Git

Kaynak Kod Yönetimi (SCM)

Kaynaklar

Vogella Git Tutorial

http://www.vogella.com/articles/Git/article.html

Pro Git

http://git-scm.com/book

Top 10 Git Tutorials for Beginners

http://sixrevisions.com/resources/git-tutorials-beginners/

Nedir

Linus Torvalds sinirlenmiş yazmış.

Dağıtık versiyonlama sistemi.

Açık kod dünyasında çok popüler. (Github, BitBucket, Google Project Hosting vs.)

Alternatif dağıtık sistemler mevcut (Mercurial)

For the first 10 years of kernel maintenance, we literally used tarballs and patches, which is a much superior source control management system than CVS is, but I did end up using CVS for 7 years at a commercial company and ***I hate it with a passion.***

When I say I hate CVS with a passion, I have to also say that if there are any SVN users in the audience, you might want to leave. Because my hatred of CVS has meant that ***I see Subversion as being the most pointless project ever started****. The slogan of Subversion for a while was "CVS done right", or something like that, and if you start with that kind of slogan, there's nowhere you can go. There is no way to do CVS right.*

Fark

Dağıtık, herkeste tüm versiyonlar var

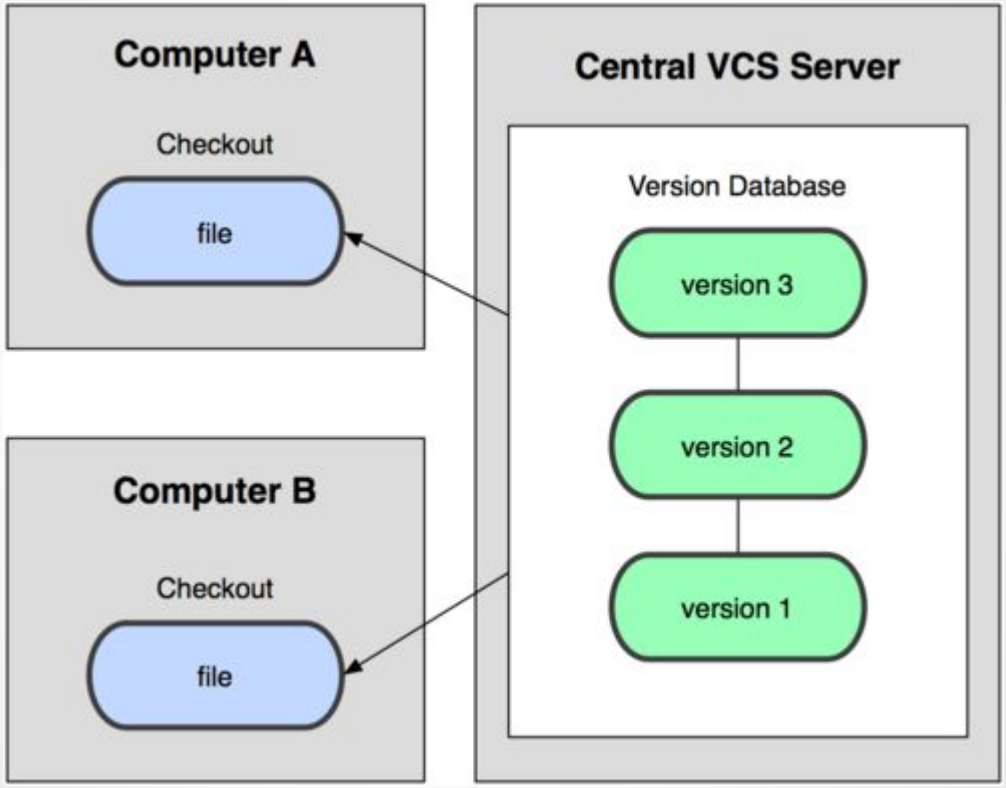
Çevrim dışı çalışma mümkün

Kolay branching-merging

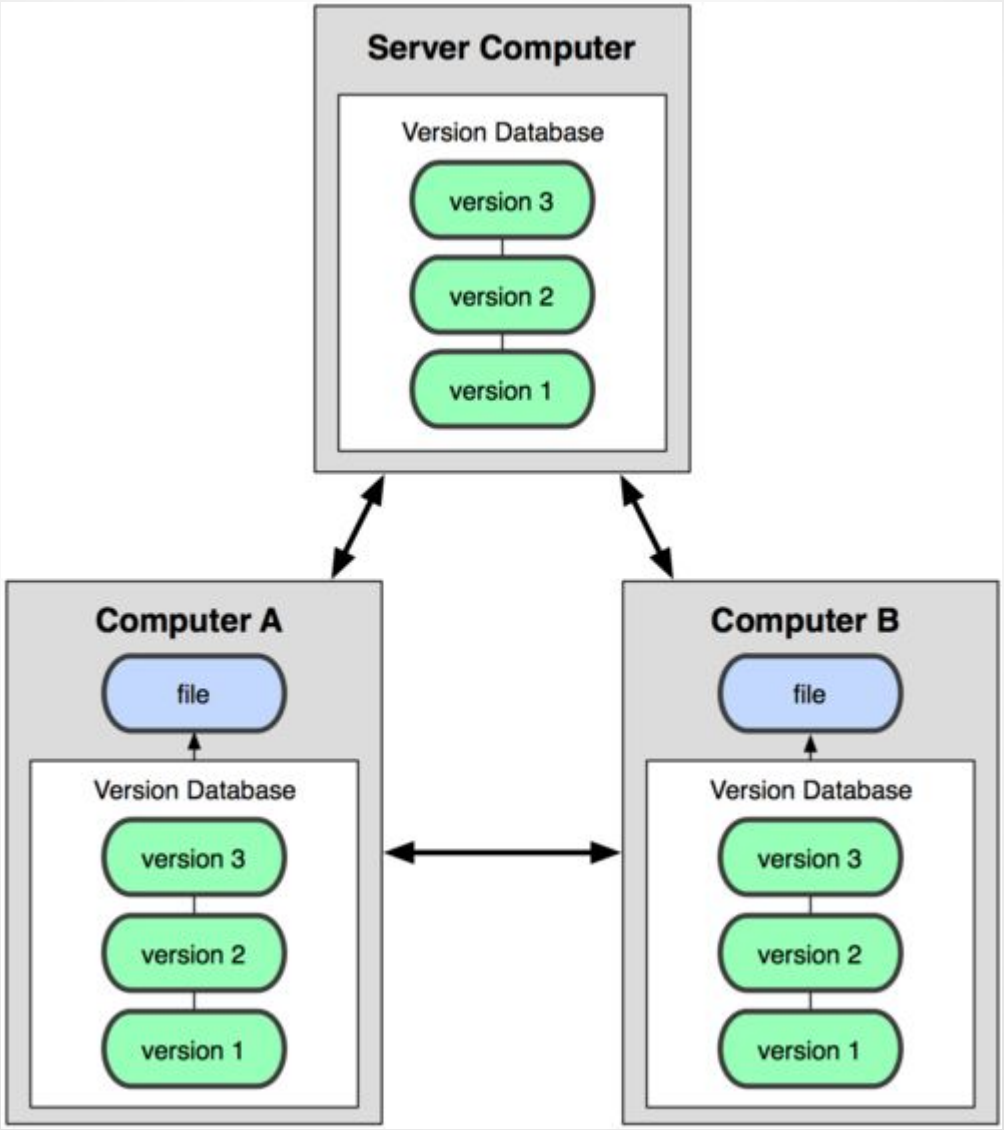
Çok hızlı

Öğrenme eğrisi

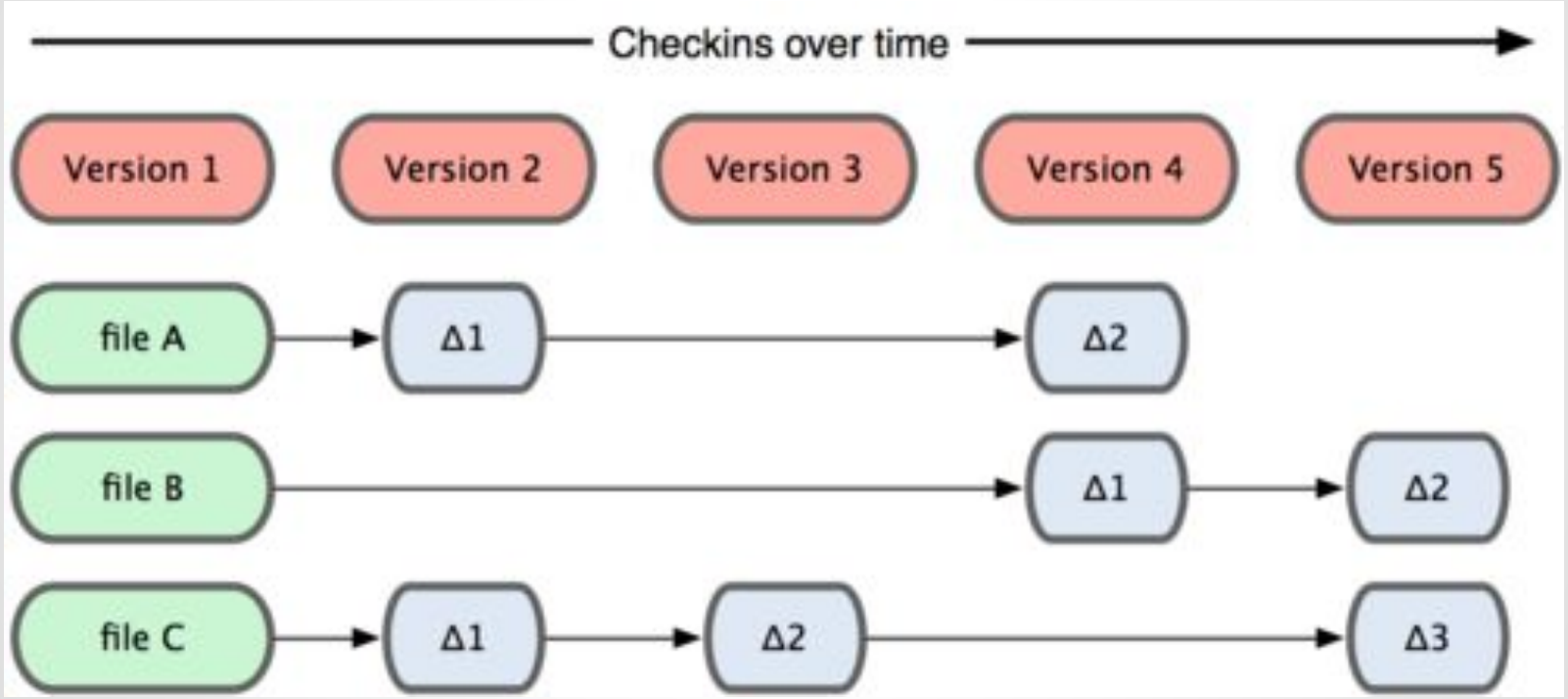
Merkezi sistem



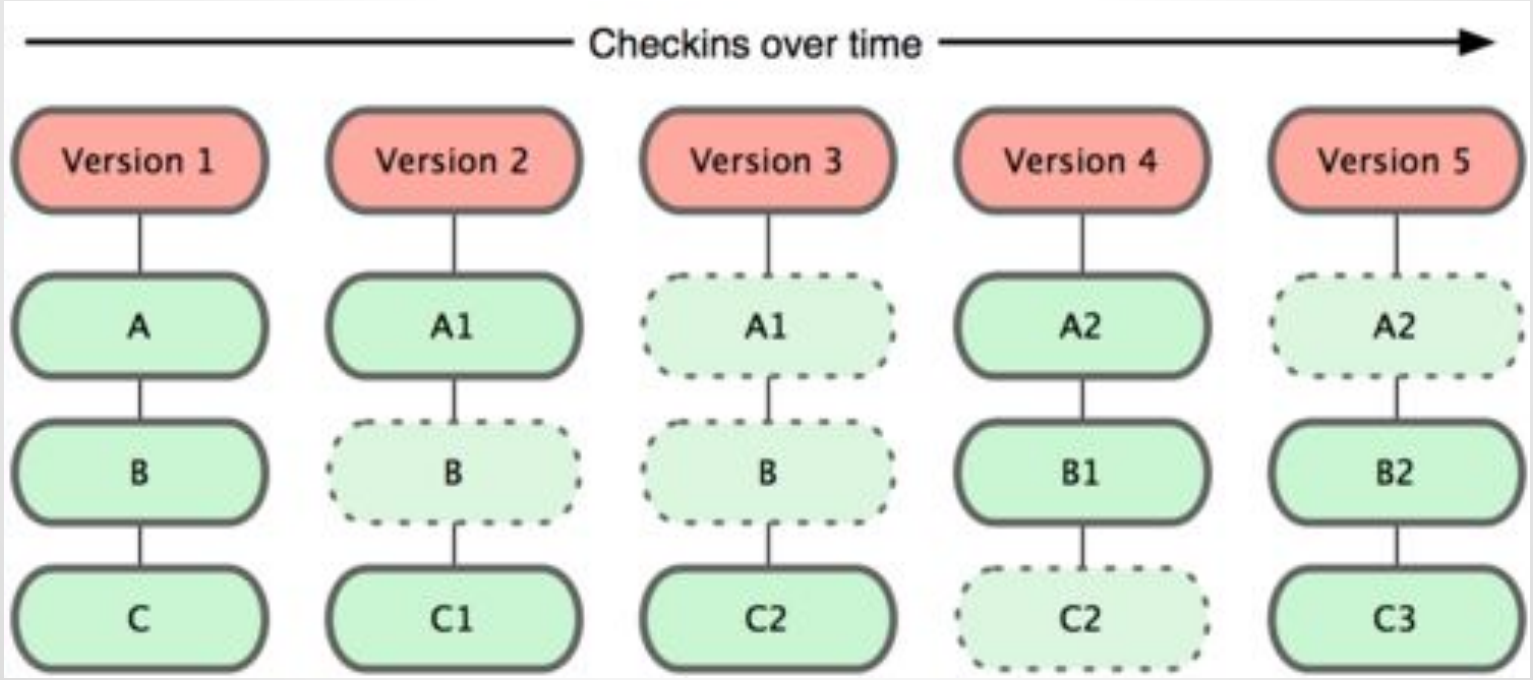
Dağıtık Sistem



Merkezi sistem sürümleri



Git sürümleri



Üç farklı yer

Depo (Repository): Tüm geçmişi ile veriler .git dizininde sıkışmış şekilde yer alır

Çalışma Dizini (Working Directory): Depodan çıkarılır ve kullanıcı dosyalara erişir.

