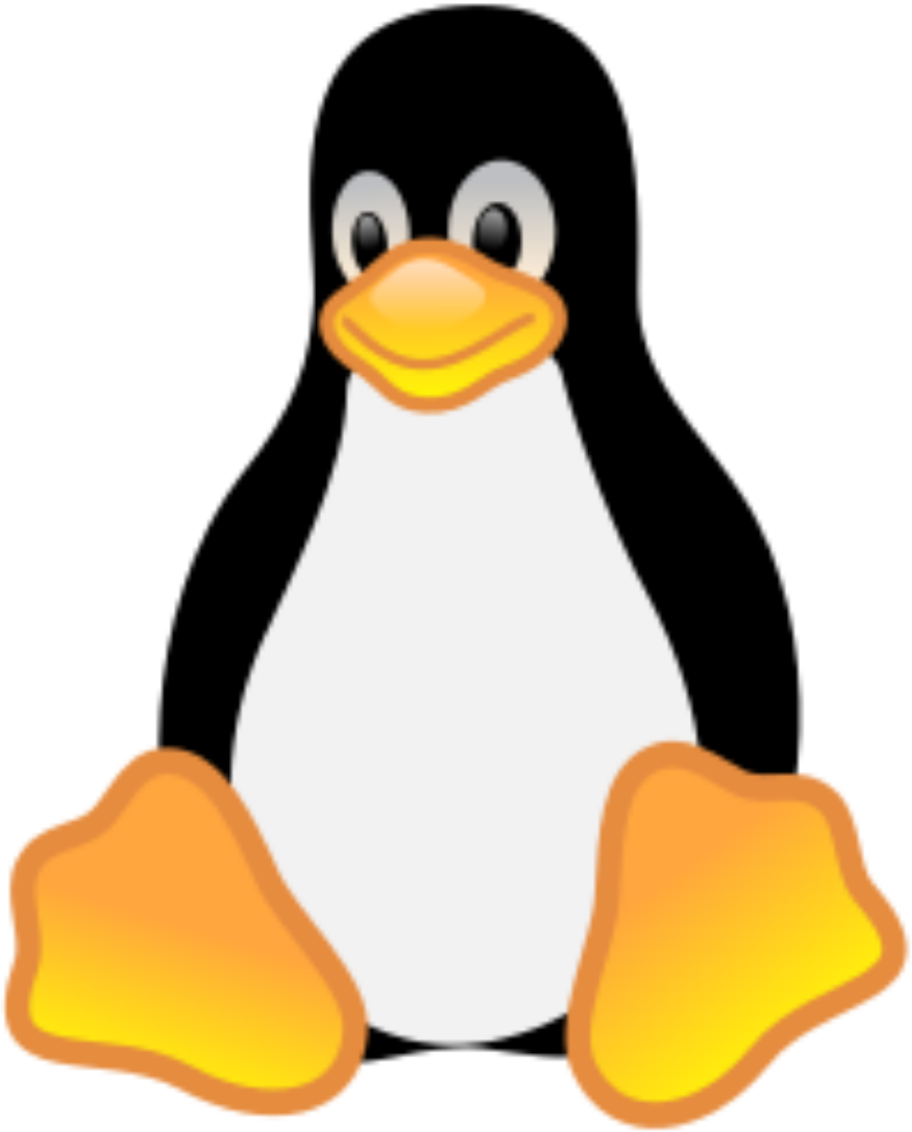
Kendi Linux'unu Yap

X Pencere Sistemi



Yazar

- Bayram KARAHAN
- Celalettin AKARSU

Paket Derleme

- Bayram KARAHAN
- Celalettin AKARSU

İletişim

- https://github.com/kendilinuxunuyap
- https://kendilinuxunuyap.github.io/x11
- kendilinuxunuyap@gmail.com

ISBN: 978-625-00-5873-2

Yayın Yılı: Nisan 2026

Ön Söz

Bu kitap, açık kaynak ve özgür yazılım dünyasına ilgi duyan herkes için hazırlanmıştır. Amacı, Türkçe kaynak eksikliğini gidermek ve kendi Linux dağıtımını oluşturmak isteyenlere yol göstermektir.

Bu serinin ilk kitabında, **Temel Linux Sistemi**'nin nasıl derlenip kullanılabilir hale getirileceği anlatılmıştı. Bu kitabımızda ise, oluşturduğumuz **Temel Linux Sistemi** üzerinde **X Pencere Sisteminin** nasıl derlenip çalıştırılacağı adım adım açıklanacaktır.

Bu kitap, **X Pencere Sistemi** oluşturan bileşenleri **X Sunucusu(Xorg)**, **X İstemcileri(xterm, xcalc, xeyes)**, **Pencere Yöneticisi(openbox)** kaynaktan derlemek isteyen linux kullanıcıları için kapsamlı bir rehber sunmayı amaçlamaktadır.

Xorg, linux sistemlerinde grafiksel kullanıcı arayüzlerinin en önemli bileşendir. Bu rehber, hem yeni başlayanlar hem de deneyimli sistem yöneticileri için hazırlanmıştır.

Xorg'u kaynaktan derlemek, Xorg'un nasıl çalıştığını anlamak için öğrenme fırsatı sunar.

İçindekiler

Temel Kavramlar

X Pencere Sistemi

X Pencere Sistemi (X), GNU/Linux ve Unix sistemlerde kullanılan grafik arayüz altyapısıdır. Bir uygulama değil, kurallar bütünüdür.

Pencerelerin nasıl çizileceğini ve klavye/fare gibi donanımlarla nasıl iletişim kurulacağını tanımlar. X sunucusu (Xorg) ile X istemcileri (xcalc, xeyes vb.) arasındaki iletişimi sağlar.

X Pencere Sistemi Bileşenleri

1- X Sunucusu (Xorg)

X Pencere Sistemi'nin en önemli bileşeni X sunucusudur (**Xorg**). X server, X sunucusu veya X11 server olarak da adlandırılır.

Ekran, klavye ve fare ile uygulamalar arasında aracı görevi görür. Pencereleri oluşturur ve grafik çıktıyı sağlar.

Xorg, bir **görüntü sunucusudur (display server)**. Masaüstü ortamlarını çalıştırır ancak kendisi bir masaüstü ortamı değildir.

2- X İstemcileri (Uygulamalar)

X sunucusuna bağlanan programlardır(xterm, xcalc, xeyes).

3- Pencere Yöneticisi (Window Manager)

Xorg yalnızca pencereyi oluşturur. Pencereyi taşıma, kapatma, büyütme ve küçültme işlemlerini yapar. Basit ortam sunan pencere yöneticileri(Openbox, i3).

Xorg + Openbox => Grafik ortamda çalışan bir sistem oluşturur.

4- Giriş Ekranı Yöneticisi (Display Manager)

Giriş ekranı yöneticisi, sistem açıldığında kullanıcı seçimi ve parola ekranı sunar. Girişten sonra kullanıcı oturumunu (masaüstü yöneticisini) başlatır.

Tercih edilen giriş ekranı yöneticileri: gdm, lightdm, lxdm. X Pencere Sistemi için zorunlu değildir.

5- Masaüstü Yöneticisi (Desktop Manager)

Masaüstü ortamını sağlayan uygulamalar bütünüdür. Dosya yöneticisi, pencere yöneticisi, ayarlar ve panel gibi bileşenleri içerir.

Tercih edilen masaüstü yöneticisi(Cinnamon, LXDE, XFCE, KDE, GNOME). X Pencere Sistemi için zorunlu değildir.

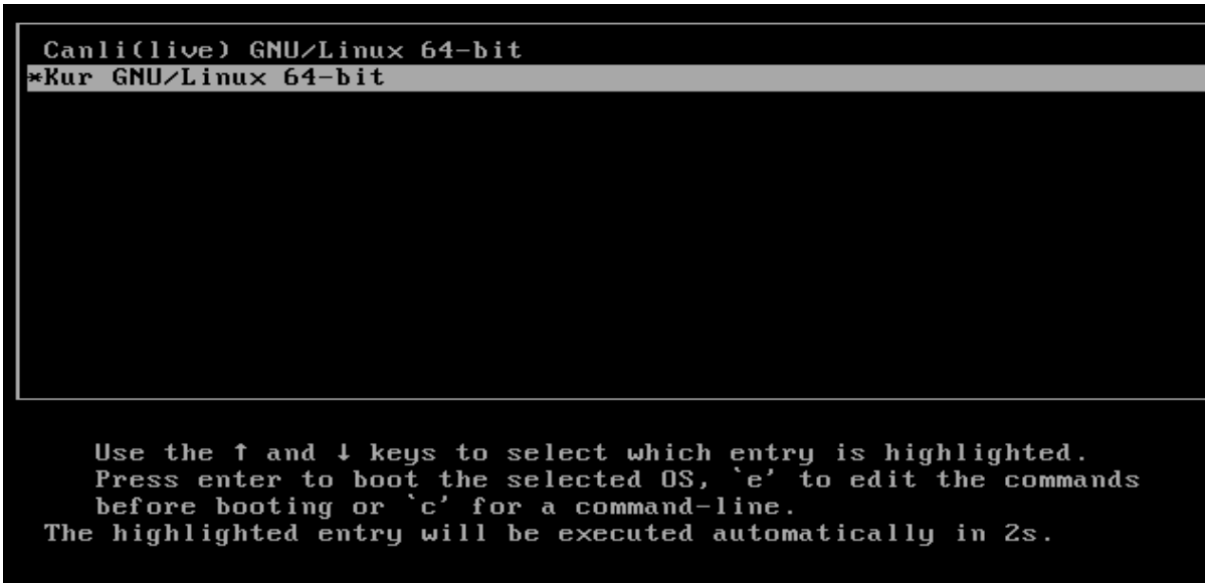
Bu dokümanda, **Xorg + Openbox** ortamında xcalc, xeyes ve xterm çalıştırılabilen bir sistemin kurulumu anlatılacaktır.

Temel Sistemin Hazırlanması

Bir önceki kitabımızda **Temel Linux Sistemini** oluşturmayı ve iso halinde kurulumu anlatmıştık. Şimdi ise bu temel sistemi kurarak bu sistem üzerine **xorg** derleyerek **x11** pencere sisteminin nasıl çalışacağı analatılacaktır. **Temel Linux Sistem** https://github.com/kendilinuxunuyap/kly-base-distro/releases/download/current/kly-base-distro.iso adresinde bulunmaktadır. İso indirip kurulum yapabilirsiniz.

Temel Sistem Kurulumu

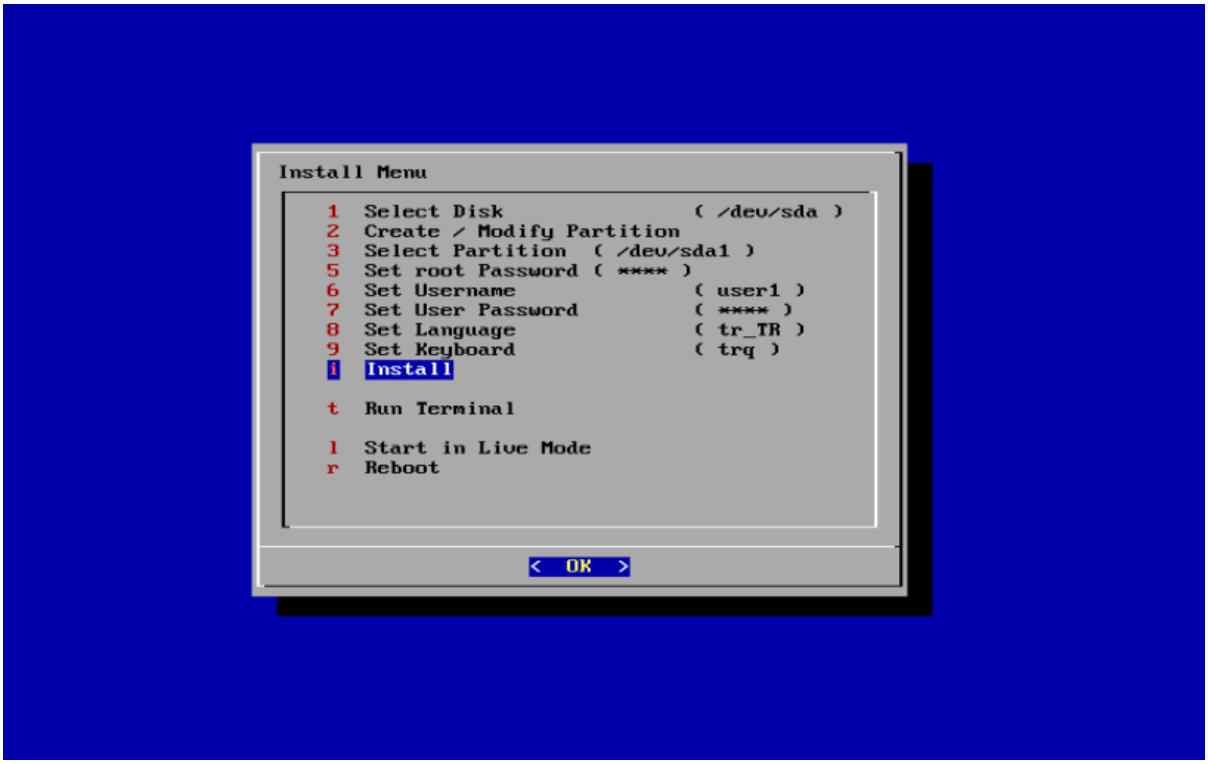
Sistemin kurulumu için resimlerde görünen sıraya göre seçimler yapmalıyız.



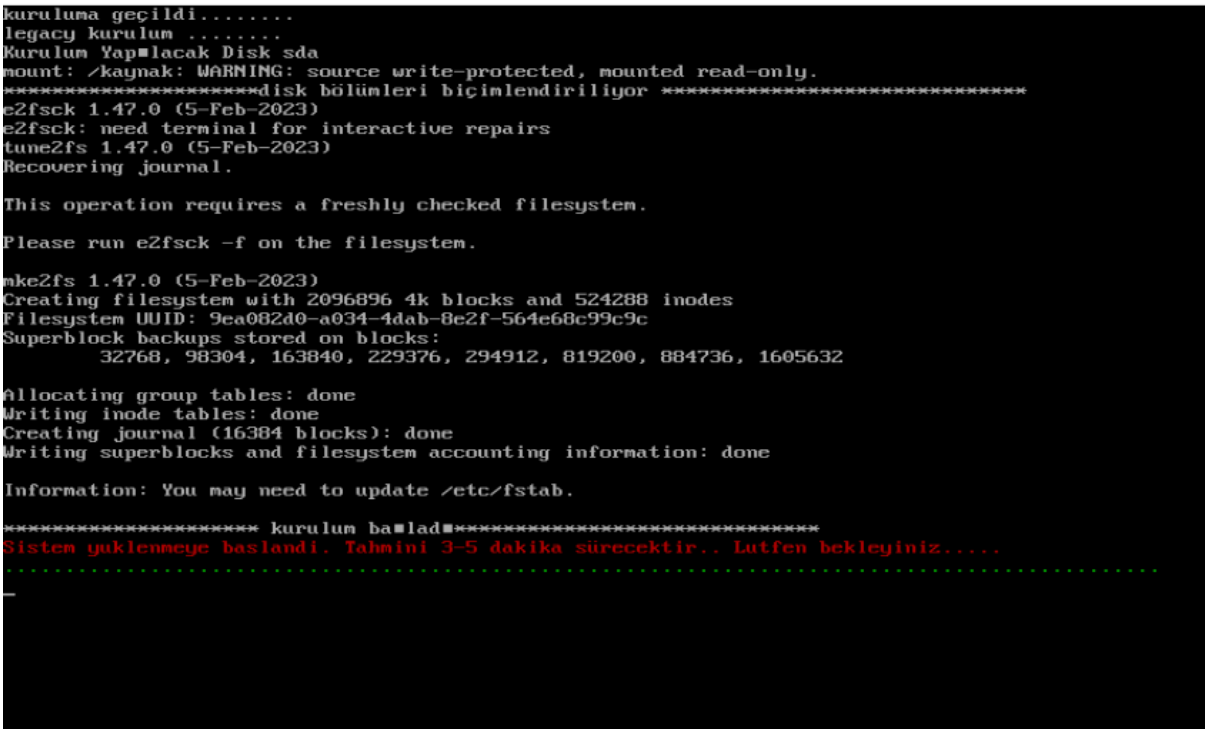
Kurulum menüsünde kullanıcı adları ve parolaları, klayve varsayılan olarak;

- Kullanıcı: root Parola: 1
- Kullanıcı: user1 Parola: 1
- Dil : tr_TR
- Klavye : trq

menüden değişiklik yapabilirsiniz. Değişiklik yapmadan sadece kurulum diskini ve disk bölümünü seçip Install(Yükle) işlemi yapabilirsiniz.

