

LINUX NEDİR? | Open-source

Açık kaynak işletim sistemi. Ücretiz olduğu için Windows'a göre tercih edilir.

Why Linux?

- (*) Total cost of ownership
- (*) Beginner friendly and easy to use
- (*) Reliability
- (*) Annoying crashes and reboots
- (*) Hardware (Windows'un koşulları var.)
Linux'in çok
- (*) Software
- (*) Server segment
- (*) Linux is everywhere
- (*) Security
- (*) Freedom

* Linux Windows'a göre daha performanslı çalışır.

* Linux'da hersey açık kaynak olduğu için virüs kuyrukları. Çünkü bunu sen görsün - Windows virüsleri?

Linux'da çalışmaz. Linux'da virüsleri görebiliyor sinir.

Android telefonlar Linux tabanlı ve çok virüs var. Windows kullanıcıları, insanlara virüs gelir. Bu şebeke tercih ediliyor.

* Yeniden başlatı mesajları Windows'ta çok, Linux'ta az.

Where Linux?

⇒ Google

- Space
- Twitter
- McDonalds
- Raspberry pi
- Facebook
- Submarines
- Desktop Computing
- Amazon
- NASA
- Nuclear project
- IBM (Bilgi işlemci firması)
- Bullet trains
- Traffic controlling
- Mobile Devices
- Weapons



Open source olduğu için halkın tarafın da
düzenlebilir. Bu yolda gelişimi hızlandı.



Linux → UNIX'ten移植 olarak geliştirildi.
→ Hayırsever

Linus Torvalds → Linux'un babası (1991)

Debian → 1993

First Android Smartphone → 2008 (Linux çok hızlandı.)
Nokia battı.

Ubuntu → 2004 (Hoca bunu tavsiye ediyor. Kullanıcı
desteklemeye.)



Git → 2005 Versiyon Kontrol Sistemi

Linux'un maskotu (Penguin) → 1996

Masaüstü ortamı } G.NOME (Masaüstü, arayüz) → 1997 } Linux'un
} KDE (GNOME muadili) → 1998 } vazgeçilmezi

redhat (RHEL) → Linux'un paralel versiyonu (2002)

Versiyon Kontrol Sistemi

git \Rightarrow Degriftir digin dosyaların kaydını alır.
İstersen 15 gün önceye, bir ay önceye
gidebilirsin. (Kod yazanların facebook'u.)
 \downarrow
Github Projekti paylaşıyorlar

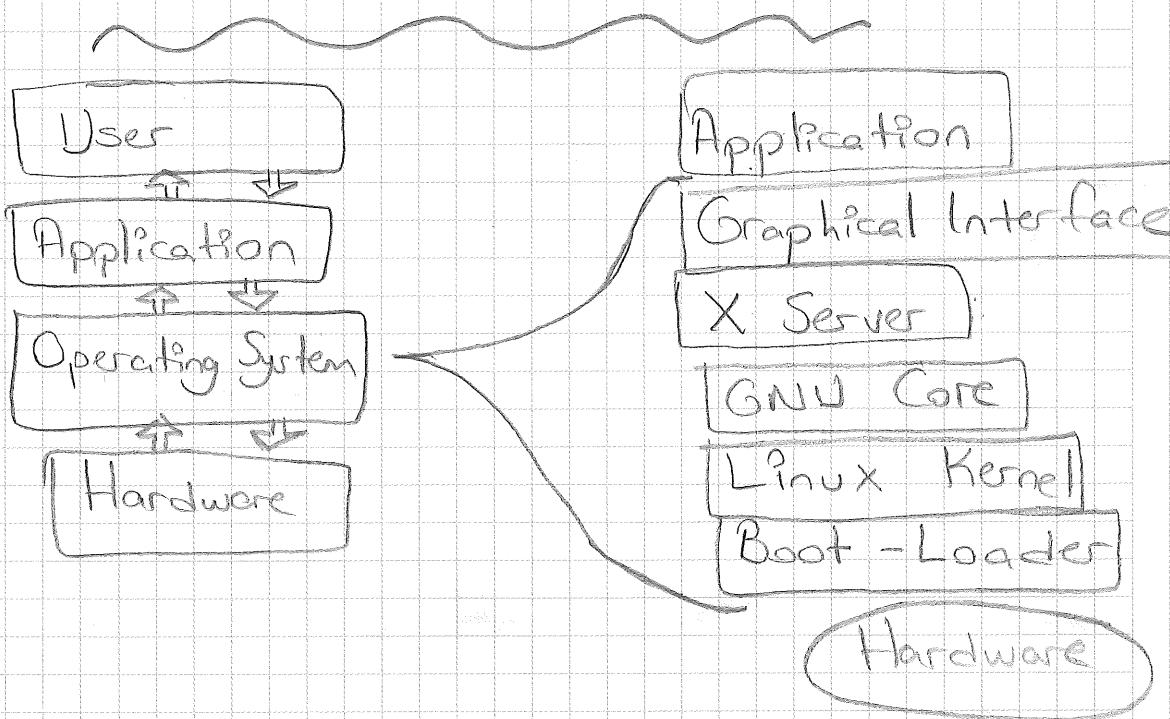
2009 → Google kendi işletim sistemini çıkardı.