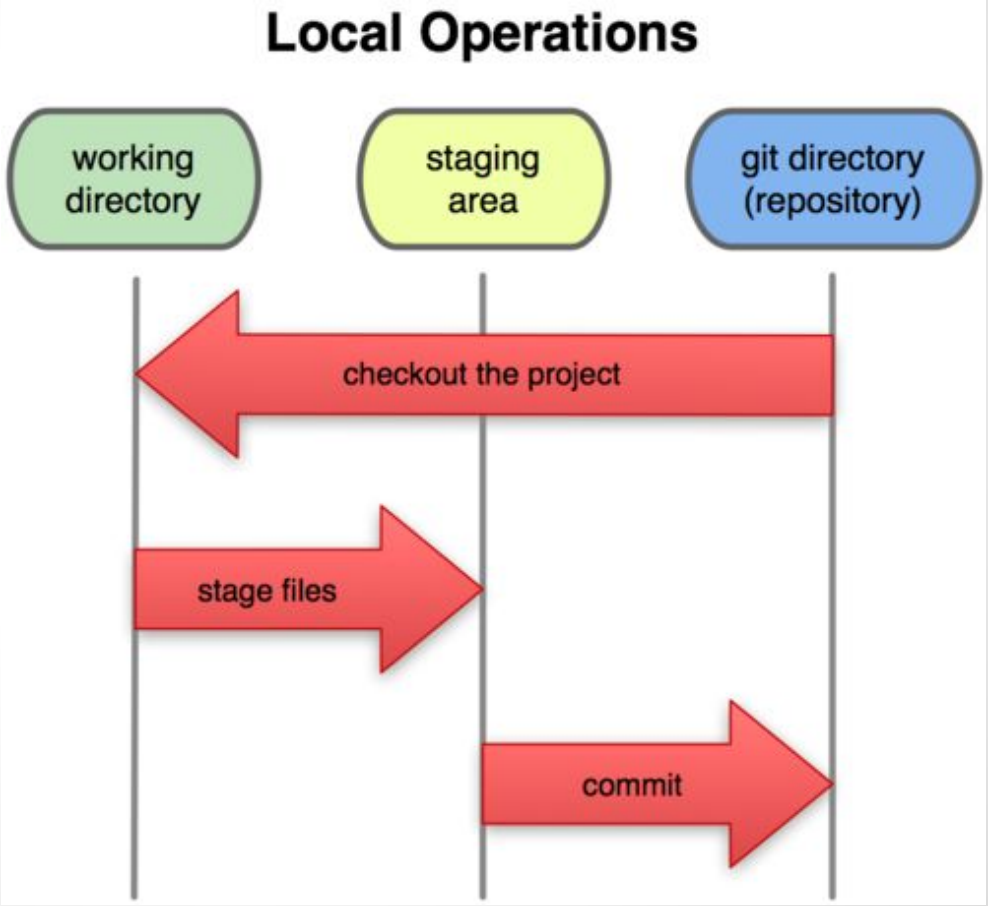
Hazırlama Alanı (Staging): Depoya gönderilecek dosyalar önce buraya alınır.

Dosyalar üç farklı durumda olabilir

Modified

Staged

Comitted



Komut Satırı

IDE'ler iyi Git desteği veriyor (Intellij IDEA, Eclipse).

Ancak her zaman GUI elimizin altında olmayabilir.

Komut satırından temel Git kullanımını öğrenmek önemli.

Kurulum

RPM Yum (CentOS, RedHat, Fedora vs.):

sudo yum install git-core

Ubuntu

sudo apt-get install git

Windows kurulum

http://code.google.com/p/msysgit

İlk konfigürasyon

```
Kullanıcı Bilgileri
git config --global user.name "Example Surname"
git config --global user.email "your.email@gmail.com"

Biraz renk
git config --global color.status auto
git config --global color.branch auto

Şu anki konfigürasyon
git config --list

Yardım
git help <komut>
```

Depo oluşturma

```
>mkdir gittest
>cd gittest
>mkdir src
>cd ..
>git init
```

Initialized empty Git repository in /home/ahmet/projects/gittest/.git/

İçerik oluştur ve depoya gönder

```
>echo hello > a.txt
>echo world > b.txt
>echo damla > src/c.txt
Dosyaları hazırlama alanına al (Staging)
>git add .

Hazırlama alanını depoya gönder. (commit)
>git commit -m "Initial commit"
[master (root-commit) fce9a51] ilk gönderme
 3 files changed, 3 insertions(+)
 create mode 100644 a.txt
 create mode 100644 b.txt
 create mode 100644 src/c.txt
```

Değişiklikleri incele

```
>echo güle güle > a.txt
> git diff
diff --git a/a.txt b/a.txt
index ce01362..2f7b5f5 100644
--- a/a.txt
+++ b/a.txt
@@ -1 +1 @@
-hello
+güle güle
```

git commit -a -m "içerik değişikliği"

-a ile sadece içeriği değişenler gönderilir.

```
[master 3197f0f] içerik değişikliği
 1 file changed, 1 insertion(+), 1 deletion(-)
```

Değişiklikleri incele - 2

git rm b.txt

git status

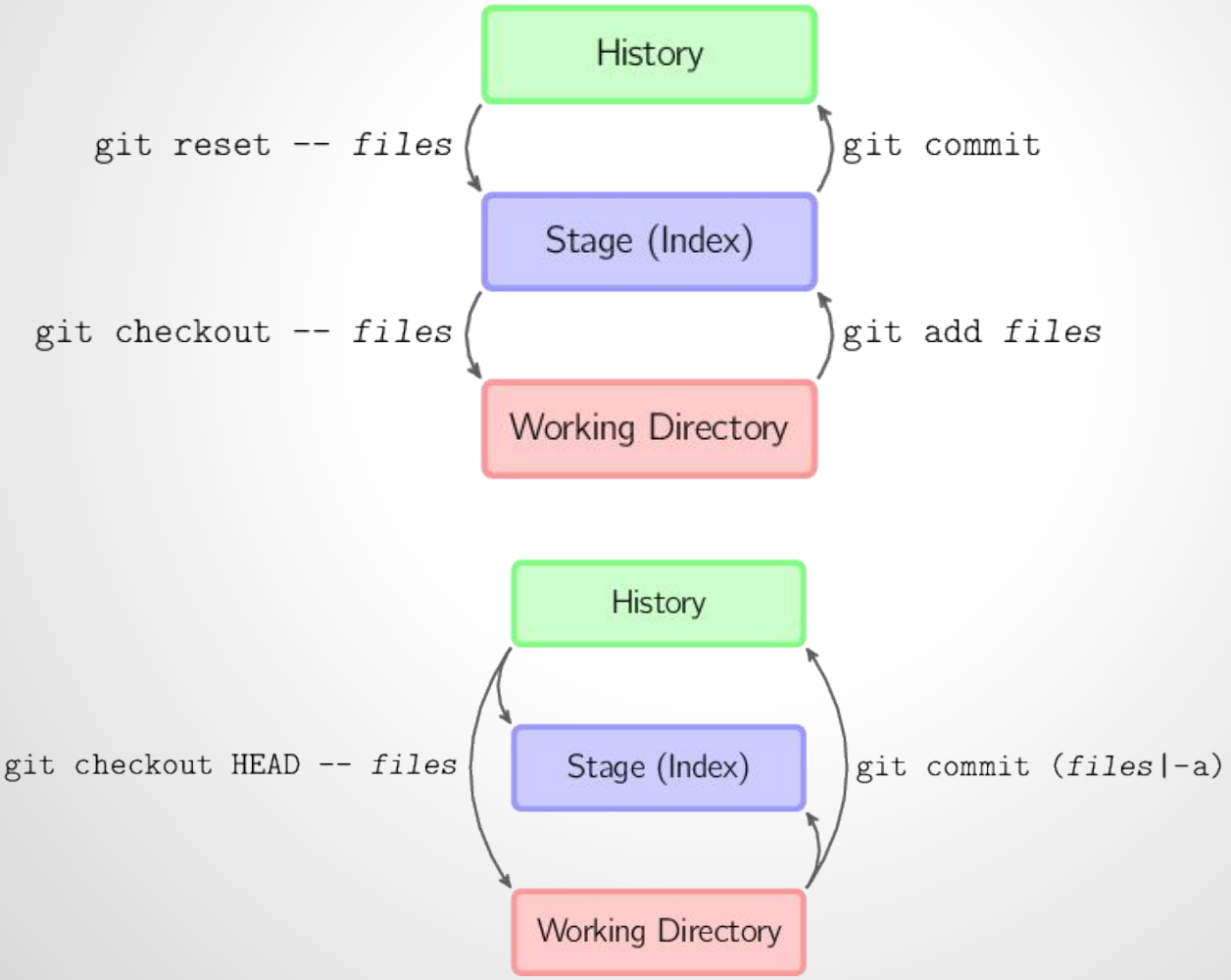
```
# On branch master
# Changes to be committed:
#   (use "git reset HEAD <file>..." to unstage)
#
#    deleted:    b.txt
#
```