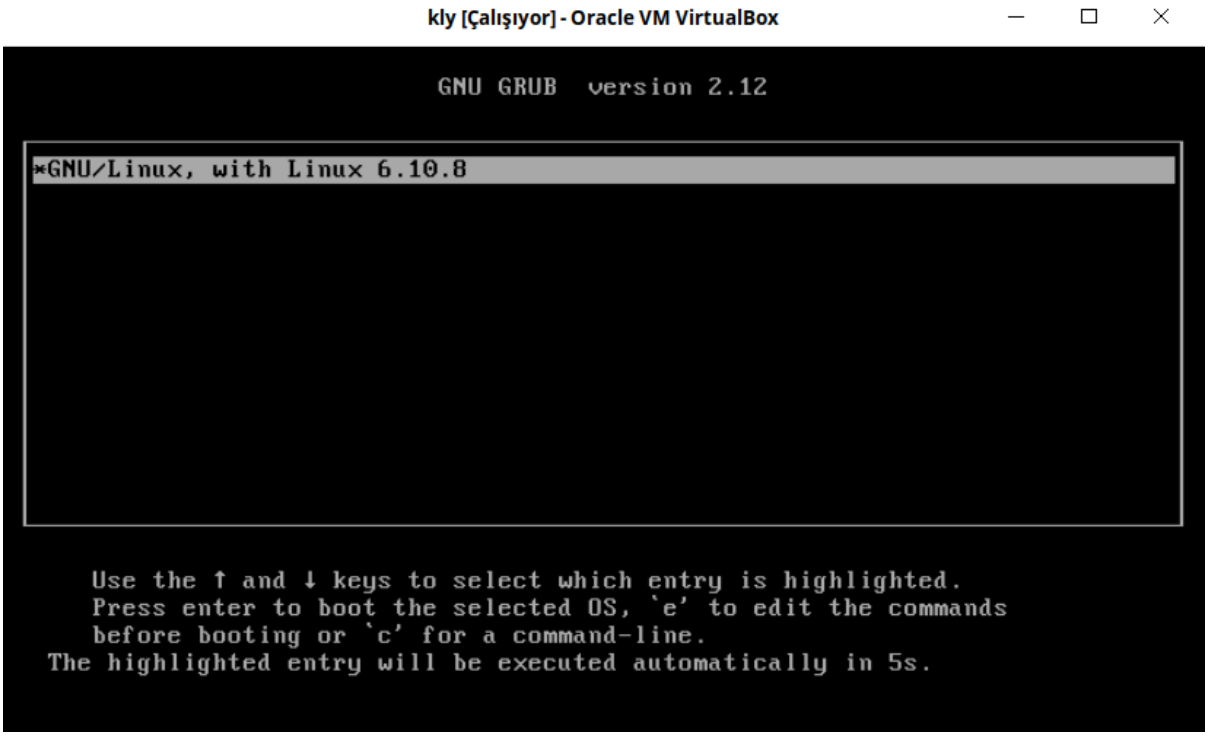


kly [Çalışıyor] - Oracle VM VirtualBox

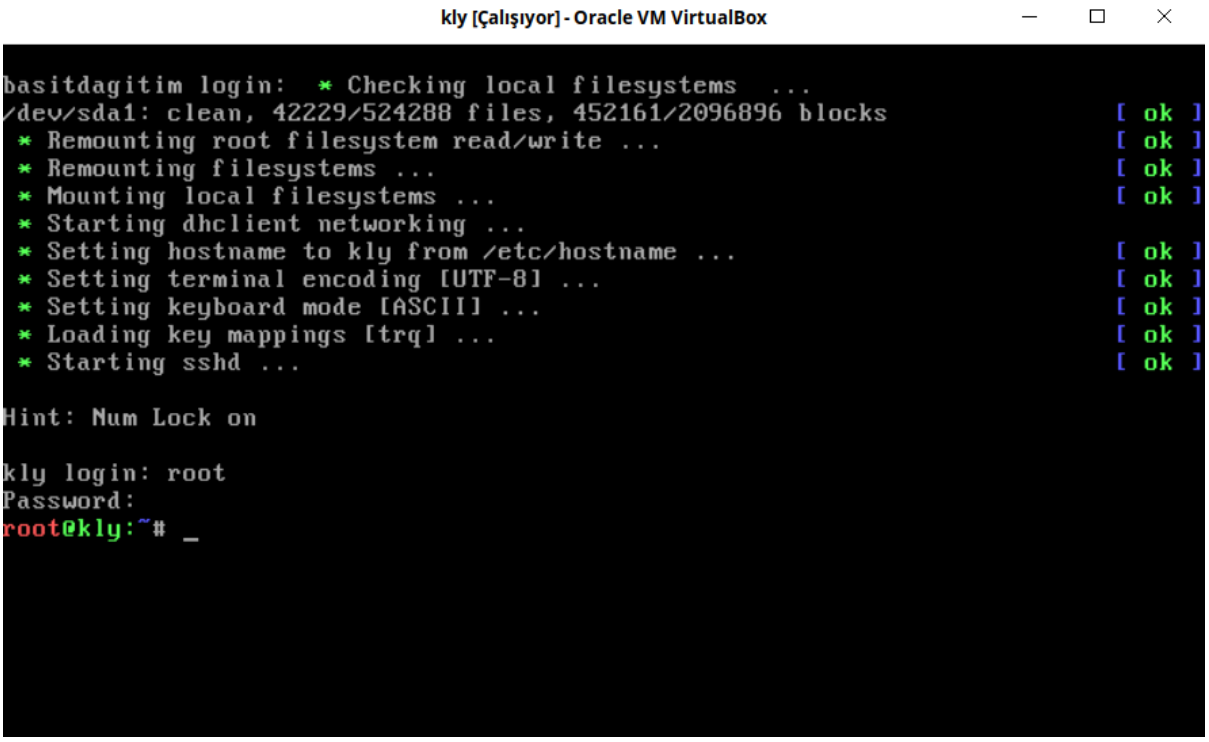


Sistemin Çalışması

Sistem kurulumu gerçekleştiğinde sistem resimde görüldüğü gibi açılmalıdır.



Sisteme **root** kullanıcısı olarak giriş yapıldığı görülmektedir.

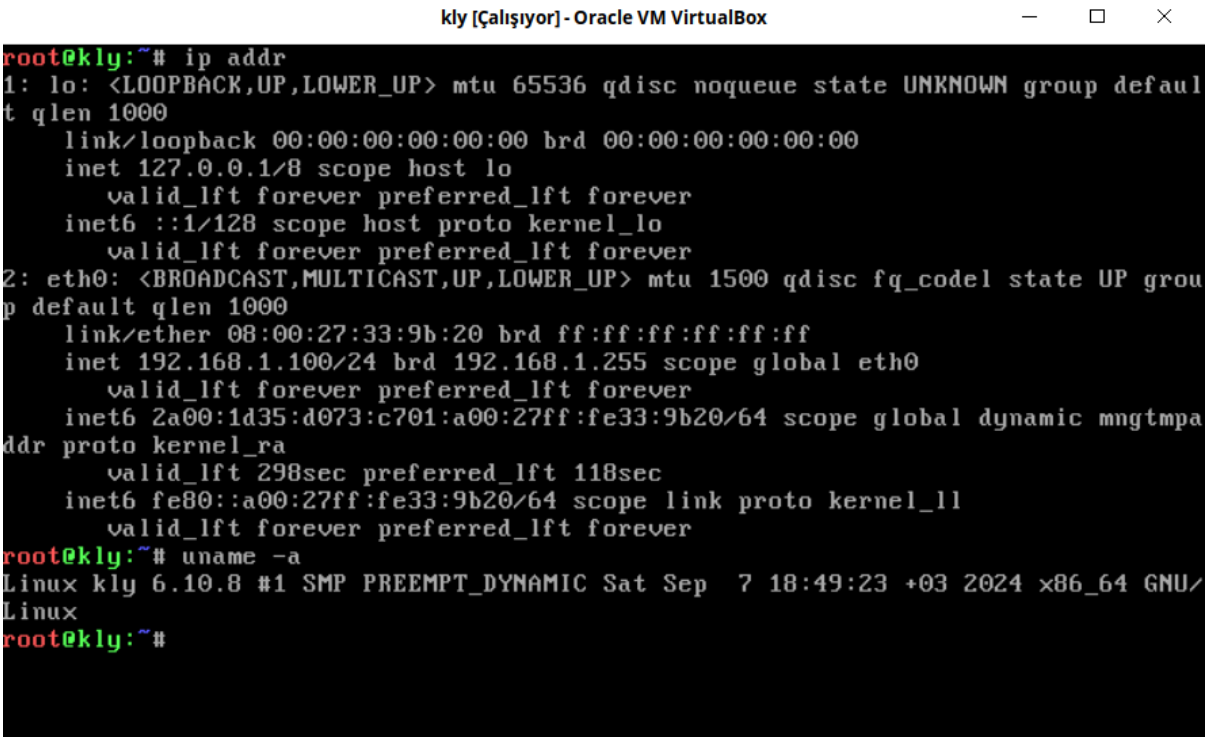


Temel Linux Sistemi görüldüğü üzere çalışmaktadır. Artık **x Pencere Sistemi** paketlerini ve bağımlılıklarını **kly Paket Sistemini** kullanarak derlenecek ve **Temel Linux Sistemi** üzerine kurularak **x Pencere Sistemi** ortamımızı çalışır hale getireceğiz.

Temel Sisteme Internet Bağlantısı

Temel Sistem içerisinde bulunan **iproute2, dhclient, net-tools** paketleri derlendi ve açılışta **dhclient** aktif hale gelmektedir. Bu paket ile ağa dahil olunmaktadır.

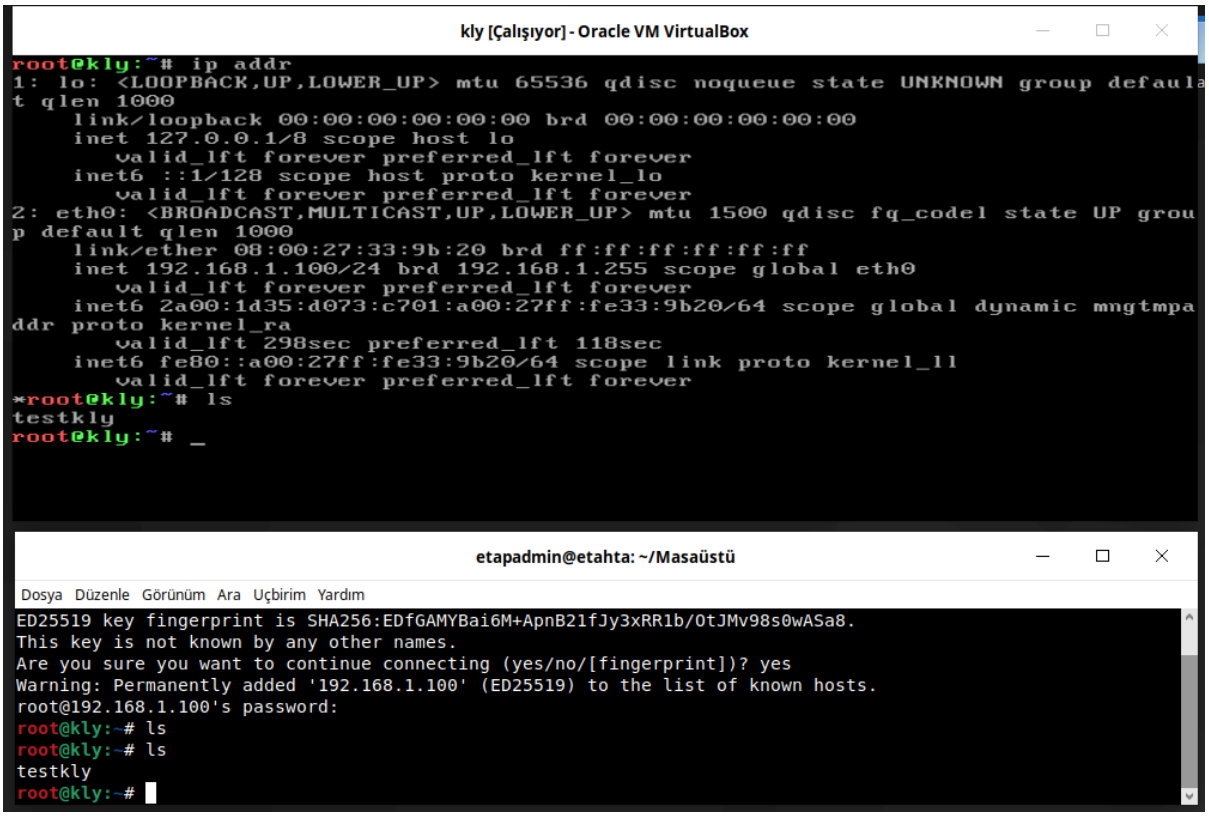
Aşağıda **kly Temel Sistem** ile atanan ip adresini görmekteyiz. Eğer ip adresi almamışsa terminalde **dhclient** komutunu çalıştırınız.



Temel Sisteme openssh ile Bağlanma

ssh terminal üzerinden uzak bilgisayarlara erişim yapan bir uygulamadır. **Temel Sistem** içerisinde ssh paketi derlendi ve açılışta aktif hale getirildi. ssh kullanımı **Yardımcı Konular** bölümünde anlatılmıştır.

Aşağıda **kly Temel Sisteme ssh** ile erişimin nasıl yapıldığı görülmektedir.