* Hoca WSL'i kurmanız isteyecek (2019'da oldu.)

GIMP \Rightarrow Image Editing

* The most popular web server in Linux is \Rightarrow Apache

LINUX'UN COMPONENTLERİ



Linux - GNU \Rightarrow Kol kola çalışırlar.

İşletim Sistemi \Rightarrow Taptığın isteni alır bilgisayara ileter,
orden de sana ileter.

KERNEL : İşletim sisteminin çekirdeği
Komutları ve OS'a señor ve hardware'ı
kontroller.

SHELL : Kernel'a direkt komut vermek için
kullandığımız araç.

Popular Linux Distributions

Debian	openSUSE
Ubuntu	Red Hat
Mint	Fedora
Manjaro	

MAJOR OPEN SOURCE APPLICATIONS

Apache (http server)
Firefox
MySQL

SERVER APPLICATIONS

Apache Web Server
NGINX
MySQL
Samba
ownCloud

Conflicts occur if two people change a file at the same time. This problem can be solved in two ways:

Lock-Modify-Unlock

- Only one person changes

- A file must be locked before changing it

- Different users can't lock a locked file.

- Another user must wait for the lock to be released.

Copy-Modify-Merge

- Each user has a personal copy of the file tree

- Changes are made independently and simultaneously

- Private copies are merged into a new version.

- All other copies become outdated as soon as private copy is committed

Types of Version Control Systems

There are three types of version control systems:

- * File-based → Not much used.

- * Client-server type → Lock-Modify-Unlock
→ Copy-Modify-Merge

- * Distributed → More common

- (GIT is one of them)

- Herkes aynı dosya üzerinde de farklılık yapabilir.

- Dosya fast and it can be changed.

Modified → Depistirildi Committed → Taahhüt
Staged → Asomali

In a Git repository your file can reside in three main states:

- ① Modified (Depistirildi ama henüz veritabanında kaydetmedi)
- ② Staged (Bir sonraki işlemde gitmek için mevcut sürümden depistirilmiş bir dosya işaretlenmiştir)
- ③ Committed (Veriler yerel veritabanında güvenilir şekilde saklandı.)

This leads us to the three main sections of a Git project.