git commit -m "b.txt silindi"

git log

```
commit 2b14aa76adf0d78c08c42b6504b77a1e8ef8d46f
Author: ahmetaa <ahmetaa@gmail.com>
Date:   Sat Feb 2 18:20:42 2013 +0200
    b.txt silindi
....
```

Grafik Gösterim



Depo incele

2b14aa7 nolu commit ile gerçekleşen değişiklikler

git show 2b14aa7

bir dosya için log

git log a.txt

bir dosya için log, farkları da göster

git log -p a.txt

isim değişiklikleri dahil tüm geçmiş

git log --follow -p a.txt

Kim hangi satırı değiştirmiş

git blame a.txt

Değişiklikleri geri al (Revert)

echo "gereksiz bir dosya" > amele.txt

git clean -n

ile clean komutunun ne yapacağını görürüz.

>Would remove amele.txt

git clean -f

ile gerçekten silinir.

Removing amele.txt

Versiyonlu bir dosyayı silelim.

rm test01

Geri almak için.

git checkout test01

git revert commit_id ile commit geri alınır.

Uzak Depo ile çalışma

git clone https://github.com/ahmetaa/gittest.git

```
Cloning into 'gittest'...
remote: Counting objects: 3, done.
remote: Total 3 (delta 0), reused 0 (delta 0)
Unpacking objects: 100% (3/3), done.
```

```
cd gittest
echo Hello > README.md
git commit -a -m "changed readme"
git push origin
```

Başkalarının depolarını eklemek

Varolan uzak depoları listele

```
git remote -v
```

Uzak depo ekleme

```
git remote add firatgit
git://github.com/firat/gittest.git
```

firatgit deposundaki değişiklikleri getir ama birleştirme

```
git fetch firatgit
```

Uzaktaki değişiklikleri getir ve birleştir

```
git pull firatgit
```

Dal oluşturma (Branching)

mybranch adında yeni bir dal oluşturur

```
git branch mybranch
```

mybranch dalına geçirir

```
git checkout mybranch
```

Kısa yol: mybranch dalını oluştur ve geç

```
git checkout -b mybranch
```

tekrar ana dala geçiş

```
git checkout master
```

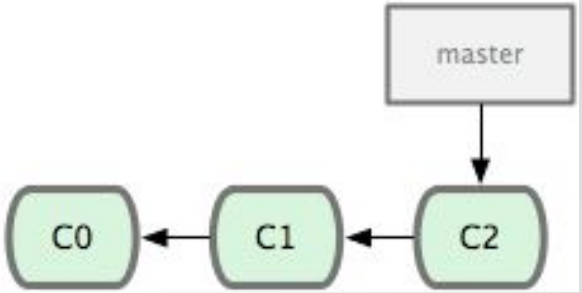
mybranch içindeki gönderilen değişiklikleri master'e birleştir.

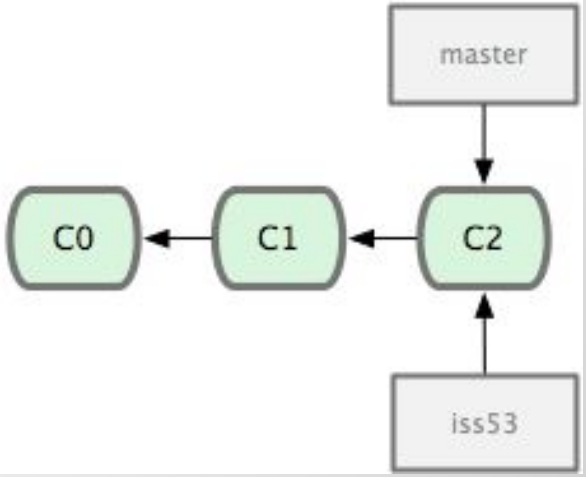
```
git merge mybranch
```

Senaryo - issue-53 çalışması

issue-53 için çalışma yapılacak

git checkout -b iss53

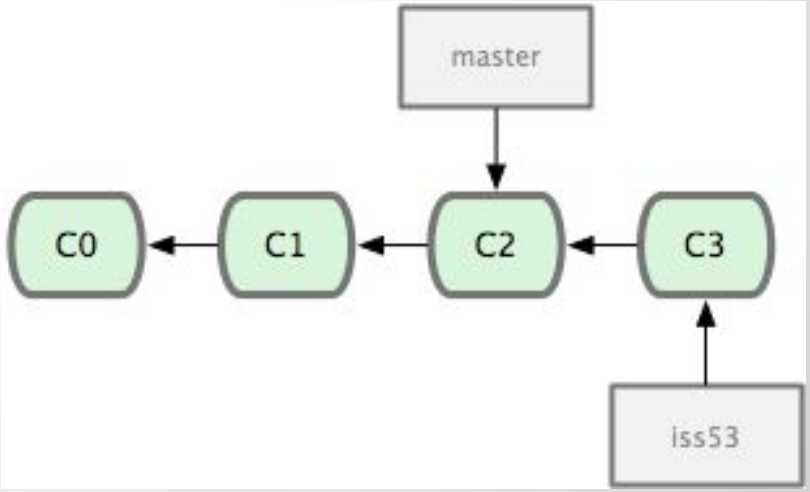




Çalışmaya devam

iss53 dalında bazı değişiklikler yapıp yollandı

git commit -a -m 'fix for [issue 53]'



Ana kodda acil değişikli isteği

Önce ana dala dön

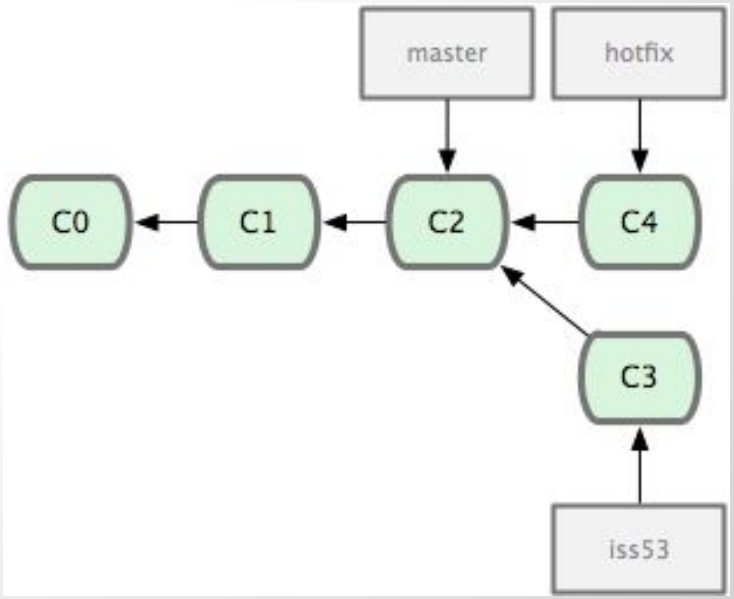
git checkout master

Acil değişiklik için branch oluştur

git checkout -b hotfix

değişiklikleri yap ve yolla

git commit -a -m 'hotfix'



