

Workbook Git

Best Practice



Martinus Ady H
martinus@[artivisi.com](mailto:martinus@artivisi.com)
last updated : 11 Jan 2011

Daftar Isi

Merge vs Rebase.....	3
Penjelasan.....	3
Panduan.....	3
Kesimpulan.....	3
Mengelola Pararel Development.....	4
Penjelasan.....	4
Panduan.....	4
Kesimpulan.....	4
Release Management.....	5
Penjelasan.....	5
Panduan.....	5
Kesimpulan.....	5

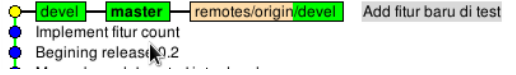
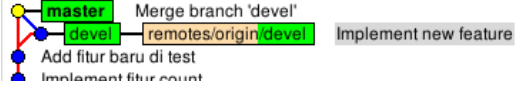
Merge vs Rebase

Penjelasan

Merge : Gunakan merge jika kita ingin ada percabangan pada history commit antar branch.

Rebase: Gunakan rebase jika kita ingin hanya 1 baris history commit saja.

Panduan

No	Aktifitas	Hasil
1.	Pindahlah ke branch devel, dan buatlah beberapa commit kemudian push echo "Fitur baru" >> test git add test git commit -m "Add fitur baru di test"	[devel 3fa3dee] Add fitur baru di test 1 files changed, 1 insertions(+), 0 deletions(-)
2.	Pindah ke branch master, dan jalankan rebase git checkout master git rebase devel	First, rewinding head to replay your work on top of it... Fast-forwarded master to devel.
3.	Mari kita lihat lewat gitk dengan menjalankan perintah gitk	
4.	Sekarang pindah lagi ke branch devel, dan lakukan beberapa commit spt dibawah ini : git checkout devel echo "Fitur lagi" >> test git add test git commit -m "Implement new feature"	[devel 2d23dbf] Implement new feature 1 files changed, 1 insertions(+), 0 deletions(-)
5.	Sekarang pindahkan ke branch master dan jalankan perintah merge seperti dibawah ini : git checkout master git merge --no-ff devel	Merge made by recursive. test 1 + 1 files changed, 1 insertions(+), 0 deletions(-)
6.	Sekarang mari kita lihat hasilnya secara visual dengan mengetikkan perintah gitk seperti dibawah ini: gitk	

Kesimpulan

Gunakan rebase untuk sinkronisasi antar private branch saja, dan gunakan merge untuk penggabungan ke branch utama.

Mengelola Pararel Development

Penjelasan

Pengelolaan pararel development di git dapat dipisahkan dengan cara membuat branch-branch yang terkait dengan topik dari development itu sendiri. Pada umum-nya buatlah 3 branch yaitu master, devel/next dan bug-fixing. Pembagian ini tujuan-nya adalah agar source code pada topik development dan topik bug-fixing bisa dipisah dan dapat berjalan berbarengan.

Panduan

No	Aktifitas	Hasil
1.	Buatlah branch devel berdasar dari branch master. git checkout -b devel	Switched to a new branch 'devel'
2.	Buatlah branch bug-fixing berdasar dari branch master git checkout -b bug-fixing	Switched to a new branch 'bug-fixing'
3.	Lakukan beberapa bug-fixing dan commit ke branch bug-fixing touch bug-fixing git add bug-fixing git commit -m "Bug fixing for bug #123"	[bug-fixing 99f2d8c] Bug fixing for bug #123 0 files changed, 0 insertions(+), 0 deletions(-) create mode 100644 bug-fixing
4.	Merge branch bug-fixing ke branch master git checkout master git merge --no-ff bug-fixing	Switched to branch 'master' Merge made by recursive. 0 files changed, 0 insertions(+), 0 deletions(-) create mode 100644 bug-fixing
5.	Lakukan penambahan fitur dan commit ke branch devel echo "Added function b" > test git add . git commit -m "Implement fitur b"	[devel 9d128b1] Implement fitur b 1 files changed, 1 insertions(+), 0 deletions(-)
6.	Merge branch devel ke master git checkout master git merge --no-ff devel	Switched to branch 'master' Merge made by recursive. test 1 + 1 files changed, 1 insertions(+), 0 deletions(-)

Kesimpulan

Dengan memisahkan branch seperti diatas, kita tetap dapat melakukan monitoring terhadap branch bug-fixing maupun pada branch devel dengan mudah.

Release Management

Penjelasan

Untuk memudahkan melakukan release, usahakan buat sebuah branch tersendiri yang bebas dari proses development dan bug-fixing. Branch ini bisa menggunakan branch master ataupun branch baru. Dan lakukan merge ke branch ini hanya ketika akan melakukan release saja.

Panduan

No	Aktifitas	Hasil
1.	Asumsikan kita mempunyai 3 branch, dan cek dengan perintah sbb : git branch	bug-fixing devel * master
2.	Jika ingin melakukan release, tambahkan keterangan commit dengan mengupdate file ChangeLog seperti dibawah ini : echo "Release 0.1" > ChangeLog git add ChangeLog git commit -m "Adding Changelog for release 0.1"	[master 1174a6f] Adding Changelog for release 0.1 0 files changed, 0 insertions(+), 0 deletions(-) create mode 100644 ChangeLog

Kesimpulan

Dengan membuat branch untuk release tersendiri, ini akan memudahkan kita untuk mengambil source code sama persis pada waktu release tersebut.