- ① The working tree.
- ② The staging area.
- ③ The GIT directory

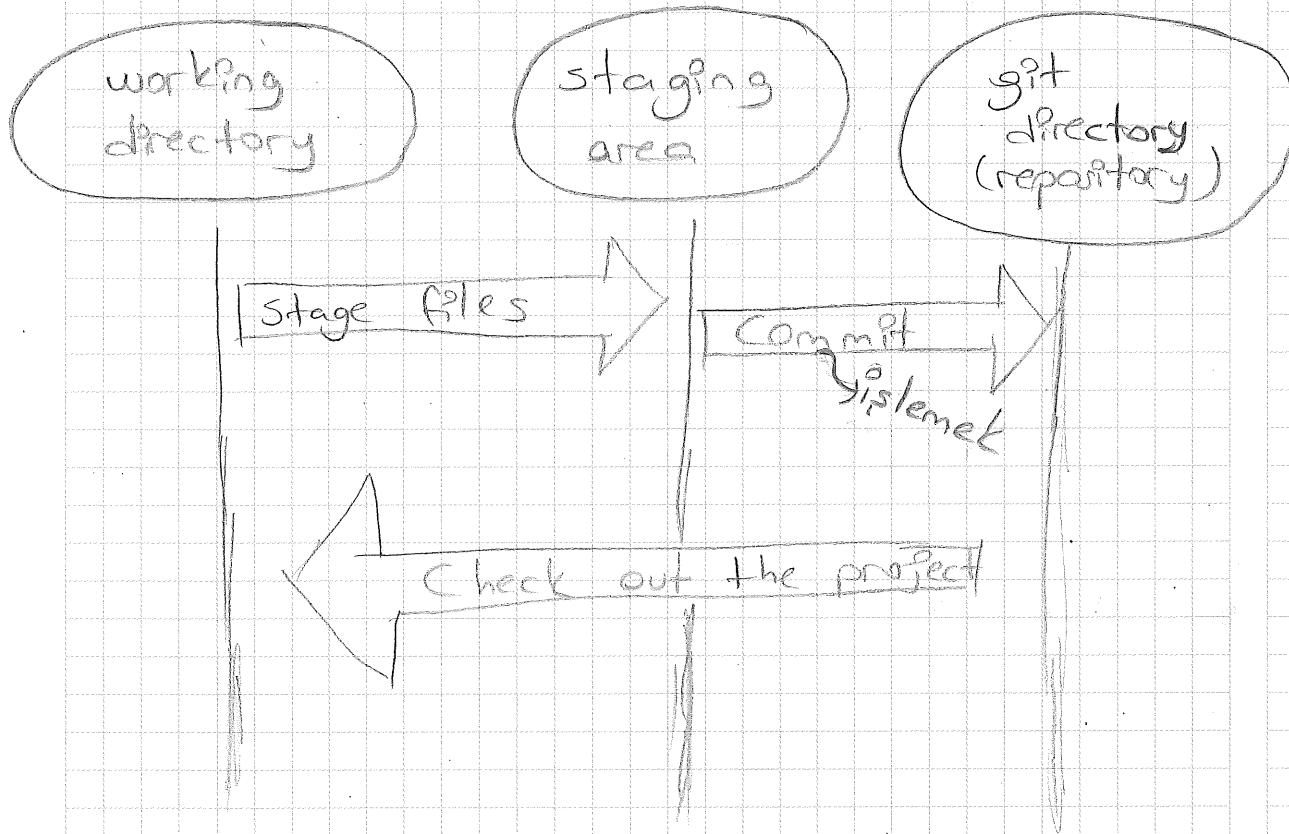
The Working Tree: Projenin bir versiyonunun tek cikisi

Bu dosyalar sikistirılmış veritabanında
cikarılır ve kullanılır veya depistirme-
riż zin diske yerleştirilir.

The Staging Area: Genelde Git dindinde bulur.

↓
Bir sonraki taahhütünüzde nelerin
göntereği hakkında bilgi depolar

The Git Directory (.git) \Rightarrow Git'in projenin meta
verileri ve nesne veritabanının
depoladığı yerdir.



GIT COMMANDS

git init \rightarrow (New local repository)

git add.

git commit.

git status.

git config.

git branch.

git checkout.

git merge.

Shift + Tab

* How many states does GIT have?

- ① Modified
- ② Staged
- ③ Committed

* What is repository?

Directory or storage space where your project live.

ends with \Rightarrow True veya false dir.

Root ACCOUNT

Pislettiler oluttur? (simrisiz piletisim yetkisi?)

CURRENT LESSON | 03.07.11

(VCS)

* What is Version Control?

Yapılan değişiklikleri kaydetmek,

Değişiklikler arasında mühakeme etmek
Değişiklikler arasında genis yapılabılır.

* GIT nedir? → Bir tool'dur.(Araç)

Bir VCS' dir.

Piyasada en çok kullanılan tool.

Free ve open-source olduğu için tercih ediliyor.

Linus Torvalds oluşturdu.

İleri ve geniştirilebilir.

Yaptığın değişiklikleri Kayitlar ve bu değişiklikler içinde
gezilebilirsin.

Distributed' dir. (Merkezi değildir.)

VERSION CONTROL SYSTEM

- Track changes
- Unlimited (Undo / Redo)
- Time Travel
- Collaborative development environment (Aynı anda birden fazla kişi çalışır.)
- Compare and Blame

What changed?

When it changed?

Why it changed?