Acil iş için kod birleştirme

Acil değişikliği ana dala aktar

git checkout master

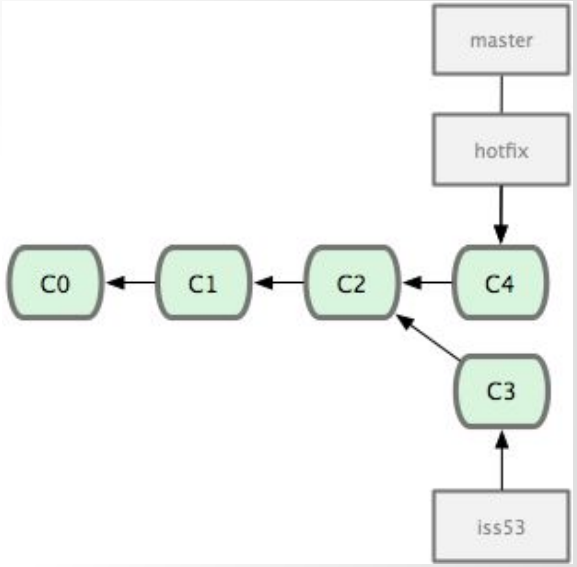
git merge hotfix

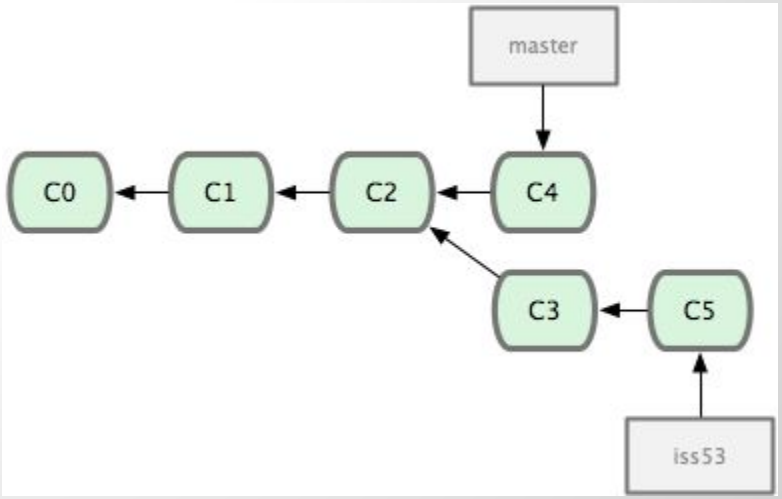
istenirse hotfix silinir

git branch -d hotfix

issue-53 için çalışmaya devam

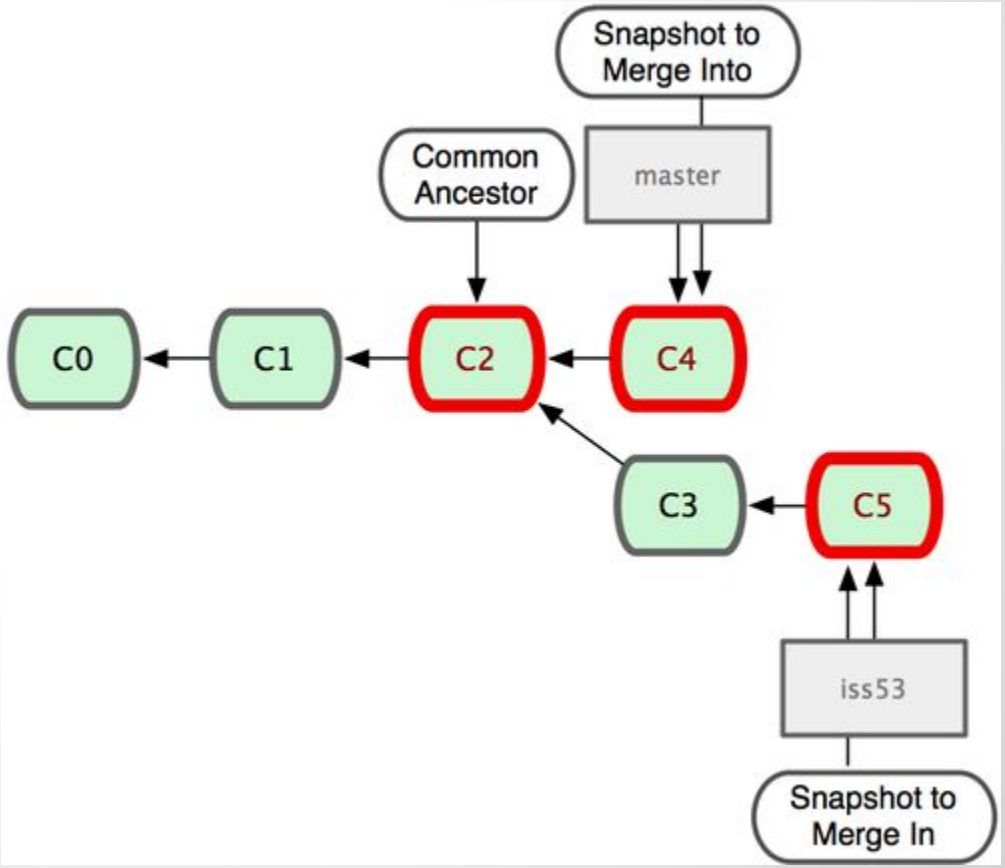
git checkout iss53





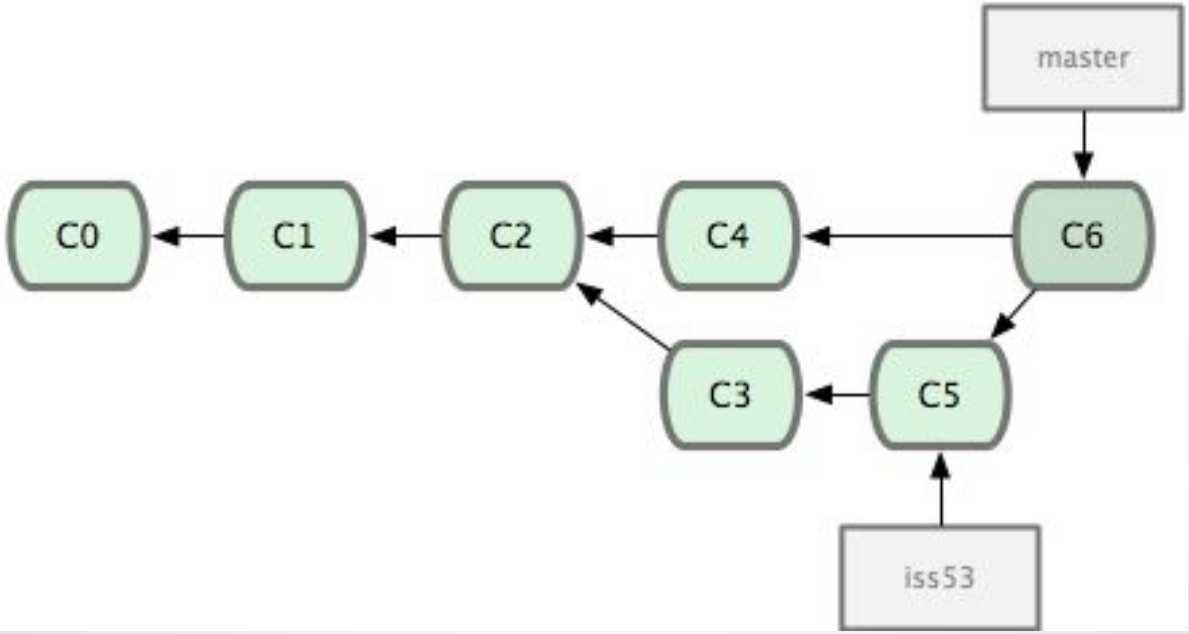
issue-53 bitti. Birleştir

```
git checkout master
git merge iss53
```



Depoda son durum

Birleşme sonunda Git yeni bir commit oluşturur. Bundan sonra istenirse **git branch -d iss53** ile iss53 silinebilir.