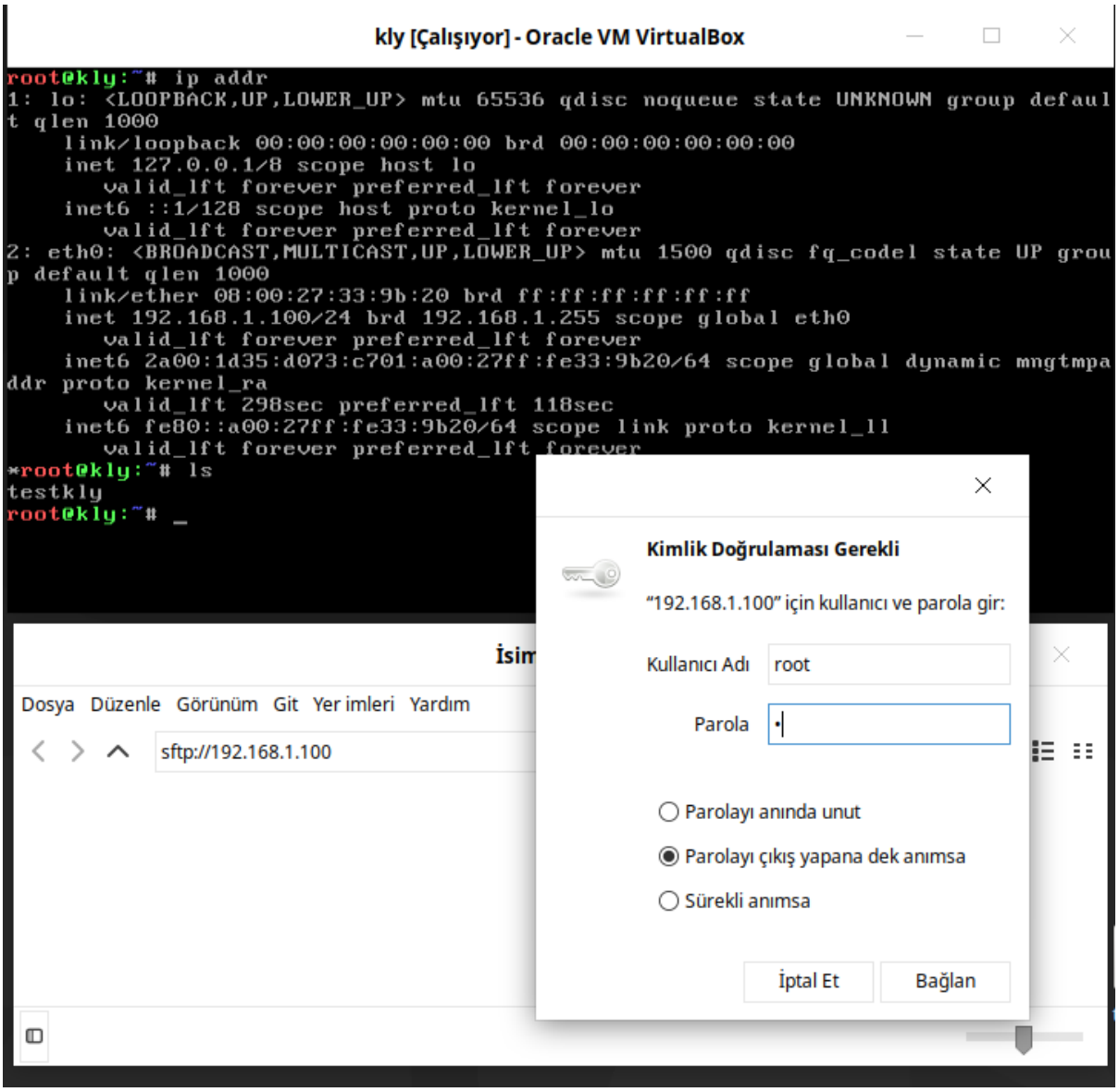


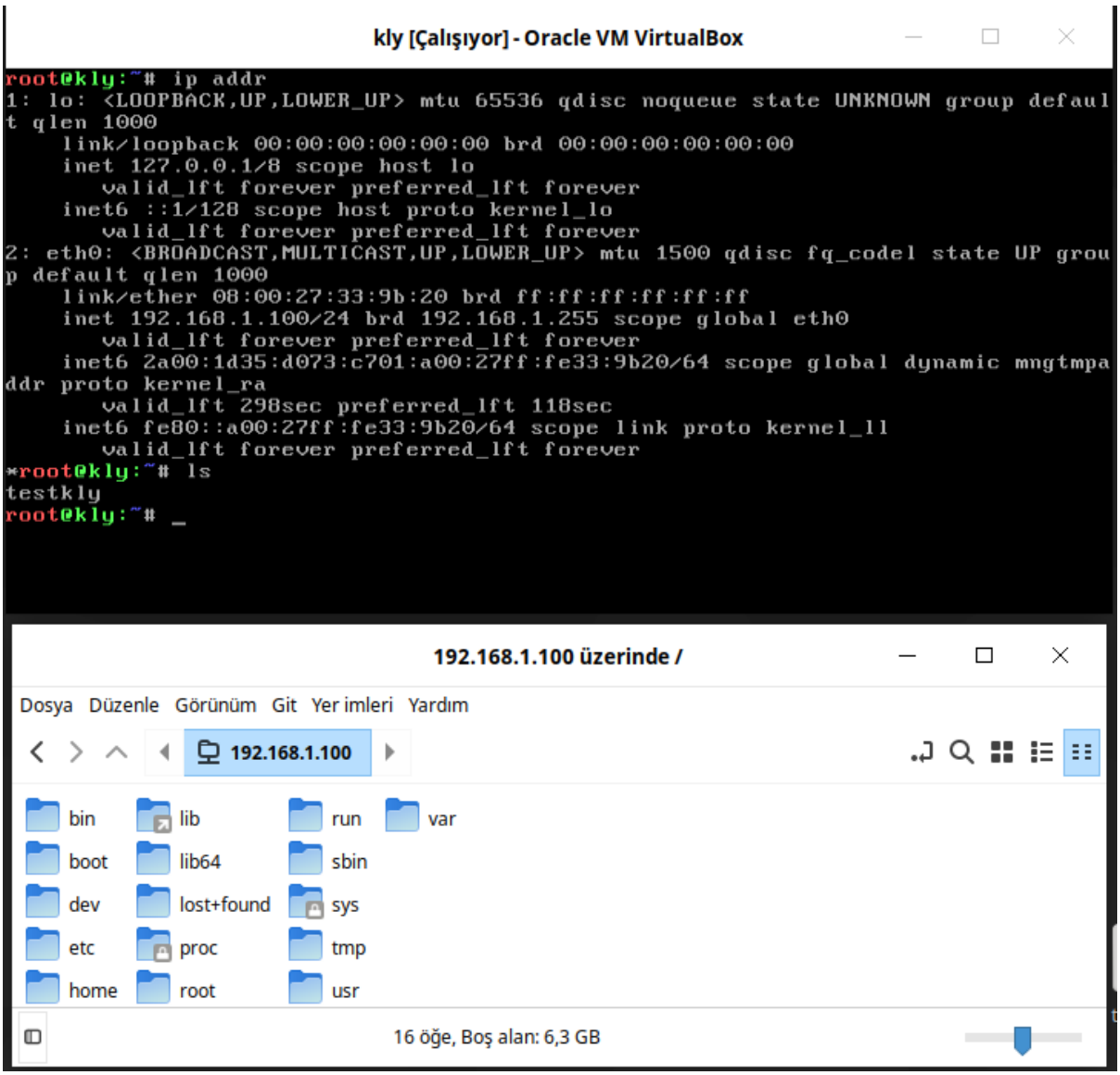
Bu doküman boyunca paketleri **Debian** ortamında derleyeceğiz. Derlediğimiz paketleri **Temel Sistem'e** kopyalacağız. Kopyalama işlemini **ssh** paketi içinde gelen **sftp** veya **scp** ile yapacağız. Kopyalanan paketlerin **kurulması ve kaldırıması** gibi işlemleri **ssh** bağlantısı üzerinden gerçekleştireceğiz.

Temel Sisteme sftp ile Bağlanma

sftp terminal veya pencere yöneticisi üzerinden uzak bilgisayarlara erişim yapan bir uygulamadır. **Temel Sistem** içerisinde **ssh** paketinin bir parçası olarak gelmektedir. sftp kullanımı **Yardımcı Konular** bölümünde anlatılmıştır.

Aşağıda **kly Temel Sisteme sftp** ile erişimin nasıl yapıldığı görülmektedir.

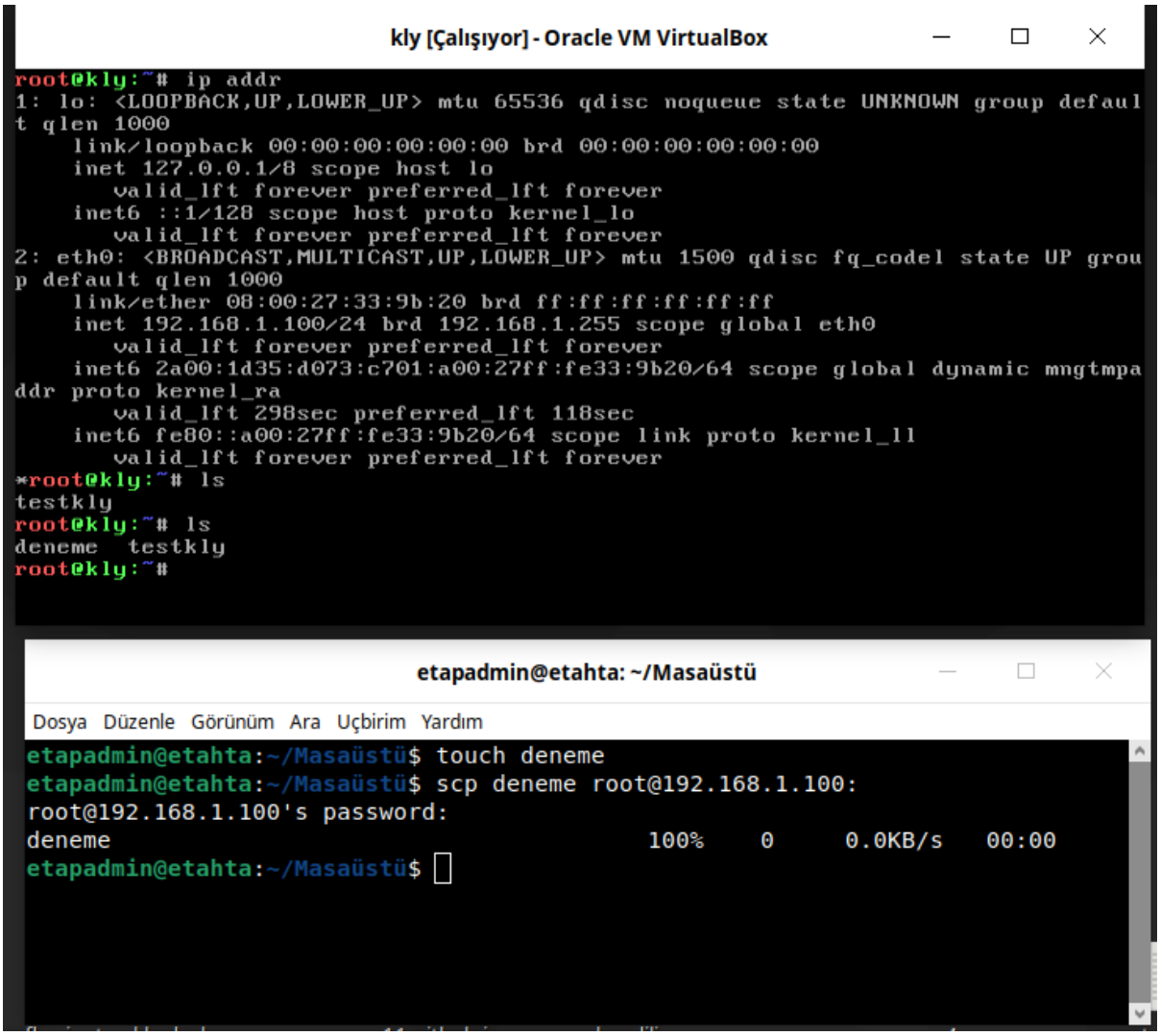



kly [Çalışıyor] - Oracle VM VirtualBox
root@kly:~# ip addr
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group defaul
t qlen 1000
link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
inet 127.0.0.1/8 scope host lo
valid_lft forever preferred_lft forever
inet6 ::1/128 scope host proto kernel_lo
valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP grou
p default qlen 1000
link/ether 08:00:27:33:9b:20 brd ff:ff:ff:ff:ff:ff
inet 192.168.1.100/24 brd 192.168.1.255 scope global eth0
valid_lft forever preferred_lft forever
inet6 2a00:1d35:d073:c701:a00:27ff:fe33:9b20/64 scope global dynamic mngtmpa
ddr proto kernel_ra
valid_lft 298sec preferred_lft 118sec
inet6 fe80::a00:27ff:fe33:9b20/64 scope link proto kernel_ll
valid_lft forever preferred_lft forever
*root@kly:~# ls
testkly
root@kly:~# _
192.168.1.100 üzerinde /
Dosya Düzenle Görünüm Git Yer imleri Yardım
192.168.1.100
bin
lib
run
var
boot
lib64
sbin
dev
lost+found
sys
etc
proc
tmp
home
root
usr
16 öğe, Boş alan: 6,3 GB

Temel Sisteme scp ile Dosya Kopyalama

scp terminal üzerinden uzak bilgisayarlara dosya kopyalaması yapan bir uygulamadır. **Temel Sistem** içerisinde **ssh** paketinin bir parçası olarak gelmektedir. scp kullanımı **Yardımcı Konular** bölümünde anlatılmıştır.

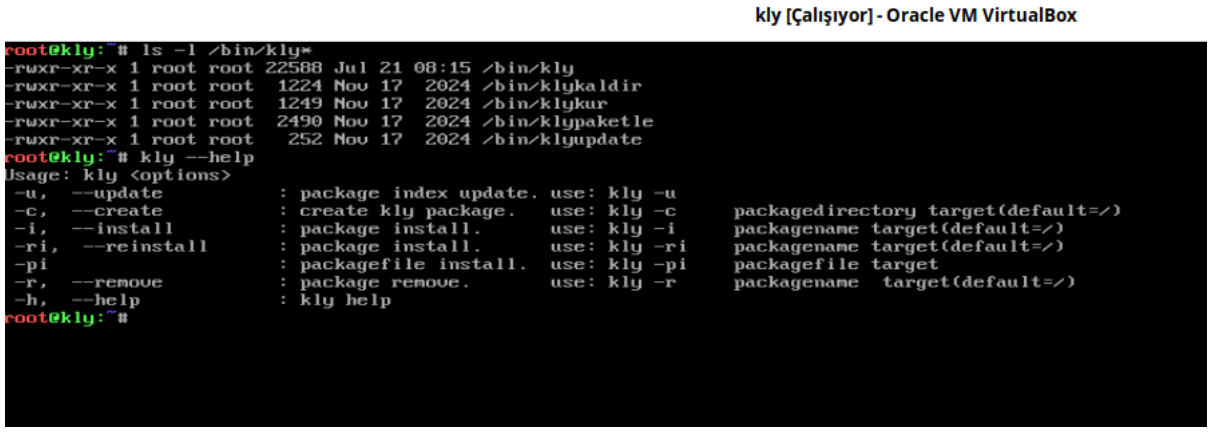
Aşağıda **kly Temel Sisteme scp** ile **deneme** dosyasının nasıl kopyalandığı görülmektedir.



kly Paket Sistemi Kullanımı

Paket sistemi, sisteme paket **(kurma, kaldırma, index güncelleme)** gibi temel işlemlerin yapılmasını sağlar. **Temel Sistem** kitabımımızda **Paket Sistemi(kly)** tasarımı anlatılmıştı. Bu kitapta, kly paket sistemi kullanılarak paketleri derleyeceğiz. Paket sistemi paketleri tekrar tekrar derlemek yerine paket olarak hazırlamakta ve istediğimiz zaman oluşturulan sisteme yüklenmekte veya kaldırılmaktadır. Bir sorun olduğu farkedildiğinde ise tekrar derlenerek sadece bu paket güncellenebilmektedir. Bu tür avantajlardan dolayı paket sistemini kullanacağız.