Who changed it?

VERSION CONTROL SYSTEM

CENTRALIZED

VCS

DISTRIBUTED

You need to be connected
to the server

Gök tercih edilirler.

You can work while offline



Internet olmasa bile kullanabilirsin.
Kaynak dasyalar herkese açık olur.

- Tracks and records changes to files over time.
- Can track any type of file, but most commonly used for code.
- Can retrieve previous version of files at any time.
(Geç dönen bilgisi)
- Retrieve files that were accidentally deleted.

GIT → VCS'ın bir tool'udır.

Git is a software

Content Tracker

Distributed Version Control System(VCS)

Linus Tornalds

Why do we need GIT?

Backup / Archive / Versioning / History

Undo Changes

Comparing

Code Review

Blame

Git REPOSITORY

Kayıtları tutan klasör \rightarrow REPOSITORY (Repo, depo)

A directory or storage space where your projects can live



Local Repository
(internettez)

Remote Repository
(internette)

! Linux'ta kullanılan bir text editörüdür.

To create a new local repo

`git init`

\rightarrow Takibi başlat

To see the commands

`git help`

To see the status of your repo

`git status`

\rightarrow VCS yapılış dosyanın durumunu gösterir.
(Commit yapıldı mı / yapılmadı mı)

Git CLONE ADDRESS

To create a new remote repo and connect it with your local repo (after you create a remote repo on Github / Bitbucket etc.)

git init → yeni dosya olustur → git add

git commit -m "ilk commit"

(kalıcı olmasi için)

WORKFLOW

Working Directory
(untracked file)

VCS tarafından takip ediliyor.
Edilmesi için
git add ile takip edilebilir.

Aliş-veriş arabası

Starting Area
(Index)

Merket arabasını da hürdün.

ama dylem kılınan sıkışmadı.
o zaman repository'ye atıwasın.

Giriş alanı gibi.

Kayıtların tutulduğu
git uzantılı klasör

Repository

FILE STAGES

COMMITTED → Unmodified changes from the last commit.
snapshot.

f1.txt

01.py

.git

Bu bitim
repository

TAKİBE DOSYA EKLE, !TAKİP !

git add filename \Rightarrow Dosyaları tek tek takibe eklemek için
git add . \Rightarrow Dosyaları toptan takibe eklemek için

* git commit -m "first commit"

* git commit (Değişikliği kim ne zaman yaptığı?)

* git checkout (Silinecek dosyaları geri alma)

* git commit -m "message" (Korsan tarafı bilgilendirmek veya sonradan takip etmek için)

* git commit -am "message" (Yaptığım değişiklikten eminim)
message
(Staging area'da atmadan direkt repository'e at)

* git commit --amend (Son commit'te hata yaptıysam bunu düzettirmeni sağlar.)

YENİ KLASÖR

● git init (Takibe başla)

touch file1.txt nem yaz

vim file1.txt ne yazın yazın [? yaz :wq qit]

● git add.

● git commit -m "message"

git status \Rightarrow Takip edilen dosyaları gösterir.

git log \Rightarrow Yaptığınız commitleri gösterir

echo \Rightarrow Bir seyin ismine direkt birşeyler yazılır.

git checkout (ilk basi veya hani)
eklenen? gen? alt?

TASK - 2

Create a file named file2.txt

Edit file2.txt

Stage

Delete the file file1.txt

Rename file2.txt > file3.txt → File2 - File3 yap

Stage file3.txt

Unstage file3.txt

Stage file3.txt

Commit the file to your repos

Change the message of the commit

Switch back to your first commit in Task - 1

ANSWER

- ① Dosyamın içinde bir file dosyası oluştur.
- ② git init (Takip başlat)
- ③ git add .
- ④ file 2 'yi file 3 olarak değiştirdi dosyada
- ⑤ git add . (file 3 eklandı) (Bütün nesneler eklenerek işin)
- ⑥ git rm --cached file3.txt
- ⑦ git add .
- ⑧ git commit -m "ilk commit"
- ⑨ git log (Birim bulundugumuz konumdan önceki gösterisi)
- ⑩ git checkout commitID (Komitler arasında geçişmek işin)

Kernel : Kullanıcı ile hardware arasındaki köprü^{lu}
İsletim sisteminin kalbi.

* Linux componentleri?

