόσιο αποθετήριο, να εμφανιστούν με τέτοιο τρόπο ώστε να περάσουν πρώτα στο ιστορικό (του τοπικού αποθετηρίου) τα commit που υπήρχαν στο δημόσιο αποθετήριο, και τα commit του τοπικού να εμφανιστούν πιο πρόσφατα στο ιστορικό. Αυτό ονομάζεται rebase.

γ git-push

Περιγραφή: Ανανεώνουμε κάποιο public αποθετήριο.

πχ. `git push -u origin master`

δ git-remote

Περιγραφή: Διαχείριση αποθετηρίων στα οποία κάποιο απο τα branches είναι σε trackign mode.

πχ. `git remote add origin git@github.com:penlix/test_fileee.git`

Γ.4 Επιθεώρηση και σύγκριση

Τα παρακάτω σχετίζονται με την σύγκριση εκδόσεων κώδικα που έχουμε υλοποιήσει και την εμφάνιση τυχόν log files.

- `log`
- `diff`
- `reset`

α git-log

Περιγραφή: εμφανίζει πληροφορίες για τα commits που έχουν γίνει

β git-diff

Περιγραφή: Εμφανίζει τις διαφορές στον κώδικα μεταξύ των commit και των branches.

γ git-reset --hard

Περιγραφή: Επαναφορά του αποθετηρίου στην κατάσταση του τελευταίου commit

Δ ProGit

Στο τοπικό αποθετήριο πάντα υπάρχει ένας κρυφός φάκελος .git όπου περιέχει αρχεία χρήσιμα για το αποθετήριο:

```
test/.git$ ls
branches  COMMIT_EDITMSG  config  description  HEAD  hooks  index  info  logs
objects  refs
```

- COMMIT_EDITMSG : περιέχει το σχόλιο όταν κάναμε το commit
- config : περιέχει πληροφορίες για τον χρήστη και τα remote add <name> που κάναμε
- description : περιέχει την περιγραφή του αποθετηρίου(αν δώσαμε)
- HEAD : μας δείχνει που ακριβώς γίνονται τα commits(path)
- index : ίσως είναι και το πιο σημαντικό. Είναι ένα αρχείο το οποίο μεγαλώνει καθώς κάνουμε git add κάποιο άλλο αρχείο. Φανταστείτε το σαν έναν buffer που μόλις κάνουμε commit αδειάζει.

Είναι σημαντικό να έχετε στο μυαλό σας τα 3 δέντρα/ρόλοι όπως λέγονται. Συνοπτικά οι 3 ρόλοι είναι:

- *The HEAD* – τελευταίο commit που έγινε, επόμενος πατέρας
- *The Index* – πιθανώς το επόμενο commit που θα γίνει
- *The Working Directory* – sandbox

Μόνο και μόνο με αυτά τα τρία είμαι σίγουρος πως θα ξεκαθαρίσουν πολλά τοπία!

Δ.1 The HEAD

Σε γενικές γραμμές είναι ένας pointer στο τρέχων branch. Αυτό σημαίνει οτι θα είναι ο πατέρας αν γίνει ένα commit. Για να μην μπερδεύεστε απλά σκεφτείτε πως είναι ένα στιγμιότυπο απο το τελευταίο σας commit. Τι περιέχει το commit? Τα αρχεία που είχατε κάνει add.

```
$ cat .git/HEAD
ref: refs/heads/master

$ cat .git/refs/heads/master
e9a570524b63d2a2b3a7c3325acf5b89bbeb131e

$ git cat-file -p e9a570524b63d2a2b3a7c3325acf5b89bbeb131e
tree cfda3bf379e4f8dba8717dee55aab78aef7f4daf
author Scott Chacon  1301511835 -0700
committer Scott Chacon  1301511835 -0700
initial commit

$ git ls-tree -r cfda3bf379e4f8dba8717dee55aab78aef7f4daf
```

```
100644 blob a906cb2a4a904a152...   README
100644 blob 8f94139338f9404f2...   Rakefile
040000 tree 99f1a6d12cb4b6f19...   lib
```

Δ.2 The Index

Όταν δουλεύετε με αρχεία και έχετε κάνει κάποιες αλλαγές κάποια στιγμή θα γίνει ένα commit έτσι ώστε να ενημερωθεί η "βάση μας". Το Index είναι ένα στιγμιότυπο απο το επόμενο commit που προόκειται να γίνει. Σημειώστε πως όταν εκτελείτε git commit το commit κοιτά μόνο στο Index για τυχόν αλλαγές(έτσι είναι το default τουλάχιστον).

```
$ git ls-files -s
100644 a906cb2a4a904a152e80877d4088654daad0c859 0       README
100644 8f94139338f9404f26296befa88755fc2598c289 0       Rakefile
100644 47c6340d6459e05787f644c2447d2595f5d3a54b 0       lib/simplegit.rb
```