Burada paket sisteminin nasıl kullanılacağı anlatılacaktır. Paket sistemimiz **Temel Sistem** üzerinde **base-file** paketiyle sisteme yüklendi. Aşağıda **kly Temel Sistem** üzerinde hazır gelen paket sistemi uygulamalarımız görülmektedir.



kly: *klypaketle, klykur, klykadir, klyupdate* scriptlerimizin hepsini tek scriptte toplanmış halidir. Ayrıca paketlerin kurulumu ve kaldırılması sırasında bağımlılıklarınıda paketle birlikte kurmakta veya kaldırmakta. Bu dokümanda paketlerin derlenmesi, kurulması, kaldırılması vb. işlemler **kly** scripti kullanılarak yapılacaktır. **klypaketle, klypaketle, klykur, klykaldir, klyupdate** scripleri bir öğretici olması nedeniyle bulunmaktadır. Daha kapsamlı işlemler için yetersiz kalacaktır.

kly Paket Siteminin Debian'a Kurulumu

kly paket sistemi **Temel Sistem** üzerinde kurulu olarak gelmektedir. Üstte verilen resimde görülmektedir. Kullanımı ve parametrelerini unutmanız durumunda **kly --help** komutuyla kullanımı öğrenebilirsiniz.

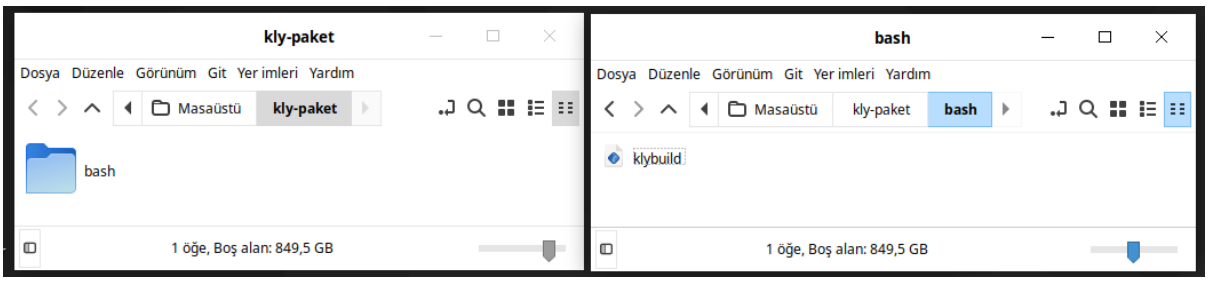
Debian üzerinde kullanmak için kly scriptimizi indirip **/bin** konumuna kopyalayalım. Aşağıda verilen komutları sırasıyla çalıştırınız. **kly** paket sisteminin sorunsuz çalışması için **busybox** paketinin kurulu olması gerekektedir. Sorunsuz çalışmışsa **Debian** sistemimize **kly** paket sistemimiz kurulmuştur.

```
wget https://kendilinuxunuyap.github.io/_static/files/base-file/kly
sudo mv kly /bin/kly
sudo chown root:root /bin/kly
sudo chmod 755 /bin/kly
sudo apt update
sudo apt install busybox-static -y
```

kly Paket Sistemiyle Paket Yapma

kly paket sistemi ile paket yapma işlemini Debian ortamında yapacağız. Debian üzerinde paket sistemimizi oluşturan scriptimiz **/bin/kly** konumunda olması gerekmektedir.

Şimdi **bash** paketinin **kly Paket Sistemi**'ni kullanarak derlemesini yapalım. Paket için aşağıda görüldüğü gibi masaüstüne **kly-paket** dizini oluşturuldu. **kly-paket** dizini içine **bash** dizini oluşturuldu. **bash** dizini içine **klybuild** dosyası oluşturuldu.



klybuild dosyasının içeriğine aşağıdaki kodu ekleyiniz.

```
#!/usr/bin/env bash
version="5.2.21"
name="bash"
depends="glibc,readline,ncurses"
description="GNU/Linux dağıtımında ön tanımlı kabuk"
source="https://ftp.gnu.org/pub/gnu/bash/${name}-${version}.tar.gz"
groups="app.shell"
setup() {
        cd $SOURCEDIR
        ./configure --prefix=/usr --libdir=/usr/lib64   --bindir=/bin \
                --with-curses --enable-readline --without-bash-malloc
}
build() {
        make
}
package()       {
        make install DESTDIR=$DESTDIR
        cd $DESTDIR/bin
        ln -s bash sh
}
```

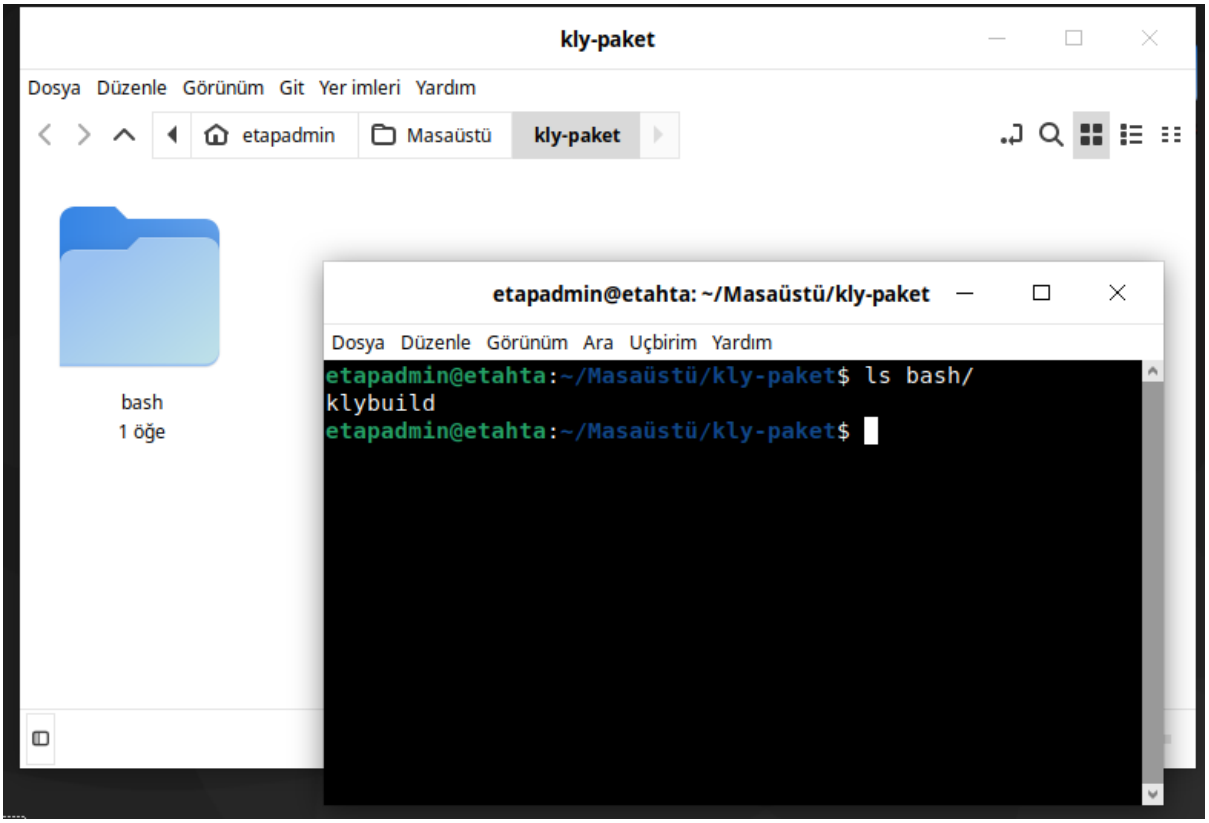
klybild dosyalarında Kullanılan Değişkenler

- **ROOTBUILDDIR:** /home/$user/distro/build → Derleme dizini
- **BUILDDIR:** /home/$user/distro/build/build-${name}-${version} → Paket derleme dizini
- **DESTDIR:** /home/$user/distro/rootfs → Yükleme dizini
- **PACKAGEDIR:** $(pwd) → Derleme scriptinin bulunduğu dizin
- **SOURCEDIR:** /home/$user/distro/build/${name}-${version} → Kaynak dizin

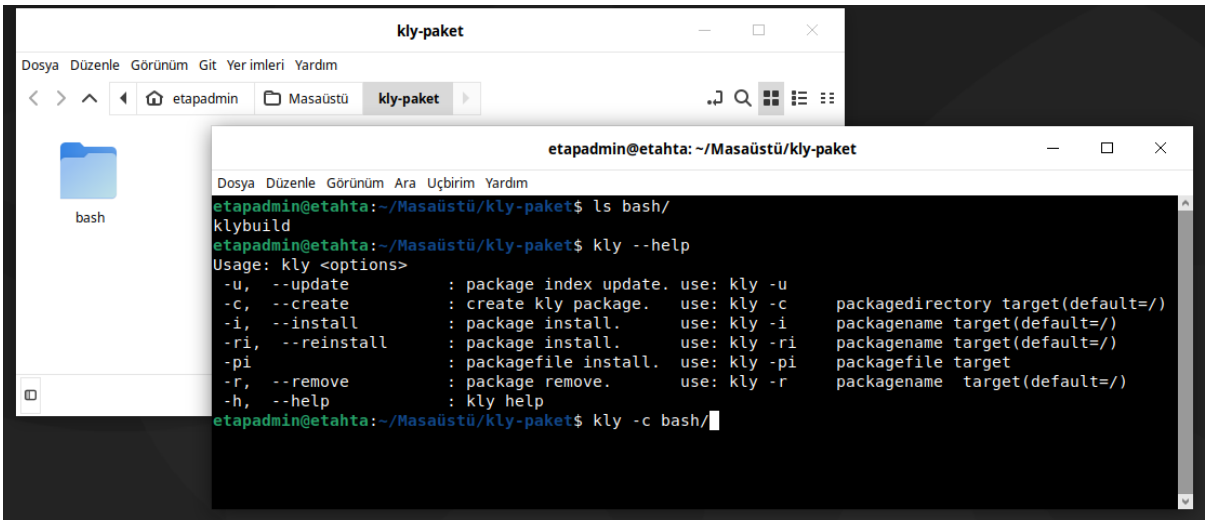
Değişkenleri derleme scripleri içinde kullanılmaktadır. Örneğin, kaynak dizinde işlem yapmak için sadece **$SOURCEDIR** kullanmanız yeterlidir. Bu yapılar tüm paketlerde geçerli olacak.

Not: Bazı paketlerin ek dosyaları olabilir. Derleme scripti altında **Ek dosya için tıklayınız** bağlantısını(link) kullanarak ek dosyaları indirin ve paketin dizini içine çıkartınız. **bash** paketinin ek dosyaları olsaydı **bash** dizini içine indiğimiz dosyayı arşivde çıkartacaktık.

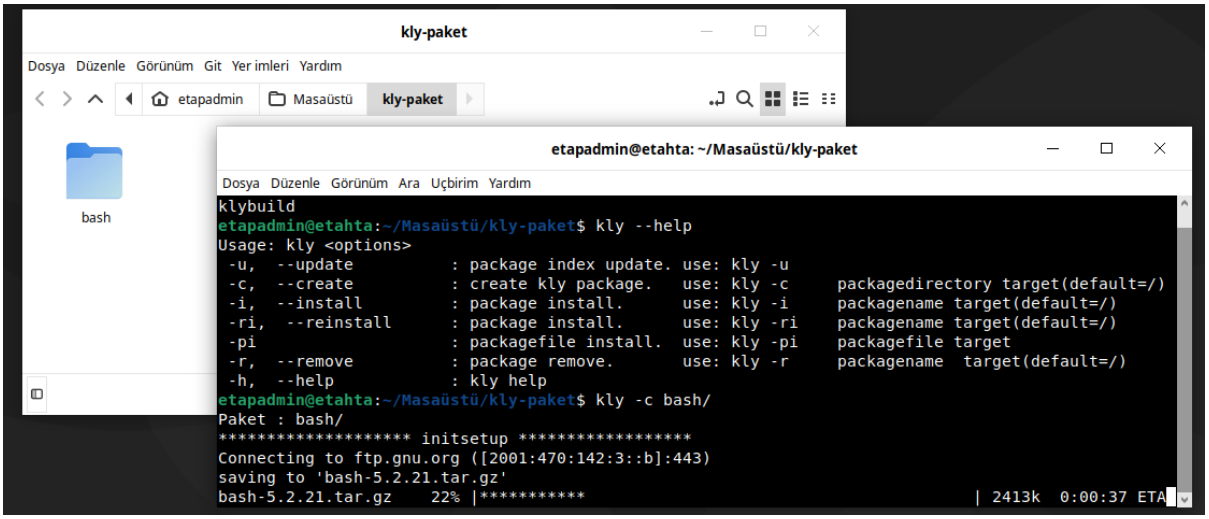
kly-paket dizini konumunda aşağıdaki gibi terminal açınız.



Açılan terminalde aşağıda görüldüğü gibi komutu çalıştırınız. komut hangi konumda çalıştırılmışsa **.kly** uzantılı pakeyimiz orada oluşacaktır. Siz istediğiniz yerde çalıştırabilirsiniz. Önemli olan paket için oluşturduğunuz **klybuild** dosyanınızın konumunu doğru vermeniz.

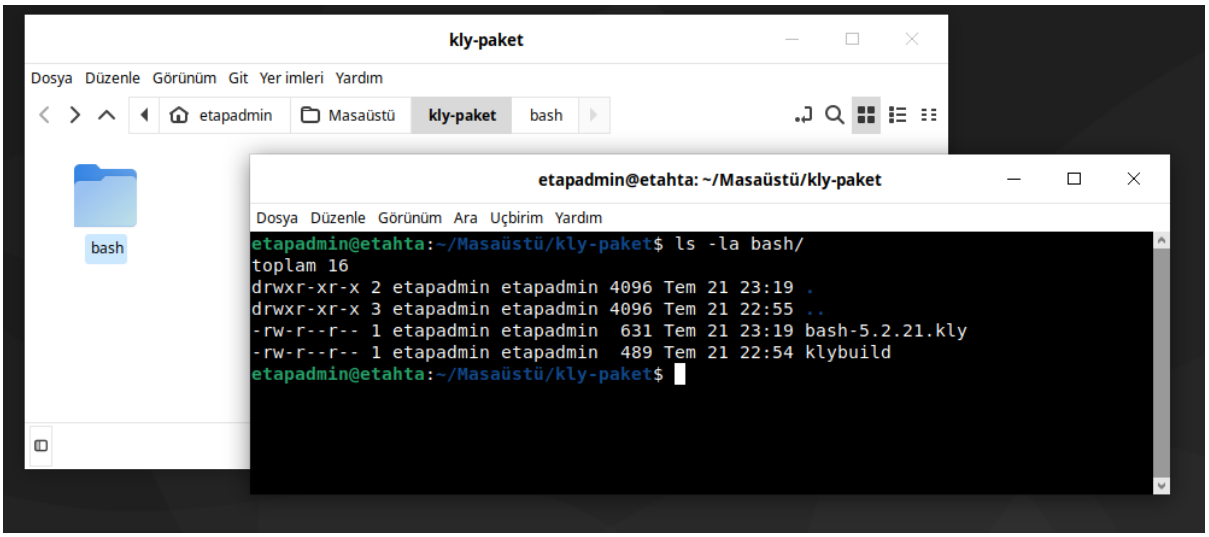


Aşağıda **bash** dizinini parametre olarak vererek **bash** paketimizin derlemesini başlatıyoruz.