GUI

Hardware

KAHOOT

* Linux'ı distro's?

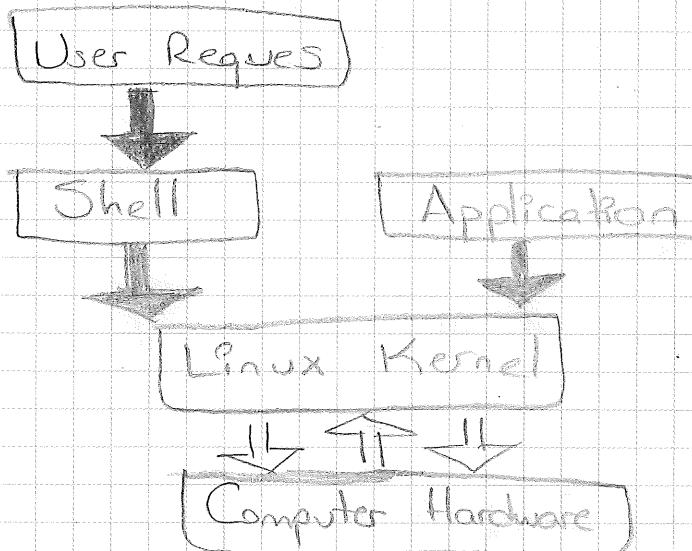
RHEL

Ubuntu

Mint

WHAT IS SHELL? → Yazılım dilidir.

Unix - Kernel'a komut verebil diliginiz programdır.



Kernel bir yazılımdır.
Her işletim sisteminin
kendine has bir
KERNEL'ı var.

Shell bizimle - Kernel arasındaki köprüdür. Biz
direkt Kernel'e komut veremeyeziz.

En çok kullanılan Shell → Bash

(Bourne Again Shell)

\$ ➔ Kısıtlı kullanıcı olduğumuz anlamına gelir
(Kendi yetki alanın dışında işlem yapamazsın.)

root ➔ Her yerde işlem yapabilir. Her seye müdahale edebilir.

user @ clansway - Linux: ~ / test \$
↓ ↓ ↓
User name host name normal user

| GUI |

Görsel kullanıcı ara yüzüdür. (Mouse simgesi, tık kutusu...)
GUI sayesinde olur.

! Which one is not a user interface of Linux

- a) GNU (İşletim sistemi)
- b) CLI (Komut satırı arayüzü)
(Komuta Line)
- c) GUI (Görsel kullanıcı ara yüzü)

| BASIC SHELL COMMANDS |

pwd ➔ Nerde olduğumuzu gösterir. (Bulundığınız klasör)

explorer.exe ➔ Dosyayı saklı diye açıyor.
(Bulundığın klasörün penceresi)

ls ➔ Klasör içindeki dosyaları listeler. (Gizl olmayanlar)

ls -l ➔ Detaylı listeler.

ls -a ➔ Gizli dosyaları da gösterir.

ls -al ➔ Hem gizli hem de detay gösterir.

cd / \Rightarrow Windows'ta ana klasör olan
C'ye konuluk gelir. Bu'nun dosyaları orda
tuflur.

* cd [dir] \Rightarrow Change director

cd.. \Rightarrow Bir üst klasöre çıktı.

cd / \Rightarrow Ana dosya sistemindeki en tepedeki klasördür.
Bütün klasörler ona bağlı, herkes ona bağlı.

cd - \Rightarrow Bir önceki klasöre git. Windows'ta \leftarrow geri ok
tuşuna tekabül eder.

* mkdir [dir] \Rightarrow Create a new directory
Dosya yap

rmdir [dir] \Rightarrow Boş klasör silme.

* touch \Rightarrow Create a file (Windows'ta sağ-ken metin
yeni birsey)

* rm \Rightarrow (remove) Bir dosyayı siler

Bosluk bırakıp yanına yazıp birden fazla
dosya silinebilir.

* cp \Rightarrow copy

uzantılı dosya yapar.

(*) touch folder1 / newfile

Dosyanın içine girmekten yeni file yapmak için

(*) cp folder1 / newfile folder2

Folder2'ün içine folder1'i kopyalar.

cp folder1 / newfile folder2 / newfile2

Buyle dersin içinde folder2'nin içinde folder1'i kopyalar ama ismini değiştirdi. newfile2 yapar.