Uzak Branch Ekleme/Silme

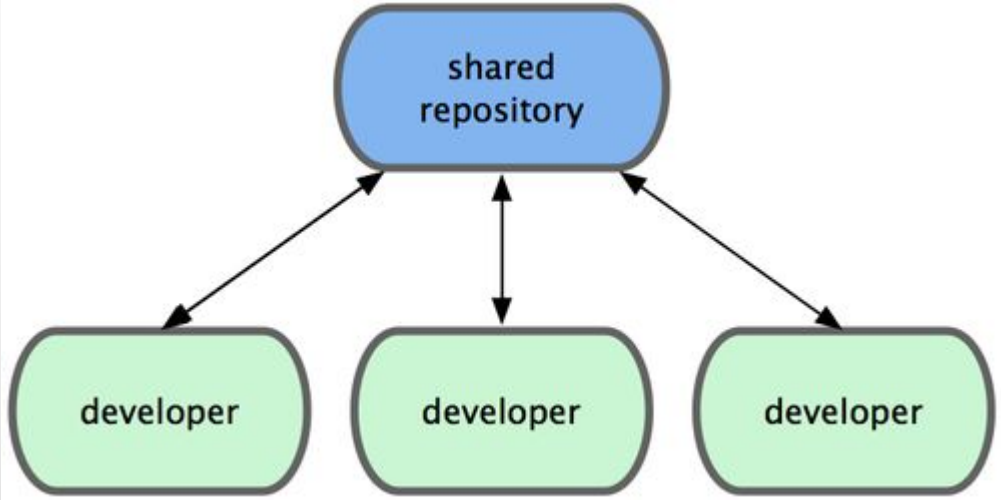
```
>git clone
http://stash.blah.in/scm/BLAH/oyun_bahcesi.git
Cloning into 'oyun_bahcesi'...

>git branch git_sunum
>git checkout git_sunum
>git push origin git_sunum

Silme
>git push origin :git_sunum  // ya da
>git push origin --delete git_sunum
```

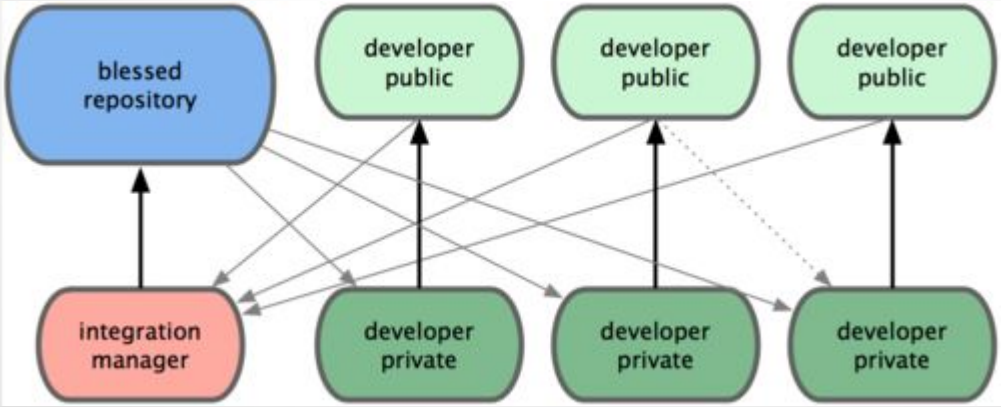
İş akış modelleri 1 - Merkezi

- Git'in SVN modeline uyarlanması
- Küçük takımlar
- SVN modeli alışkanlıkları



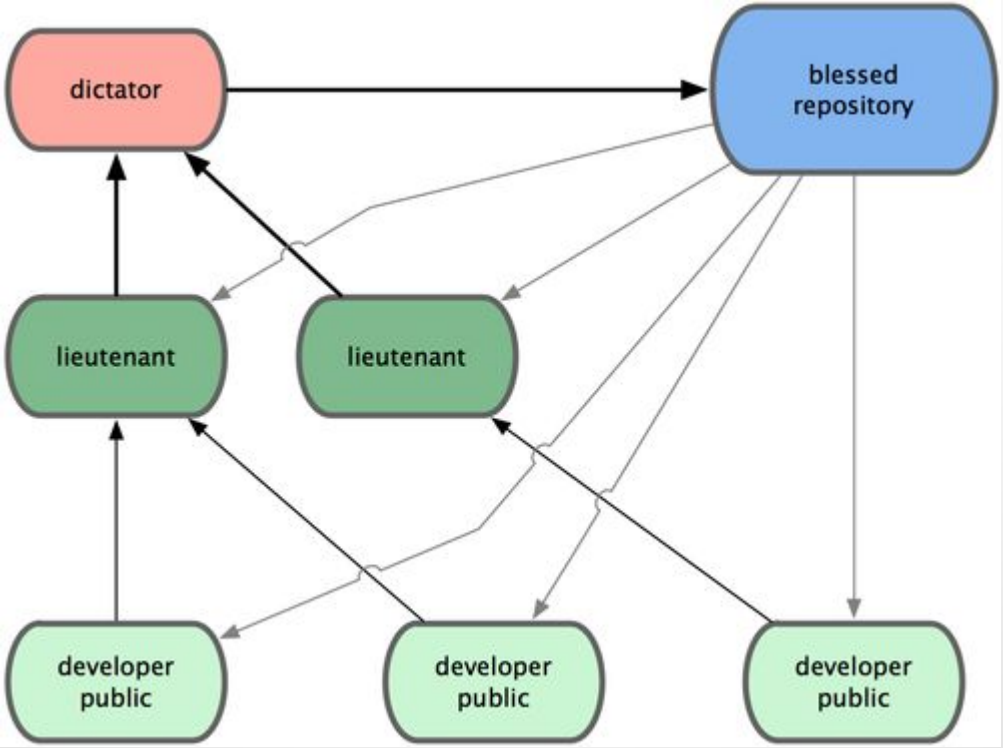
Kaynaştırma yöneticisi akışı

- Tüm geliştiricilerin yazma hakklı olan public deposu var.
- Kaynaştırma yöneticisi değişiklikleri alıp kontrol edip kutsanmış :) depoya gönderir.
- Github bu şekilde çalışır.