Derleme işlemi paketin büyüklüğüne bağlı olarak zaman alacaktır. Paket derlemesi bittikten sonra aşağıda görüldüğü gibi terminal çıktısı almalısınız. Sorun çıkması durumunda terminalde hata mesajları alırsınız.

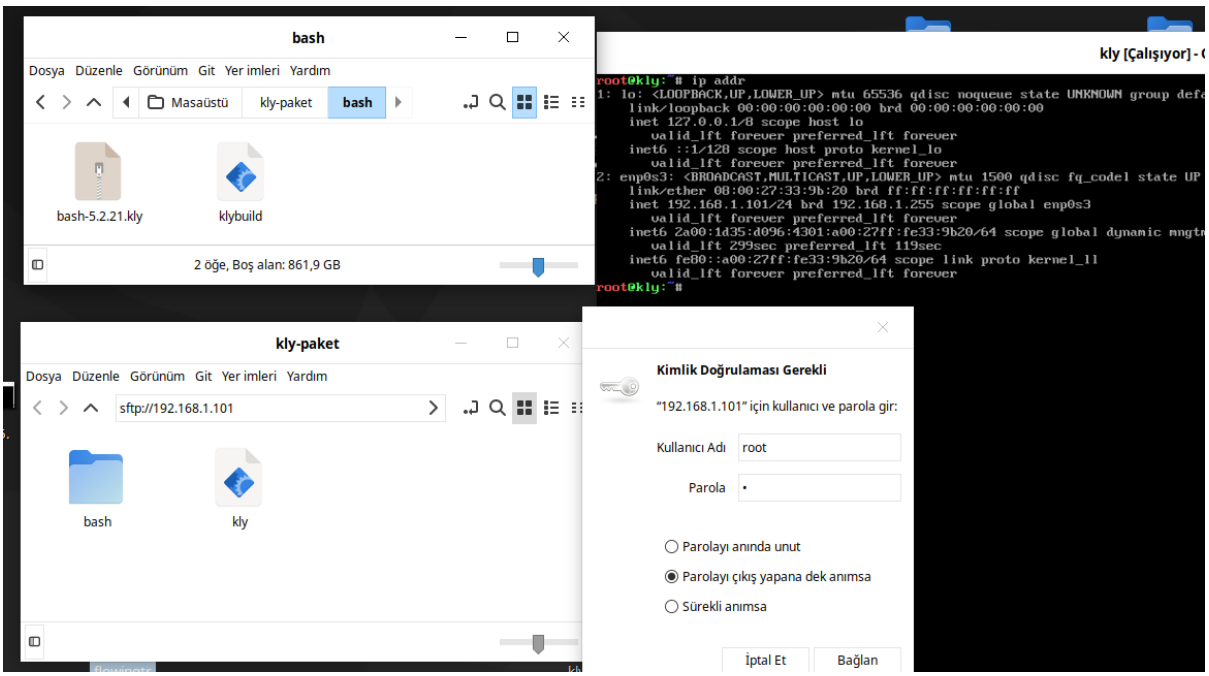
Derleme işlemi bittikten sonra **kly-paket/bash** dizini komununda aşağıda görüldüğü gibi **bash-5.2.21.kly** paketimizi oluşturacaktır.



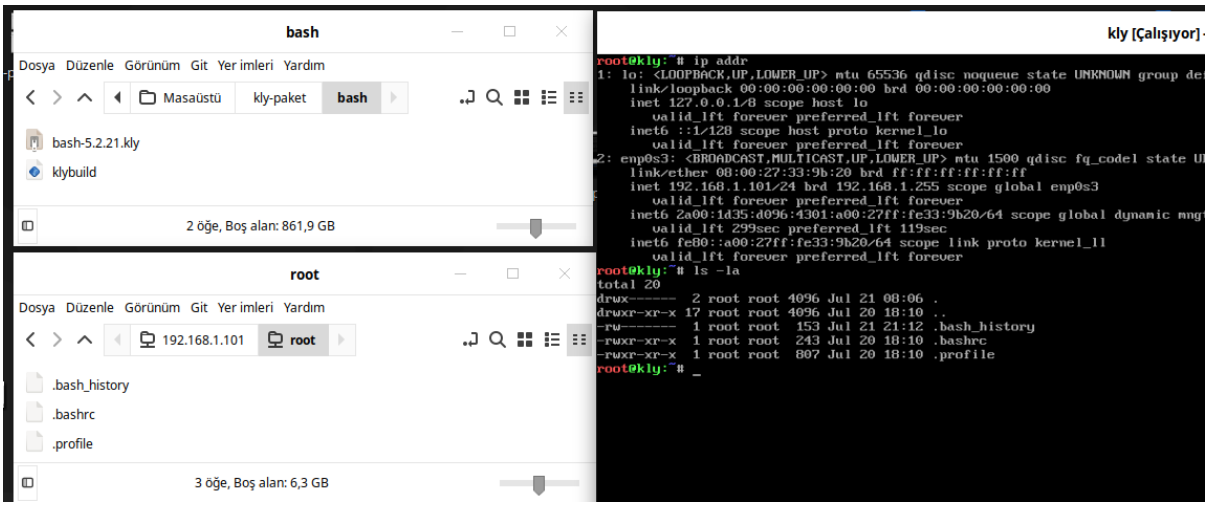
kly ile Paket Kur

kly Paket Sistemi ile paket oluşturma konusunda **bash** paketi derlenmişti. Şimdi ise bu paketi **Temel Sistem** üzerine kopyalamayı ve kurma işlemini yapalım. **bash** paketimiz masaüstümüzde **kly-paket/bash/bash-5.2.21.kly** konumunda bulunmaktadır. Bu konumdaki dosyamızı **Temel Sistem** üzerine **scp** veya **sftp** kullanarak kopyalayabiliriz. **scp** terminal üzerinden kopyalar. **sftp** terminal veya pencere yöneticisi üzerinden kopyalar. Burada tercih size kalmıştır. **sftp** ile pencere yöneticisini kullanmak daha kolay olabilir. **scp ve sftp** kullanımı **Yardımcı Konular** bölümünde detaylıca anlatılmıştır. Ayrıca **Temel Sistemin Hazırlanması** bölümünde de **scp** ve **sftp Temel Sistem** ile nasıl kullanılacağı anlatıldı.

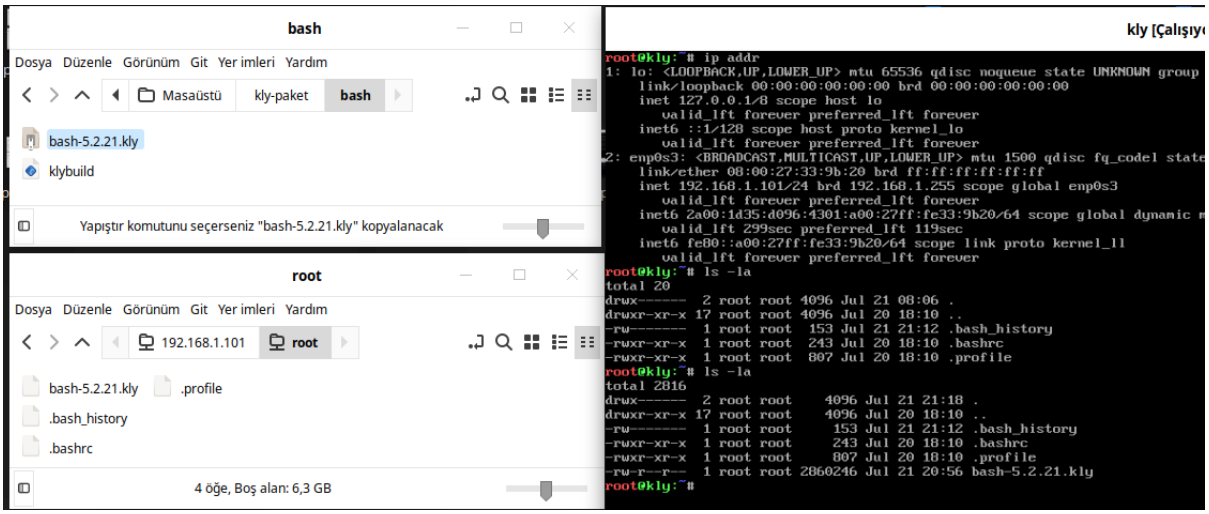
Aşağıda **sftp** ile nasıl bağlantı yapılacağı görülmektedir.



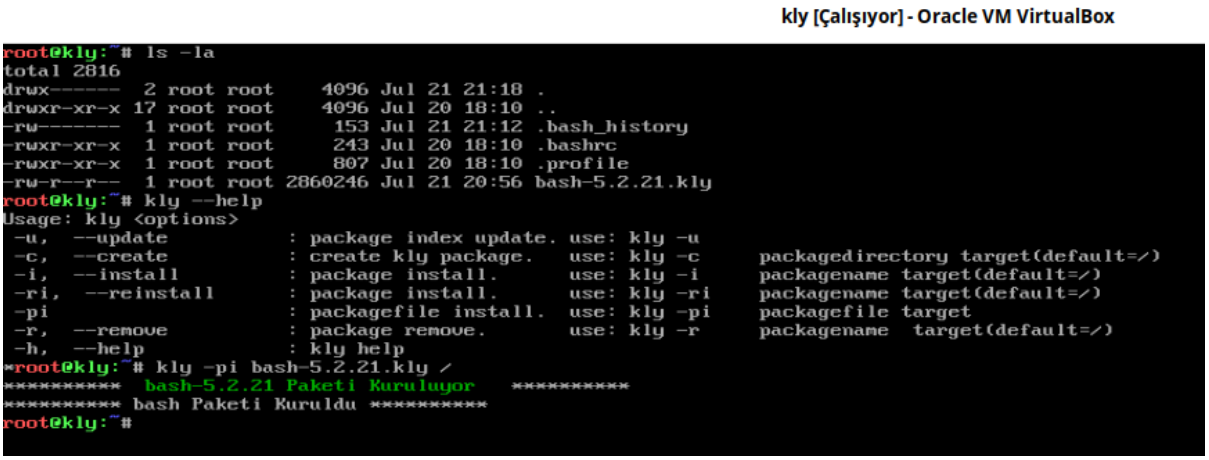
Aşağıda **sftp** ile **bash-5.2.21.kly** paketini kopyalanma öncesi **Temel Sistemde** olup olmadığını **ls -la** komutuyla kontrol ediyoruz.



Kopyalama işleminden sonra **bash-5.2.21.kly** paketinin **Temel Sistemde** olup olmadığını **ls -la** komutuyla görüyoruz.



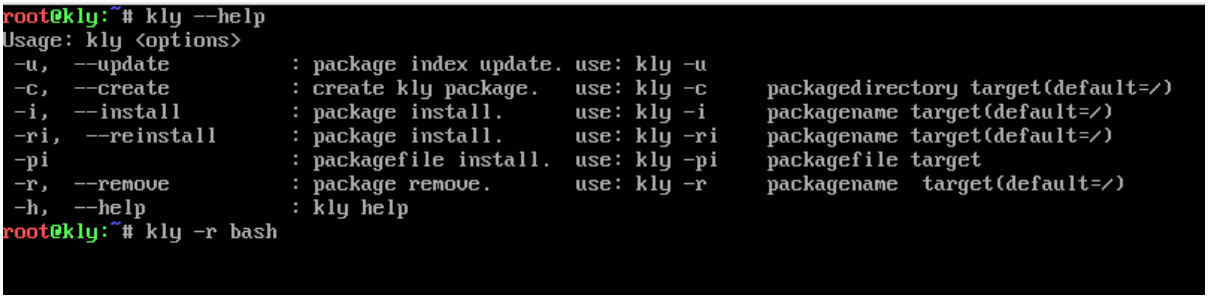
kly paket sistemini kullanarak yerelde bulunan paketin kurulumunu **kly -pi bash-5.2.21.kly /** komutuyla yapıyoruz.



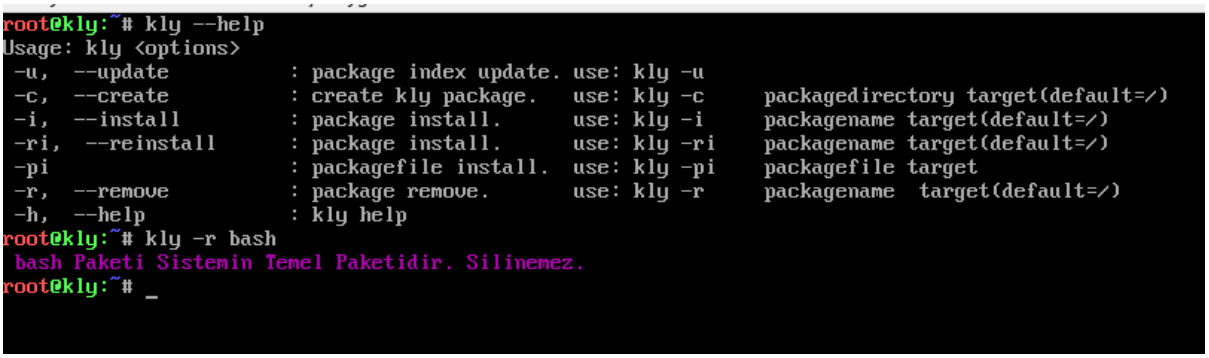
kly ile Paket Kaldir

kly Paket Sistemiyle sistemde yüklü olan bir paketi **kly -r paketadı** şeklide komut kullanılarak kaldırılabilir. Daha önce paket oluşturma ve paket kurulum işlemlerinde **bash** paketini kullanmıştık. Şimdi **bash** paketini kaldırmamız durumunda sistemde terminali kulanamaz duruma getirecektir. Bazı temel paketleri kaldırmak sistemimizin bozulmasına sebep olabilir. Paket kaldırırken dikkatli olmak gerekir.

Aşağıda **bash** paketinin nasıl kaldırılacağı görülmektedir.