(*) mv ⇒ (move) Bir dosyanın başka yere taşımak.

J. copy ile benzer Dosya ismini değiştirmek için de kullanılır.

(*) cat ⇒ Dosyaların içeriği okunabilir.

(*) echo ⇒ Ekrana yazdığını yazdırır. Aynen. Ekraya yapar.

echo Hello] Dosyanın içine metin yazdır.
→ Hello

Case Sensitivity

Windows case sensitive değil A=a

Linux case sensitive. A≠a

Wildcard Character

Bir karakterin yerini tutar

? → Question mark Match any single character

* → Asterisk

Match any number of character(s)

→ Birde çok karakterin yerini tutar.

da!

BRANCH , HEAD

Branch

Yan kol , uygulamaya girmeden yan kolu de-
gistirmek.

Ayn proje içinde birden fazla kipi kullanmak
kullanılır.

HEAD → Sen su an burdasın demek
Sen hangi branch de salisyonan head onu
isaret eder.

MASTER → Ana branch'ıdır. (Ana dal)

git branch → Hangi branch 'de olduğumu görmek

git branch -r → "

git branch -a → Her ikisi deki branch görmek

git checkout → Branch lar arası geçiş. (Branch değiştir.)

git checkout -b → Branch oluştur ve ana git.

git branch -d → Branch silme

git commit -m → Yeni commit oluşturulur

FAST FORWARD MERGE

Merge: Dal içerisinde yani bir dal aynılmış sonra ikinci dal tekrar birleşmiş. Buna merge atıf denir (IKİ dali birleştirme).

Merge etmek \Rightarrow Birleşirmek

Merge - Conflict

Dökümanın ilk kısmını ben değiştirdim, arkadaşım da başka yeri değiştirdi. Bunları birleştirmek istediklerinde hatalı oluyor. Bunu proje yöneticisi birleştirir.

Yazılımda bulundığın dali silmemesin. Mesela

git checkout master (başka dala geçti)

git branch -d yeni-dal2 (siliyor)

git branch ile sildiğimi kontrol et

git checkout ile yeni dala geç

$\text{git pull} = \text{git merge} + \text{git fetch}$

| GIT HUB |

GIT

Version control system

GITHUB

Repository hosting service.

* Birçok kişi aynı çalışma dosyalarını saklayorsa github repo oluşturur. Kendi reporu oluşturabilirsin.

Github

Sadece yetkili olan

/ git commit / file upload

/ Data çok kullanılır.

Bitbucket

Github

→ Github'in rakipleri

5 tane ücretsiz.

Public repository herkes görebilir.

* Origin: Github'da tutulan repo. (Yazaktaki repo)
Origin'deki main branch'ı.

* Cloning: Yazaktaki dosyanın repounu kendi bilgisayarına indir.

* Pull \Rightarrow fetch+merge (Depository al birim repoma working directory'e ugula)

! git log --oneline \Rightarrow Commitlerin gösterisi (Bastı haliyle direkt git log dersen ayrıntılı sonuc gösterir.)

Dosya oluşturma \rightarrow 1. commit

İçine yazı yazma \rightarrow 2. commit

* git push

* Clone \rightarrow ilk basta yazaktaki repounu kendi bilgisayarına indirir. git init'e gerek kalmadı, bu yapılmıştır.

! Değişikliği commit etmeden önce \Rightarrow stage etmemiz gerektir.

! Tek bir komutta hem commit yap hem mesaj yarat \Rightarrow git commit -m "Mesaj"

! Working directory - Staging area arasındaki farkı kim gösterebilir
git status

! git repository'ının lokaldeki ismi \Rightarrow Master

! Uzaktaki makinede bir repository'ı nasıl sekerin \Rightarrow Git Clone

! Buğun branchleri listeleyen Puan \Rightarrow git branch

! Git repository'ı nasıl başlatırız \Rightarrow git init

! Master branch'deyken feature branchdeki değişiklikler nasıl alırız?

git merge feature

! Merge conflict neden kaynaklanır?

Aynı satırdaiki iki metnin birbiryle çetinmesi sonucu olur.

31.07.2021 dersinin son 40 dk's,

yok!

LAST LESSON

(*) git config --list \Rightarrow Kimsin kullanıcı adı, hangi filtreler gösterir.