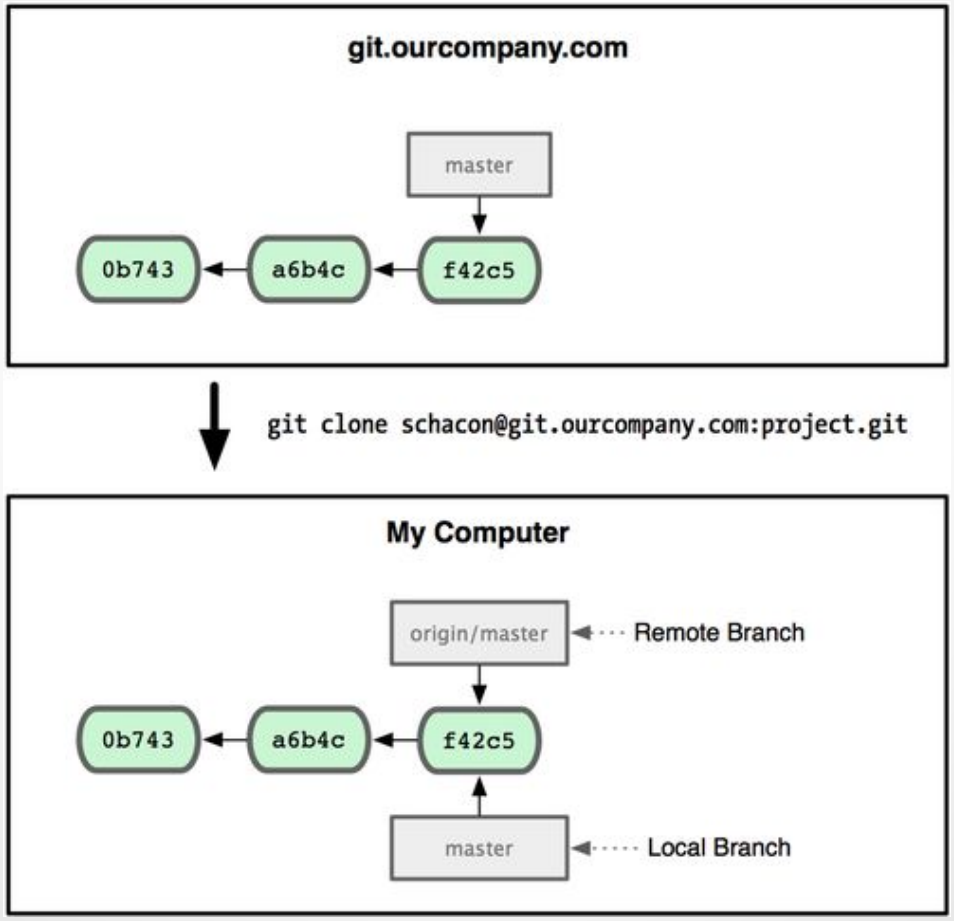


Diktatör ve subaylar akışı

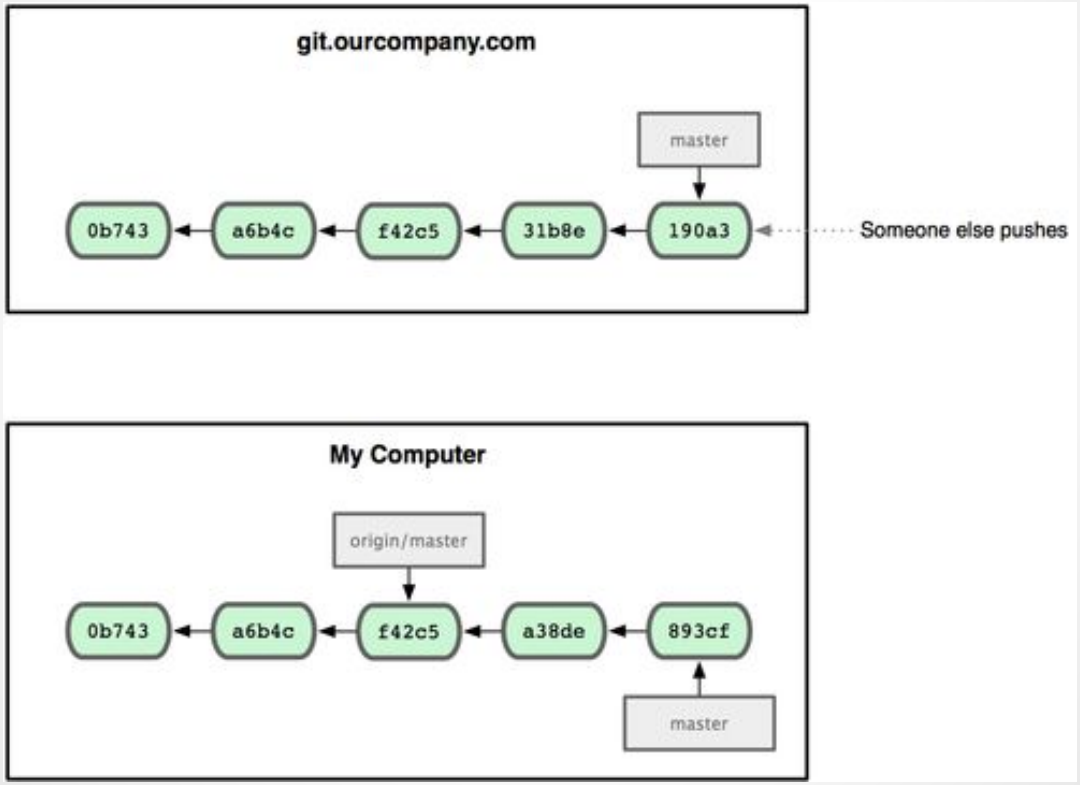
- Çok büyük projelere uygun (Linux Kernel)
- Subaylar ana modüllerden sorumlu
- Son karar paşa tarafından verilir