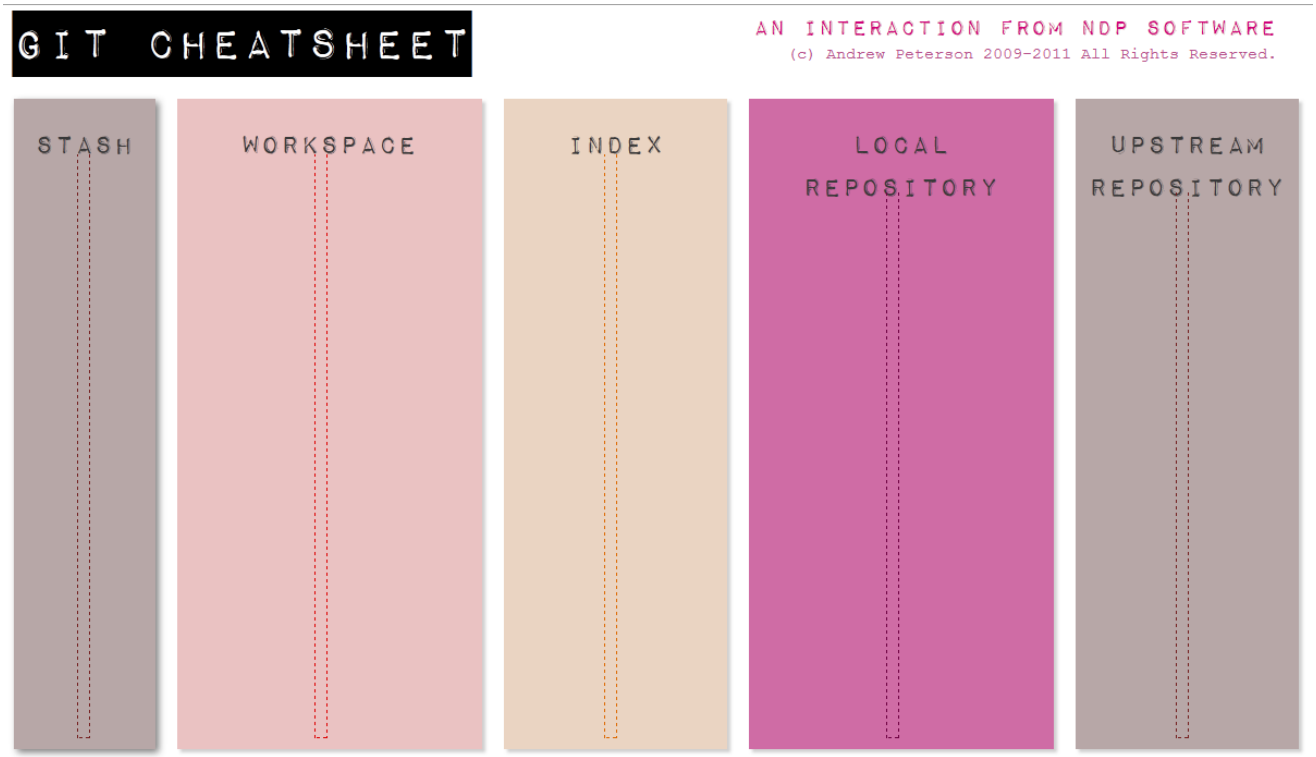
Δ.3 The Working Directory

Το directory αυτό δεν είναι τίποτα άλλο απο τον φάκελο με όλα τα αρχεία σας μέσα. Ο χώρος στο local σύστημά σας που μπορείτε εύκολα να κάνετε modify κάποια αρχεία. Δείτε και την παρακάτω εικόνα.

```
$ tree
.
├── README
├── Rakefile
└── lib
    └── simplegit.rb

1 directory, 3 files
```

Ρίξτε μια ματιά στην επόμενη σελίδα και θα δείτε σε γενικές γραμμές όλα τα “δέντρα” που έχετε στην διάθεσή σας. Προφανώς για μια βασική χρήση του git δεν είναι απαραίτητα αλλα όποιος θέλει να το μάθει εκτενές μπορεί να ακολουθήσει τον σύνδεσμο: http://git-scm.com/about



Για περισσότερες πληροφορίες δείτε εδώ: http://ndpsoftware.com/git-cheatsheet.html

Ε Χρήσιμοι συνδέσμοι

- Κεντρική σελίδα του github: https://github.com/
- Βοήθεια για το github: http://help.github.com/
- From wikipedia: http://en.wikipedia.org/wiki/GitHub
- GitSvncrashcourse: https://git.wiki.kernel.org/articles/g/i/t/GitSvnCrashCourse_512d.html
- Git Reference: http://gitref.org/basic/#add
- Οδηγός Git και Github του φόρουμ της ελληνικής κοινότητας Ubuntu-gr: https://forum.ubuntu-gr.org/viewtopic.php?f=9&t=19319

- http://help.github.com/git-cheat-sheets/
- http://www.vogella.com/articles/Git/article.html
- http://www.siteground.com/tutorials/git/commands.htm
- http://davidwalsh.name/git-commands
- http://gitref.org/basic/#add

ΣΤ Manpages

Το git δεν είναι τίποτα άλλο απο ένα κέλυφος με κάποιες ειδικές εντολές. Μπορείτε και εδώ να χρησιμοποιήσετε τα manpages όπως και στο Linux και να δείτε περισσότερες πληροφορίες για κάποια εντολή του git εκτελώντας:

```
git help <εντολή>
```