(*) git config --global $\xrightarrow[\text{var.}]{\text{user.name}}$ Dijital kimliğin Bit Hub hesabı

Bunların birbirinden etkileşimi için \dots

Gelişini anlamadan önce çift hesap oluşturmak kullanıcıya...

git config user.name \Rightarrow Tek kullanıcıının kullanıldığı.

ALIAS

git config --global alias.co checkout

git checkout

git config --global alias.br branch

\rightarrow Bunların kisa kısaltmalarının oluşturduğu ve

git config --global alias.ci commit

bu kisma ise tediye edilmek istenilen kısaltmayı yazın.

git config --global alias.st status

alias'in amacı

toplu

(*) Repository'ı silmek için:

rm -rf .git

(*) git add dedik staging area ye attik. Sonra geri almak istedik diyelem ki?

git rm --cached dayadı.txt yaparsak geri working directory'e döner yani staging area'dan geri alır.

(*) git checkout \Rightarrow Commitler arasında geçiş yap. Branch'ler arasında geçiş yap.

(*) Show the branches \Rightarrow git branch

git branch -r (remote'da
branchlerini
gösterir)

git branch -a (Hen local
hen remote
olur branch
ları gösterir)

(*) git checkout -b new-branch \Rightarrow Hen bir branch
yaptık hen de
ona geçiş yaptık.

~ Alt.gr + ö
Filde git config --global -l

(*) Delete a branch:

git branch -d <new-branch-name> (Merge edilecek bir durum yoksa)
bir durum yoksa
bi

git branch -D <new-branch-name> (Zorla sil.)

(*) Delete a remote Branch:

git push origin --delete

(*) Rename Branch:

git branch -m

(*) Rebasing Branches (Cok kullanılmaz.)

Branch'i aldım Master'in Sırasına ekledim. Master
artık son branch'in Sırasına kaydi. (Bir branch'i main
branch'ine taşımak.)

Kodu bitti çok güzel yazmış. Onu haps direk
master yap gibi.

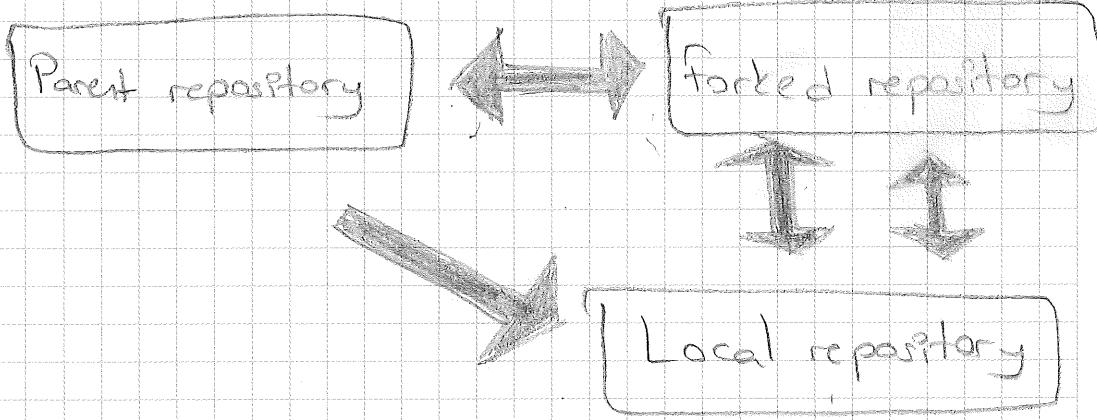
{ git fetch + git merge \Rightarrow git pull }

(*) İki commit arasındaki fark:

git diff

FORKING (Birinin projesine katkıda bulunmak)

(*) Dijitalin başka yerinden birinin çalışmalarında depistiklik yapıyorsun. Ona geni gönderme. O da bakiyor, beğenirse alır.