Bazı paketleri **(glibc, ncurses, readline, bash)** tasarladığımız **kly** paket sisteminde kaldırılmasını engelledik. Bu paketler kalkması durumunda sistem kullanılamaz hale gelir. Eğer değişiklik gerekiyorsa sadece yeniden kurulum yapılabilir. Aşağıda paketin kaldırılamadığı mesajı görülmektedir.



glibc, ncurses, readline, bash dışında başka paketler olsaydı paket kaldırılması gerçekleşecekti.

kly Paketlerini Githuba Yükleme ve İndexleme

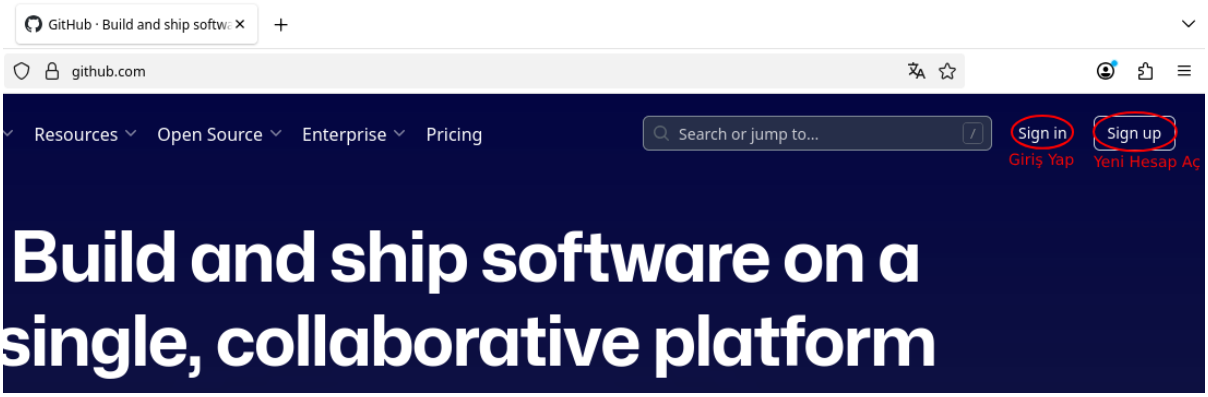
Depo, paketlerimizin olduğu alandır. Depoda ne kadar paket varsa bunların isimleri sürüm numaraları gibi bilgiler ile adreslerini liste halinde oluşturma işlemine **depo indexleme** denir. Depo indexlenirken genellikle bilgiler **paket derleme talimatı(klybuild)** dosyasından alınır. Paketlerin listesi, paketler kurulurken, silinirken ve güncellenirken kullanılmaktadır.

kly github Depo Yapma

Bu doküman kullanılarak hazırlanan paketleri bilgisayarınızda bir dizinde tutabiliriz. Fakat bu çok kısıtlı bir sistem olmasına sebep olacaktır. Paketleri bir internet ortamında bir yerde saklayarak, kurmak istediğimizde internet(uzak) üzerinden kurulması daha doğru bir yöntemdir. Bu dağıtımda paketlerimizi github.com üzerinde oluşturulan bir repository üzerinden çekilmektedir. İnternetteki paketlerimizin listesi her yeni paketi yükleme sırasında güncellenmektedir. Bu işlem github hesabı üzerinden yapılmaktadır. github hakkında temel işlemler için github konusunu okuyunuz.

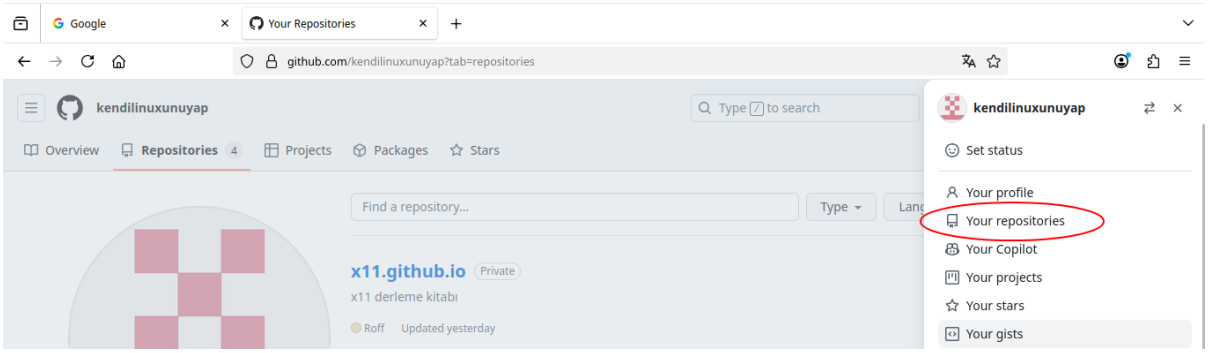
github Sitesi Açılır

Singup seçeneği seçilerek bir hesap oluşturulur. Biz bu dokumanda kullandığımız kendilinuxunuyap hesabını açtık.



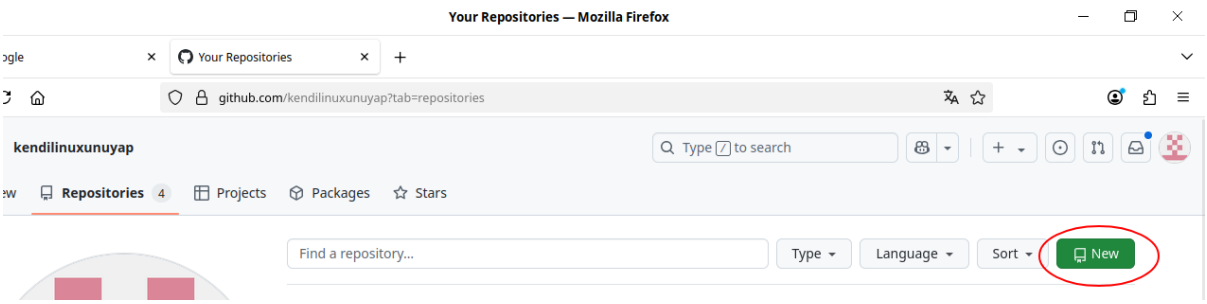
github Üzerinden Hesaba Giriş Yapılır

github hesabı açılır(kendilinuxunuyap) ve sağ tarafta bulunan menüden **Your repostrores** seçilir.

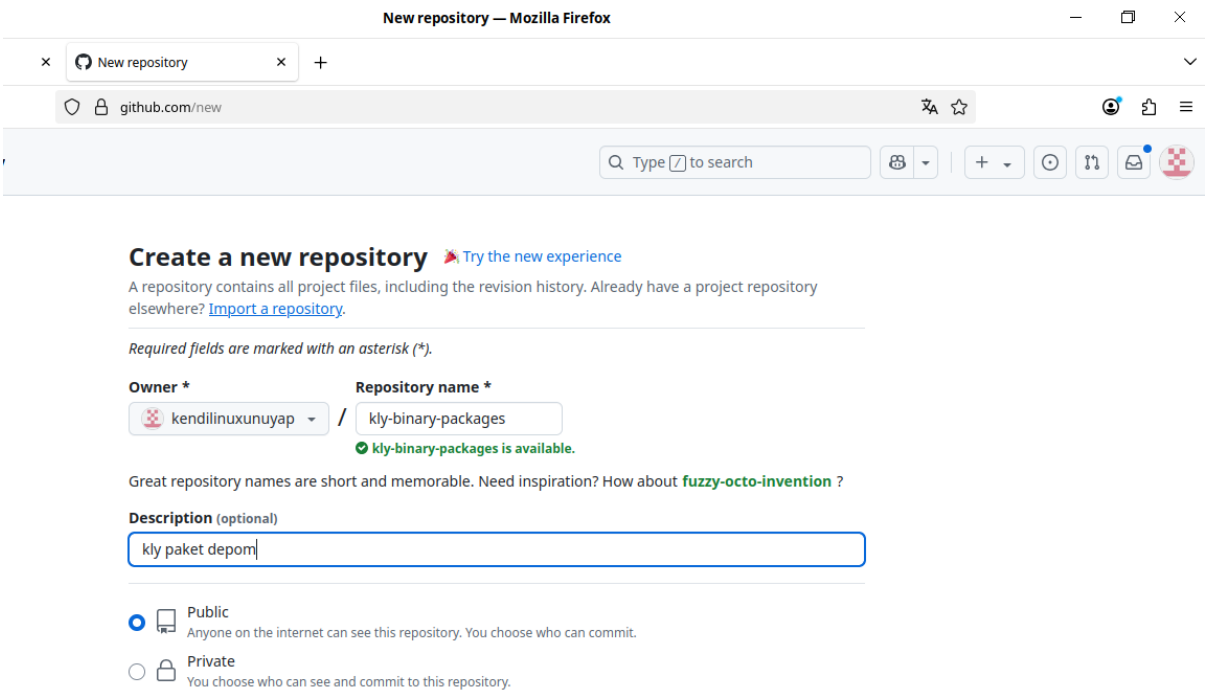


Yeni Depo Alanı Oluşturulur

Karşımıza gelen ekrandan **New** Seçeneği seçilir.

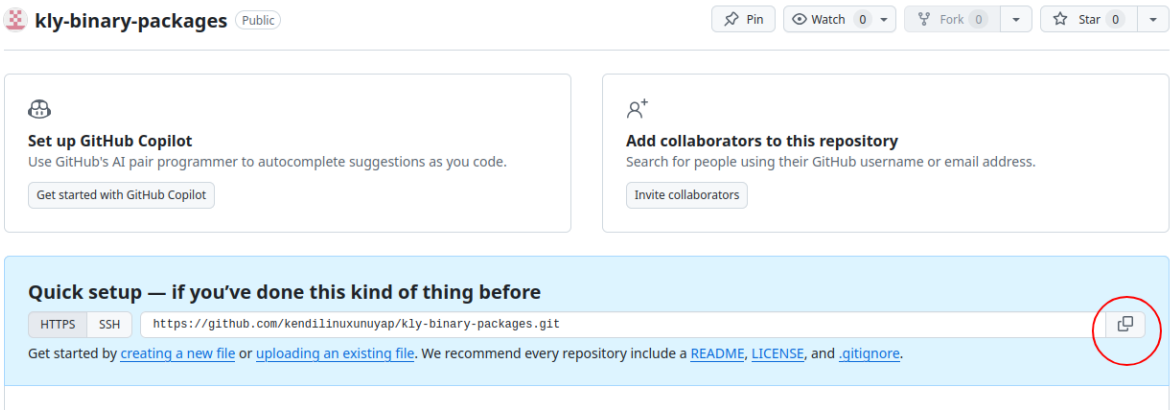


github repository oluşturulur(kly-binary-packages)



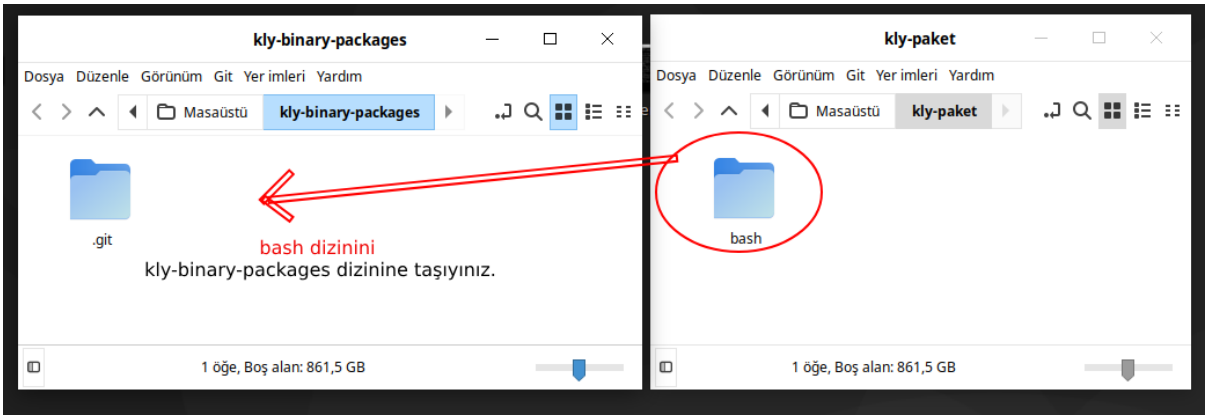
kly-binary-packages Depomuz Yerele(Bilgisayara) Kopyalanır(Clone)

kly-binary-packages adresi kopayanır.

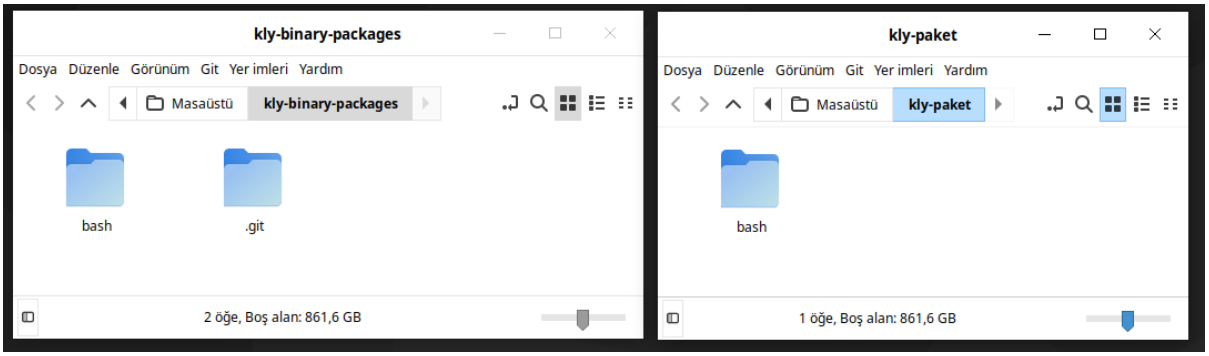


Yerelde(bilgisayarda) istediğiniz yere(masaüstünü tercih ettim) indirilir(klonlanır/download).

Paketimiz kly-binary-packages Dizinine Kopyalanır



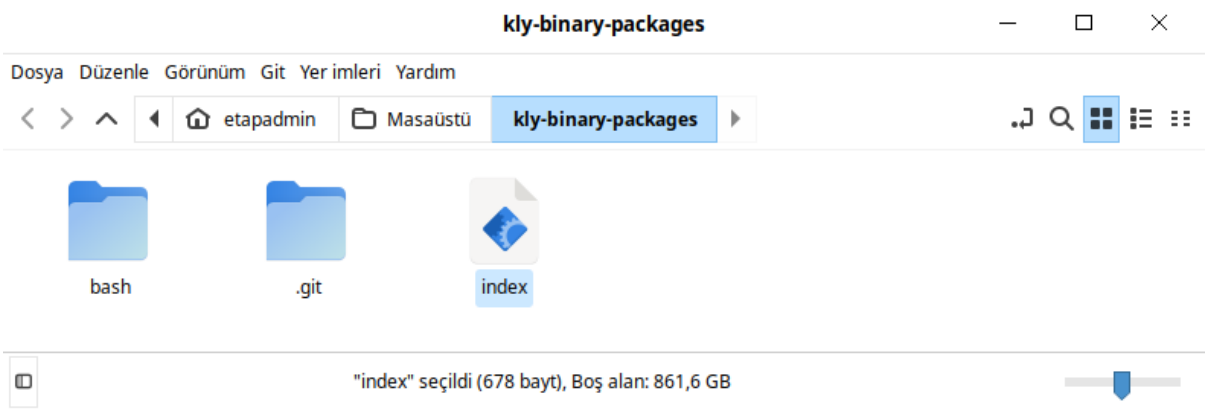
bash paketi aşağıdaki gibi taşınır.



index Dosyası Oluşturulur

Aşağıdaki script kly paket dosyalarımızın olduğu dizinde tek tek açarak içerisinden **klybuild** dosyalarını çıkartır. Paketle ilgili bilgileri alıp **index.lst** dosyası oluşturulmaktadır. İstersek paketlerin local ortamda da index dosyasını oluşturabiliriz. Bu dokümanda github üzerinde oluşturacak şekilde anlatılmıştır. Paket listesinin olduğu **index.lst** dosyası aşağıdaki gibi olacaktır. Listede name, version ve depends(bağımlı olduğu paketler) bilgileri bulunmaktadır. Bilgilerin arasında **|** karekteri kullanılmıştır.

```
name="acl"|version="2.3.1"|depends="attr"|acl
name="attr"|version="2.5.1"|depends=""|attr
name="audit"|version='3.1.1'|depends=""|audit
name="bash"|version="5.2.21"|depends="glibc,readline,ncurses"|bash
```