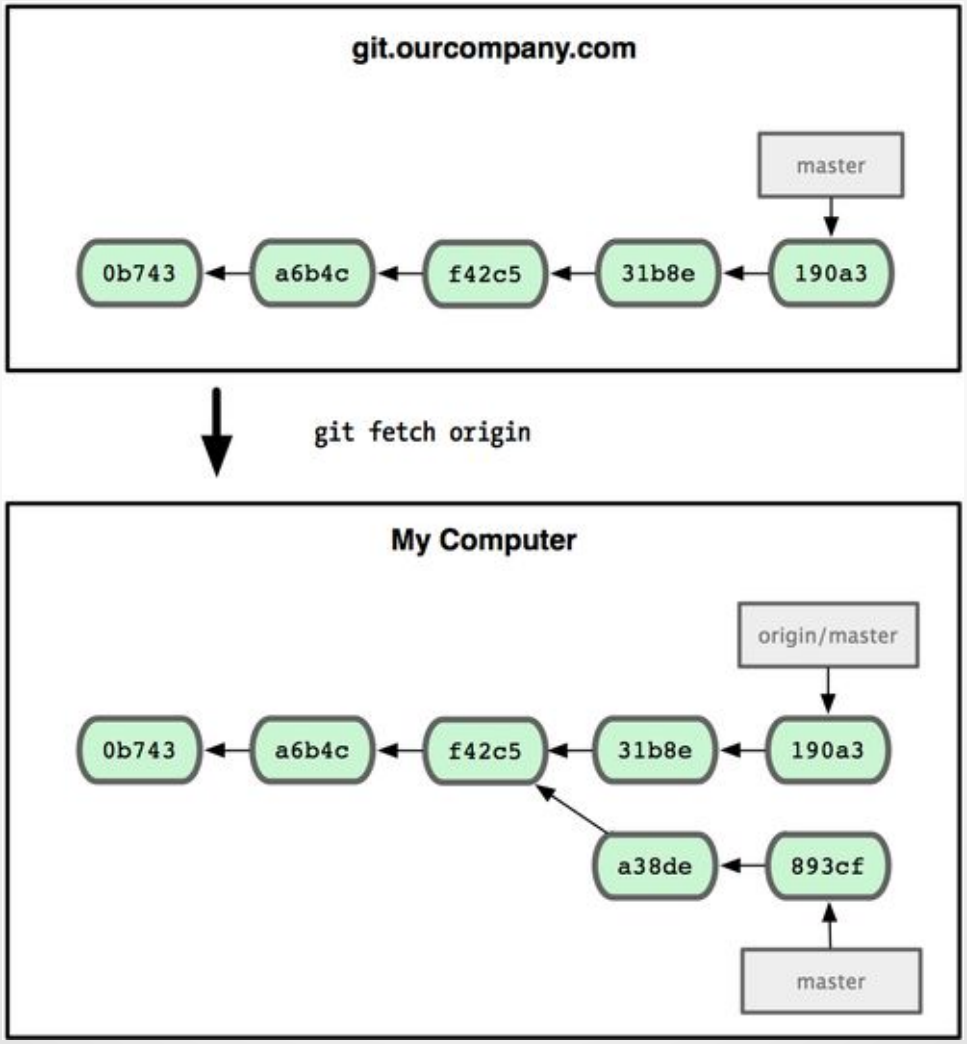
Uzak Branch Yönetimi - Klonlama



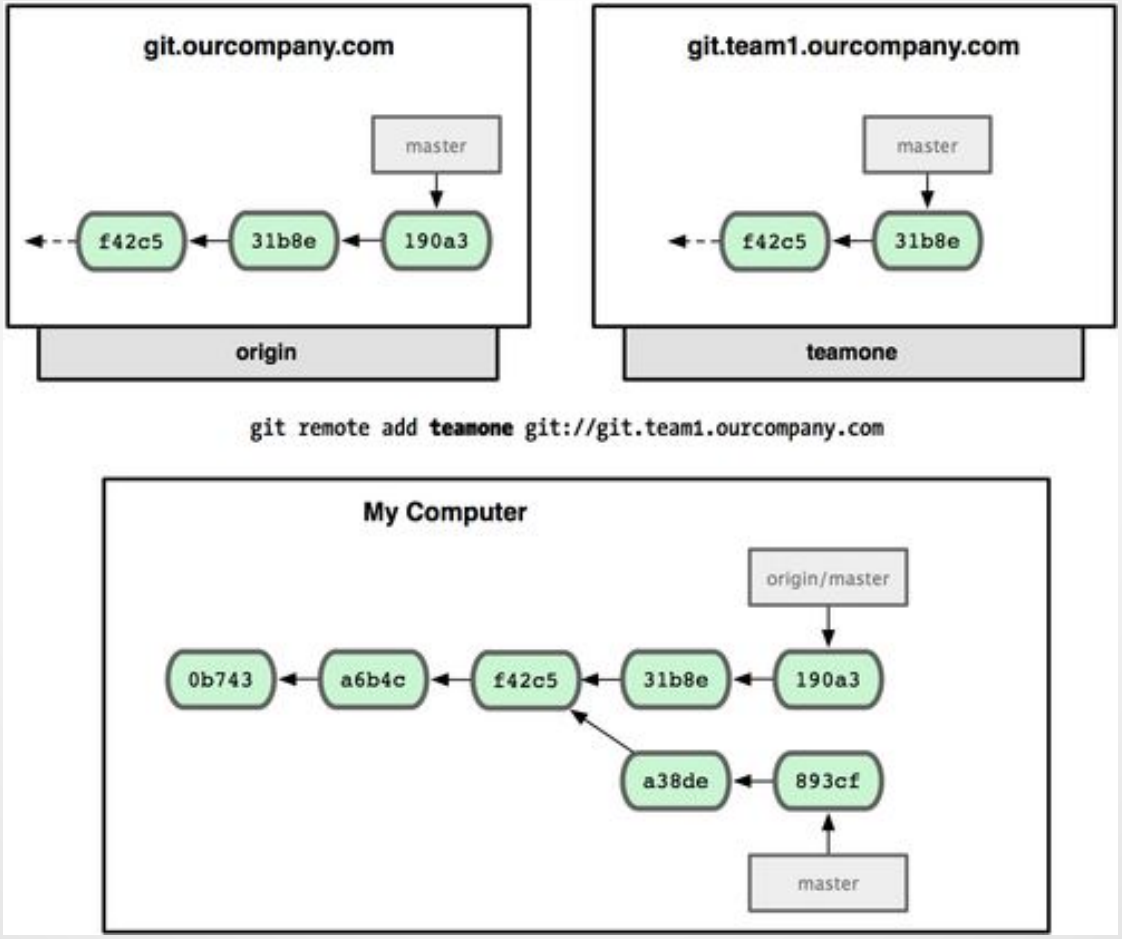
Başkası ana depoya gönderir



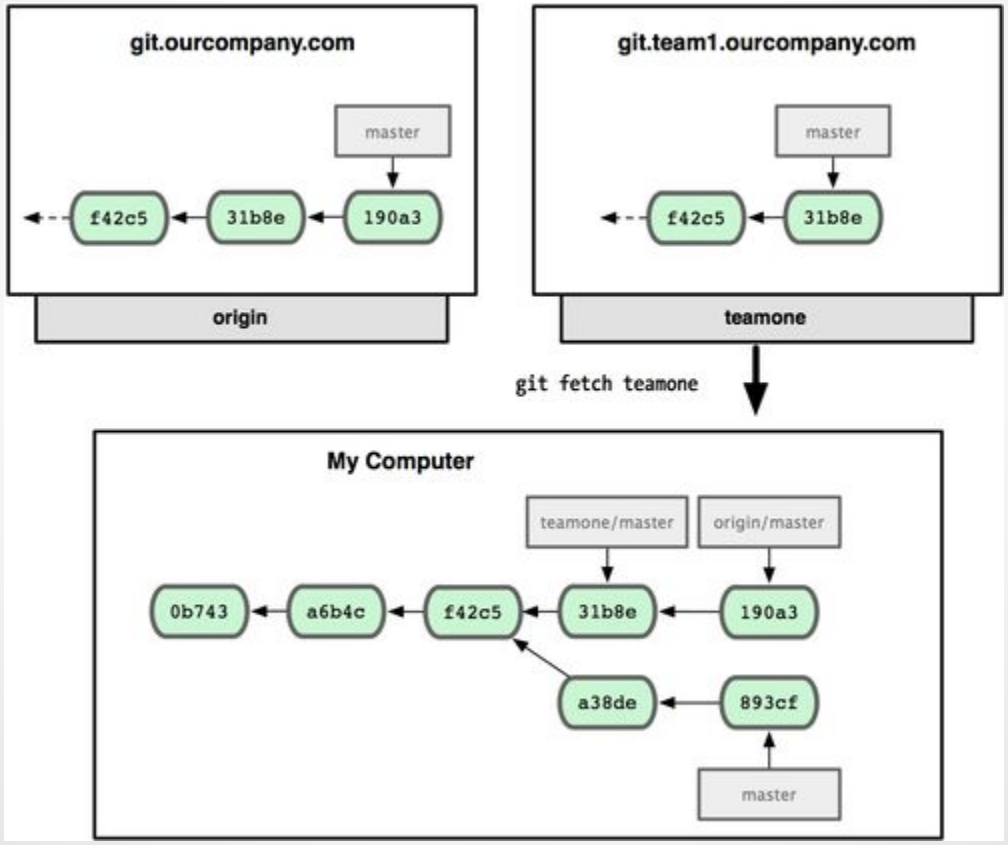
Merkezdeki değişiklikleri al.



Merkezi kullanan başka bir uzak depo ekle



Diğer depodan çekme ile yeni bir remote branch oluşur.



Atlassian Stash ile çalışma

Tarayıcı üzerinden Git proje yönetimi/gözlem

Proje oluşturma

Kod, commit, dif ve log gözlemleme

Code review