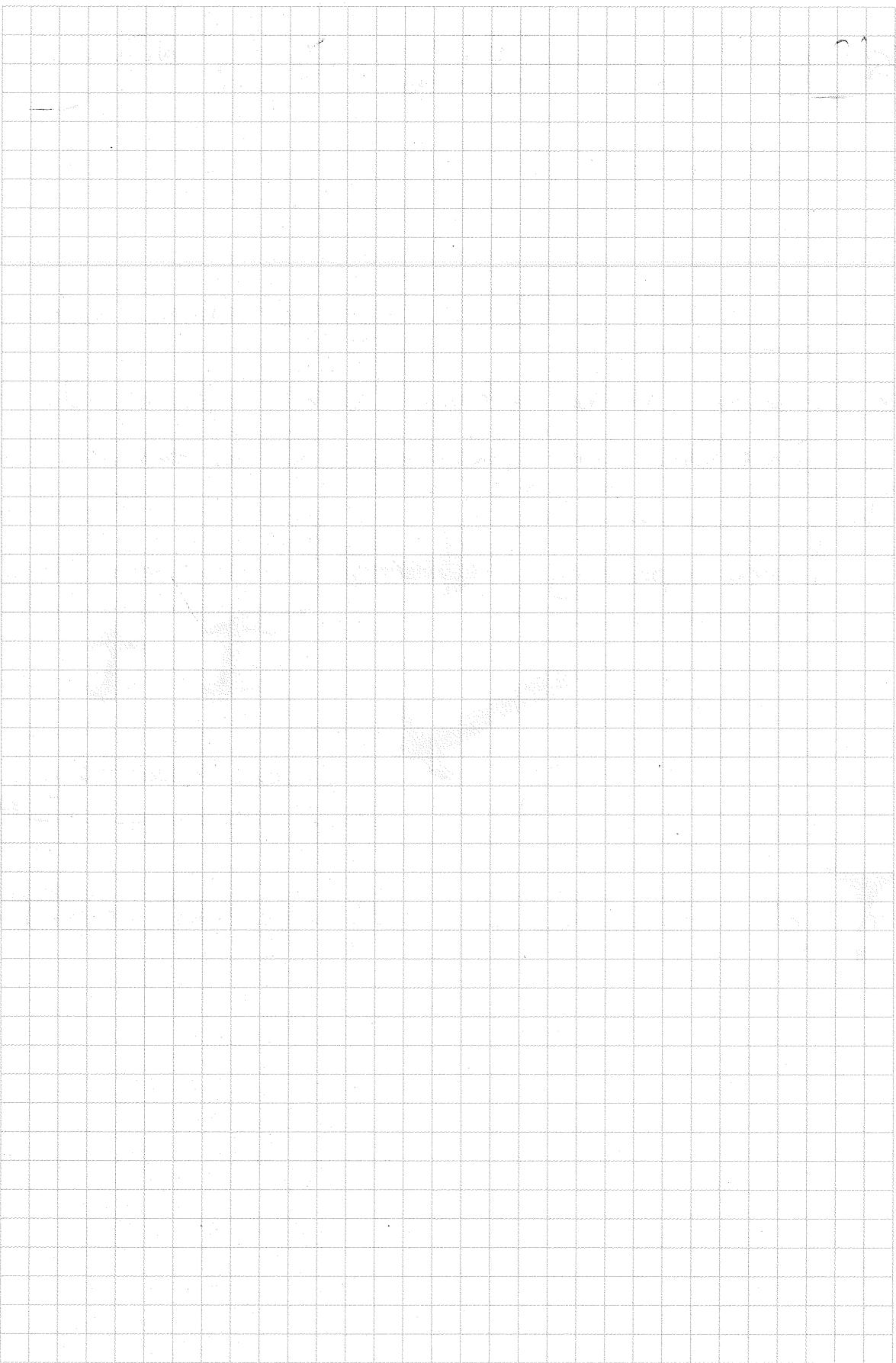
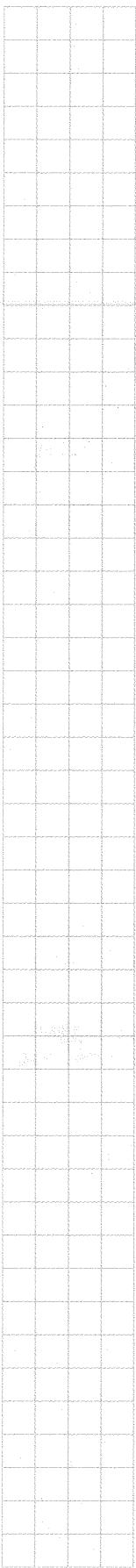


git clone $\dots \Rightarrow$ Uzaktaki repository'ye

9^ot - 3

100

100



tonguç

long