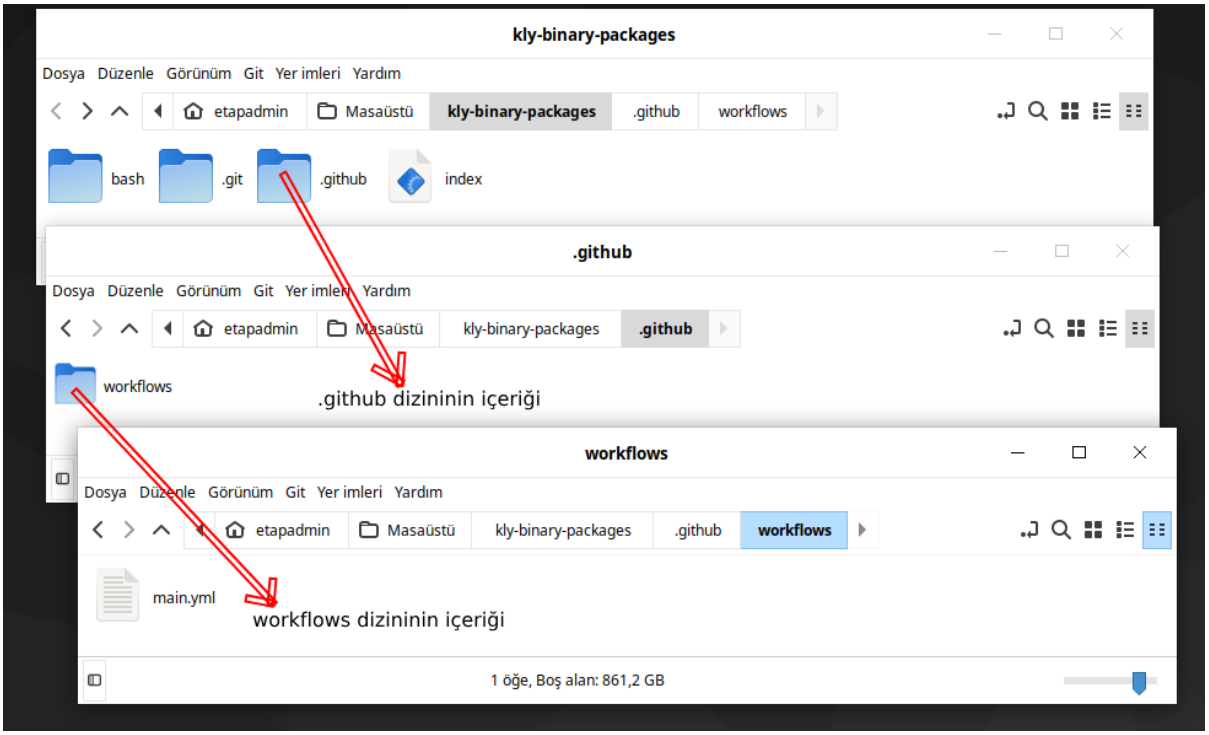


Yukarıda gösterildiği gibi **kly-binary-packages** dizininde aşağıda verilen **index** dosyasını oluşturunuz. Aşağıdaki script kodlarını **index** dosyasının içerisine ekleyin.

```
#-----------------------------------------------------------------------------------------------------------------------------
#!/bin/sh
#set -ex
mkdir /output -p
mkdir -p /klysource
>index.lst
find * -type f -name *.kly |
                while IFS= read file_name; do
                        dosya="$(dirname $file_name)/klybuild"
                        version=$(cat $dosya|grep version=)
                        name=$(cat $dosya|grep name=)
                        depends=$(cat $dosya|grep depends=)
                        echo "$name|$version|$depends|$(dirname $file_name)">>index.lst
                done
cp -rf index.lst /output

# ***************************source files*****************************
cp -prfv ./* /klysource/

find /klysource/* -type f -name *.kly |
                while IFS= read file_name; do
                rm -rf "$file_name"
                done
tar -cf /output/klysourcepackage.tar /klysource/
rm -rf /klysource
```

main.yml Dosyası Oluşturulur

github'a dosya gönderdiğimizde **index** bash scriptimizi çalıştırması için aşağıda gösterilen şekilde **kly-binary-packages** dizinine **.github/workflows** dizinini oluşturun. **.github/workflows** dizini içine **main.yml** dosyasını oluşturunuz.



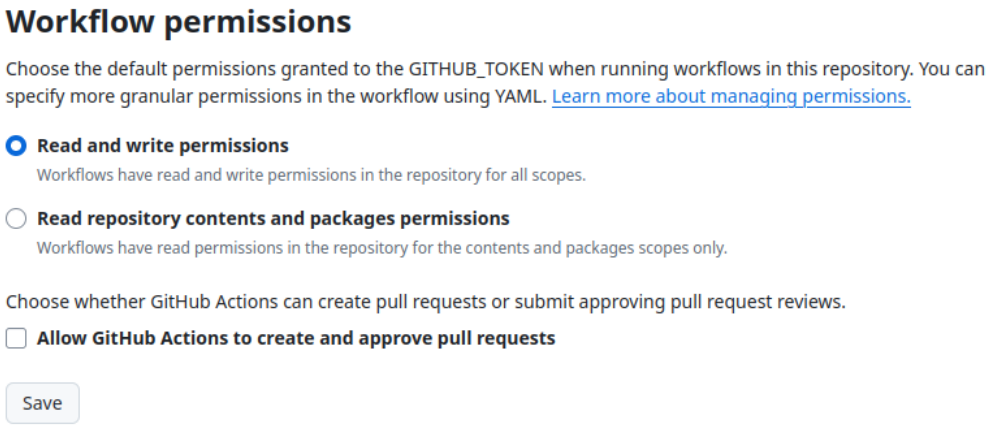
main.yml dosyasındaki **sh index** satırı **index** scriptimizi her githuba paket gönderdiğimizde(commit) çalışacak ve **index.lst** dosyasını oluşturacaktır. **main.yml** içeriğine aşağıdaki kodları ekleyiniz.

```
#-------------------------------------------------------------------------------------------------------------------------------
name: CI

on:
  push:
        branches: [ master ]
  schedule:
        - cron: "0 0 1 2 6"

jobs:
        compile:
            name: depoindex
            runs-on: ubuntu-latest
            steps:
              - name: Check out the repo
                uses: actions/checkout@v2
              - name: Run the build process with Docker
                uses: addnab/docker-run-action@v3
                with:
                    image: debian:testing
                    options: -v ${{ github.workspace }}:/root -v /output:/output
                    run: |
                        cd /root
                        sh index
              - uses: "marvinpinto/action-automatic-releases@latest"
                with:
                    repo_token: "${{ secrets.GITHUB_TOKEN }}"
                    automatic_release_tag: "current"
                    prerelease: false
                    title: "Latest release"
                    files: |
                      /output/*
```

Not: Burada **main.yml** dosyasında **[master]** ifadesi **master** dalında çılışıldığını ifade eder. Eğer farklı dalla çalışıyorsak buradaki **[master]** yerine kullandığınız dalı yazınız.

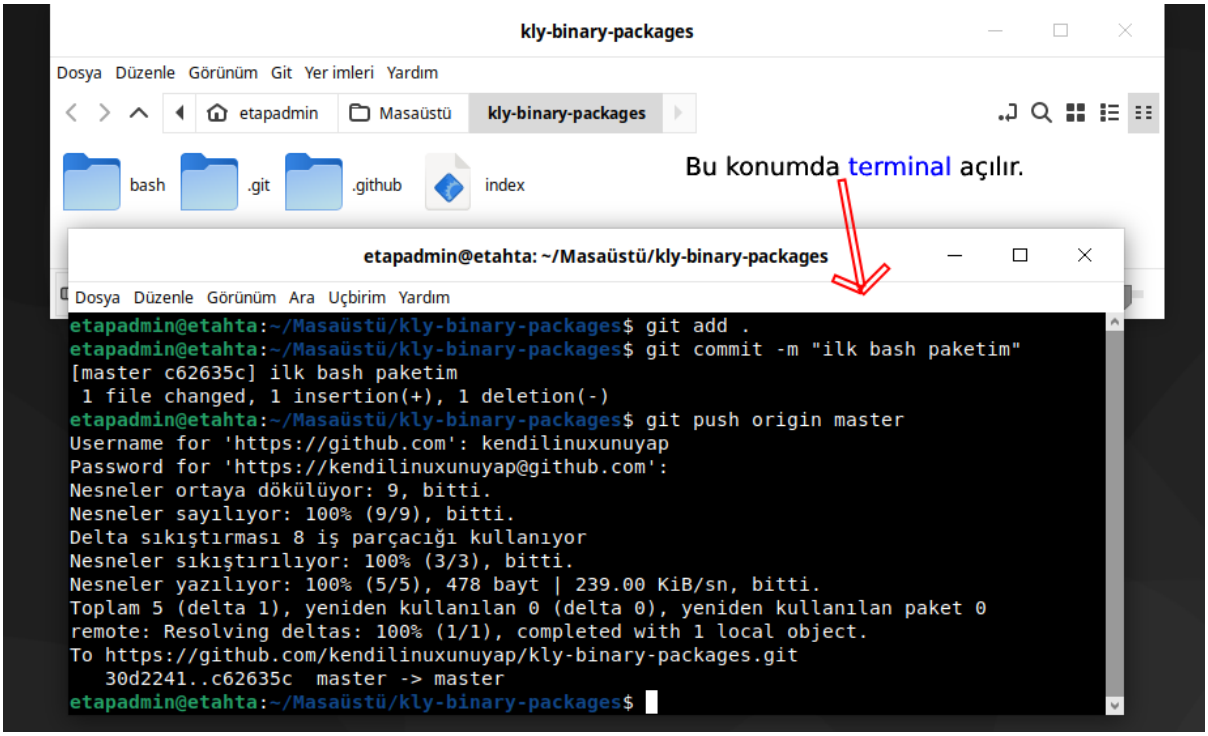
github Dosya oluşturma İzni Verin

İnternet üzerinden **kly-binary-packages** reposunda settings->action->general->Workflow permissions->Read and write permissions işaretlenmelidir.



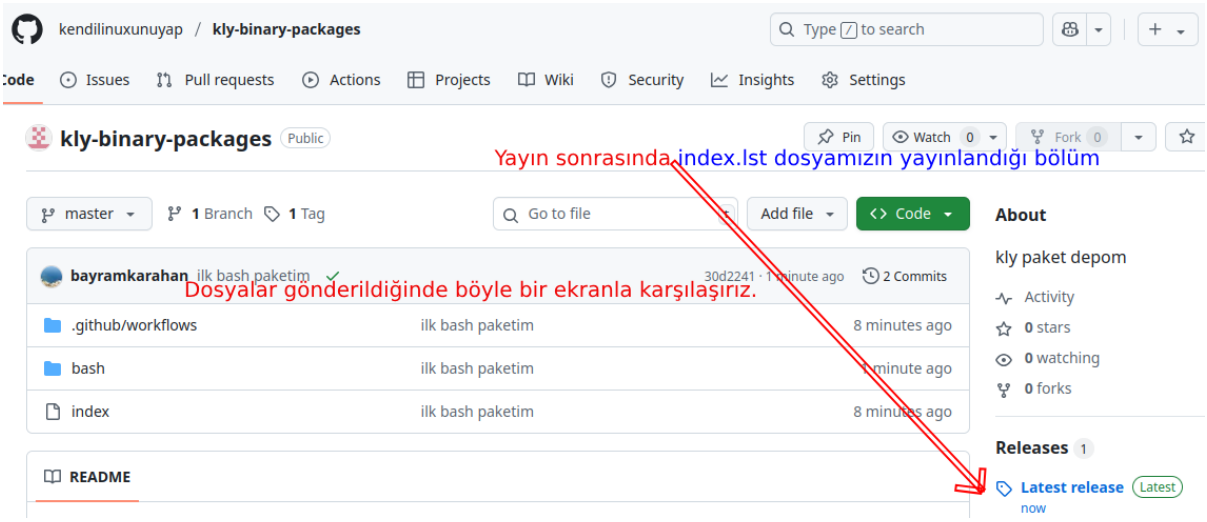
github'a kly-binary-packages Dizinini Yükleyelim(Uplod/Commit)

Yerelde **kly-binary-packages** dizini içeriğini github üzerine aşağıdakı gibi gönderilir.



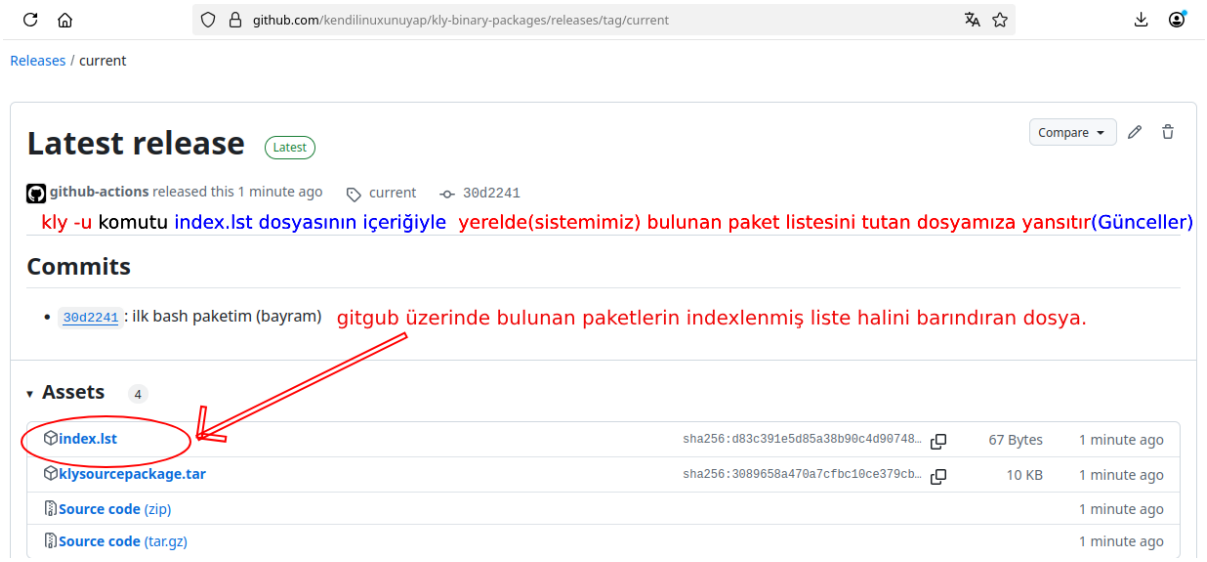
github kly-binary-packages Depo Kontrolü

Depomuza gönderdikten sonra aşağıdaki gibi gözükecektir.



index.lst İçeriği

https://github.com/kendilinuxunuyap/kly-binary-packages/releases/download/current/index.lst adresinde bulunan dosya aşağıdaki gibi liste oluşturacaktır.



index.lst içeriği aşağıdaki gibidir. Tek paket olduğu için(sadece bash) bu şekilde gözüküyor.

```
name="bash"|version="5.2.21"|depends="glibc,readline,ncurses"|bash
```

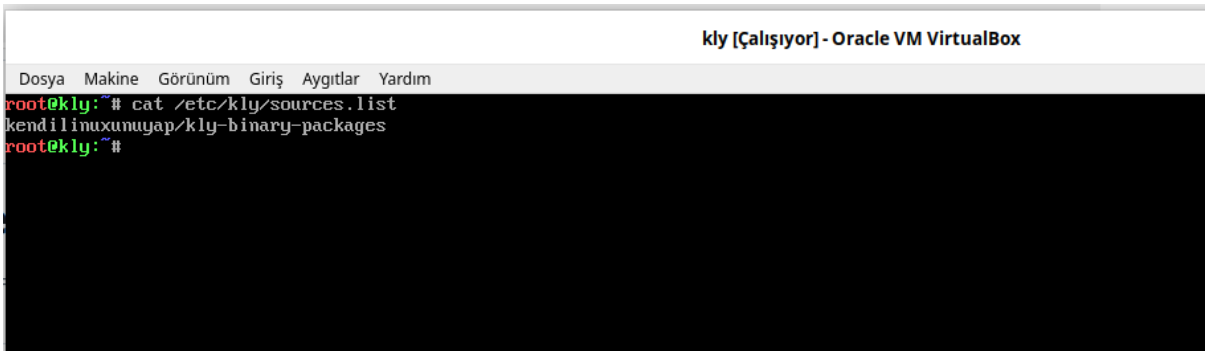
Birden fazla paketin olması durumunda aşağıdaki gibi gözükecektir.

```
name="acl"|version="2.3.1"|depends="attr"|acl
name="attr"|version="2.5.1"|depends=""|attr
name="audit"|version='3.1.1'|depends=""|audit
name="bash"|version="5.2.21"|depends="glibc,readline,ncurses"|bash
```

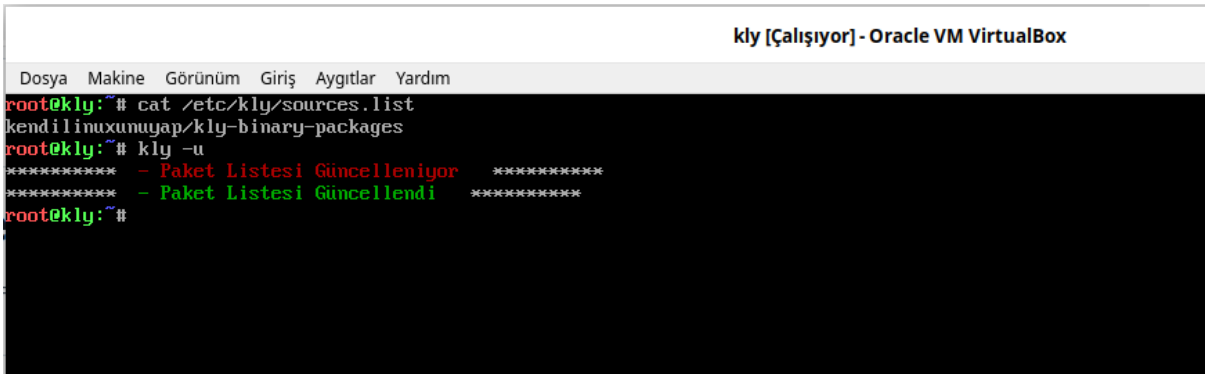
kly ile Paket Listelerini Güncelleme

kly Paket Sistemi yerelde **.kly** paket dosyalarını **kly -pi paket.kly** komutuyla kuralabilir. Ama bütün paketlerin yerelde olması bu sistemi kullanacak kişi sayısını sınırlandıracak ve birkaç kişiyi geçmeyecektir. Paketleri internet ortamında tutmak sistemi kullanacak kişilerin sayısını artıracak ve istenilen zaman ve konumda paketlere erişim imkanı sunacaktır.

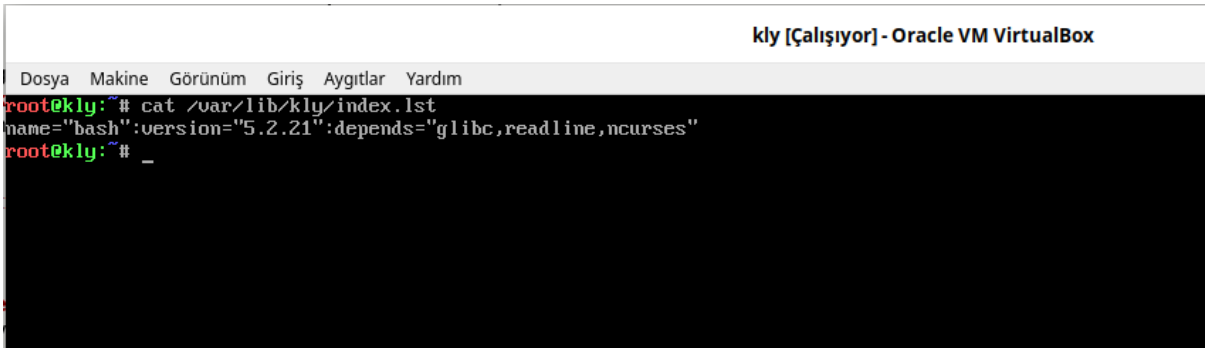
kly paketleri github üzerinde tutulmaktadır. Bu paketlerin listesini tutan index dosyası github ortamında **https://github.com/kendilinuxunuyap/kly-binary-packages/releases/download/current/index.lst** adresinde tutulmaktadır. Bu adresi sisteme kaydetmeliyizki güncelleme sırasında bu adreslerden güncel **index.lst** dosyasını yerele indirebilmeli. Adreslerin listesini tutan dosyamız **/etc/kly/sources.list** konumunda tutulmaktadır. Debianda da benzer(/etc/apt/sources.list) durum bulunmaktadır.



Yerelde **(Temel Sistem)** ise **/var/lib/kly/index.lst** dosyamızda paketlerin listesi tutulur. Mevcut sisteme yeni bir paket yaptık ve github'a yüklediğimizi varsayalım. Bu durumda yerelde **(Temel Sistem)** ise **/var/lib/kly/index.lst** konumundaki dosyanında https://github.com/kendilinuxunuyap/kly-binary-packages/releases/download/current/index.lst adresindeki dosyayla aynı olması gerekir. Bu senaryo tüm linux dağıtımlarında aynıdır. Bu sebeplerden dolayı bir paket kurulum öncesi genelde **güncelleme** işlemi yapılır. Aşağıda güncelleme işleminin yapıldığını göstermektedir.



Aşağıda **/var/lib/kly/index.lst** konumundaki dosyanın içeriğini görmekteyiz.



GNU Araçlarıyla xorg ve x11 Derleme

Ön Hazırlık

Paket derleme işlemi öncesi aşağıdaki konuları bilmemiz gerekmektedir. Bunlar;

1. Derleme(Dinamik/Static)
2. chroot Kullanımı
3. İso Oluşturma
4. ssh Kullanımı
5. sftp Kullanımı
6. scp Kullanımı
7. VirtualBox Kullanmı
8. cfdisk Kullanımı

Burada liste halinde verilen konu başlıkları bu dokümanın **Yardımcı Konular** bölümünde anlatılmaktadır.

Bundan sonraki adımlarda kendi dağıtımımızın **xorg ve x11** pencere sistemini derleyerek **Temel Sistem** üzerinde çalıştıracağız!

GNU Araçlarıyla **xorg ve x11** derleme işlemini **kly Paket Sistemi** kullanılarak derleyeceğiz. Derleme işlemini **kly -c** komutuyla yapacağız. Derlenen paketleri **scp** ve **sftp** kullanarak **Temel Sistem** üzerine kopyalayacağız. **kly -i** komutumuzla kopyaladığımız paketi **Temel Sistem** üzerine kuracağız. Oluşturduğumuz paketleri istersek github'a yükleyip. github üzerinden kururabiliriz.

xorg ve x11'in Çalışması için Gerekli Paketler

0- Ön Hazırlık	25- libX11	50- libinput
1- xorg-server	26- libICE	51- mtdev
2- pixman	27- libXrender	52- libevdev
3- libpciaccess	28- libxcb	53- libwacom
4- libXau	29- libSM	54- libgudev
5- libXdmcp	30- xf86-input-libinput	55- libffi
6- libXfont2	31- xf86-input-vmmouse	56- xinit
7- libxshmfence	32- xf86-video-amdgpu	57- xcalc
8- libdrm	33- xf86-video-ast	58- libXi
9- libxcvt	34- xf86-video-ati	59- openbox
10- libfontenc	35- xf86-video-dummy	60- libXcursor
11- freetype	36- xf86-video-fbdev	61- libXfixes
12- libpng	37- xf86-video-intel	62- pango
13- harfbuzz	38- xf86-video-mga	63- libXrandr
14- glib	39- xf86-video-nouveau	64- fribidi
15- xterm	40- xf86-video-r128	65- xcb-util
16- libXft	41- xf86-video-siliconmotion	66- libthai
17- fontconfig	42- xf86-video-vboxvideo	67- libdatrie
18- dejavu	43- xf86-video-vesa	68- dbus
19- libXext	44- xf86-video-vmware	69- elogind
20- libXaw	45- xkbcomp	70- libunwind
21- libXmu	46- libxkbfile	71-
22- libXinerama	47- libglvnd	72-
23- libXpm	48- startup-notification	73-
24- libXt	49- xkeyboard-config	74-

Bağımlılık Zinciri

Linux paketinin sorunsuz çalışabilmesi için bağımlı olduğu tüm paketlerin önceden derlenmiş olması gerekir. **x11**'in en temel paketleri **xorg-server, mesa, llvm, cairo** paketleridir. Tüm paketleri derlesek bile **xorg-server, mesa, llvm, cairo** paketleri düzgün ve uyumlu versiyonları olmadığı zaman x penceremiz açılmayacaktır. Buradaki tüm paketler ve bağımlılıkları derlendikten sonra **Xorg:0** şeklinde x pencere sistemimiz çalışacaktır.

Derleme Öncesi Hazırlık!

Paket derleme işlemine başlamadan önce, aşağıdaki temel araçları sisteminize kurmalısınız.

```
sudo apt update
sudo apt-get install debootstrap xorriso mtools make squashfs-tools gcc wget unzip xz-utils tar zstd fakeroot \
autoconf automake autotools-dev make meson cmake ninja-build pkgconf patch libtool grub-pc grub-pc-bin
```

xorg-server

Linux sistemlerinde grafik kullanıcı arayüzünün çalışmasını sağlayan bir **görüntü sunucusudur**. X pencere sisteminin en önemli paketidir.

Paketi Derleme :

Debian'da paketi derlemek için aşağıdaki paketlerin kurulu olması gerekir.

```
sudo apt install build-essential autoconf automake libtool pkg-config \
libexpat1-dev python3 libpixman-1-dev libxkbfile-dev libxfont-dev \
libxcvt-dev libxext-dev libxshmfence-dev libbsd-dev libdbus-1-dev \
libgbm-dev libepoxy-dev libaudit-dev xmlto fop \
mesa-common-dev libgl1-mesa-dev libglx-mesa0 libxcb-util-dev \
libxcb-shape0-dev libxcb-render0-dev libxcb-render-util0-dev \
libxcb-image0-dev libxcb-icccm4-dev libxcb-keysyms1-dev libxcb-randr0-dev \
libxcb-xkb-dev libx11-xcb-dev libxcb-xv0-dev xserver-xorg-dev libxcb-input-dev \
libxcb-damage0-dev libxcb-sync-dev libunwind-dev libudev-dev libselinux1-dev xutils-dev

    # xutils-dev (xorg-macros)
```

```
#!/usr/bin/env bash
name="xorg-server"
version="21.1.6"
description="X.Org X servers"
source="https://www.x.org/releases/individual/xserver/xorg-server-$version.tar.xz"
depends="libmd,libbsd,libepoxy,libglvnd,libunwind,libfontenc,libxkbfile,xauth, \
xkbcomp,setxkbmap,freetype,libXfont2,libxcvt,mesa,libX11,libdrm,pixman, \
font-util,xcb-util-renderutil,xcb-util, xcb-util-image,xcb-util-wm,xcb-util-keysyms"
group="x11.base"

setup(){
        cd $SOURCEDIR
        meson setup $BUILDDIR --prefix=/usr \
                --libdir=/usr/lib64/ \
                -Dipv6=true -Dxvfb=true \
                -Dxnest=true -Dxcsecurity=true \
                -Dxorg=true -Dxephyr=true \
                -Dglamor=true -Dudev=true \
                -Ddtrace=false -Dsystemd_logind=false \
                -Dsuid_wrapper=true -Dxkb_dir=/usr/share/X11/xkb \
                -Dxkb_output_dir=/var/lib/xkb
        }

build(){
        ninja -C $BUILDDIR
}

package(){
        DESTDIR=$DESTDIR ninja -C $BUILDDIR install
}
```

Not: Burada verilen derleme talimatı(script) **kly Paket Sistemi**'ni kullanarak paketi derler ve oluştur. Oluşan paket(**.kly uzantılı dosya**) **kly Paket Sistemi** kullanılarak siteme yüklenebilir. **kly Paket Sistemiyle Paket Yapma** konusunu okumak için tıklayınız.

pixman

Pixman, grafik işlemleri için kullanılan kütüphanedir.

Paketi Derleme :

```
#!/usr/bin/env bash
name="pixman"
version="0.42.2"
description="Low-level pixel manipulation routines"
source="https://www.x.org/archive/individual/lib/pixman-$version.tar.gz"
depends=""
group="x11.libs"


setup(){
        $SOURCEDIR/configure --prefix=/usr \
            --libdir=/usr/lib64/
}

build(){
        make
}

package(){
        make install DESTDIR=$DESTDIR
}
```

Not: Burada verilen derleme talimatı(script) **kly Paket Sistemi**'ni kullanarak paketi derler ve oluştur. Oluşan paket(**.kly uzantılı dosya**) **kly Paket Sistemi** kullanılarak siteme yüklenebilir. **kly Paket Sistemiyle Paket Yapma** konusunu okumak için tıklayınız.

libpciaccess

PCI, özellikle Linux/Unix sistemlerde kullanılan; donanım aygıtlarına (özellikle X.Org ve sürücülerde) düşük seviyede erişim sağlayan bir kütüphanedir.

Paketi Derleme :

```
#!/usr/bin/env bash
name="libpciaccess"
version="0.17"
description="Library providing generic access to the PCI bus and devices"
source="https://www.x.org/archive/individual/lib/libpciaccess-$version.tar.xz"
depends=""
group="x11.libs"


setup(){
        $SOURCEDIR/configure --prefix=/usr \
            --libdir=/usr/lib64/
}

build(){
        make
}

package(){
        make install DESTDIR=$DESTDIR
}
```

Not: Burada verilen derleme talimatı(script) **kly Paket Sistemi**'ni kullanarak paketi derler ve oluştur. Oluşan paket(**.kly uzantılı dosya**) **kly Paket Sistemi** kullanılarak siteme yüklenebilir. **kly Paket Sistemiyle Paket Yapma** konusunu okumak için tıklayınız.

libXau

X11 ile kullanılan, istemci ile sunucu arasındaki bağlantıda kimlik doğrulaması yapan bir yetkilendirme kütüphanesidir.

Paketi Derleme :

```
#!/usr/bin/env bash
name="libXau"
version="1.0.11"
description="X.Org X authorization library"
source="https://www.x.org/releases/individual/lib/libXau-${version}.tar.xz"
depends="xorgproto"
builddepend=""
group="x11.libs"


setup(){
        export PKG_CONFIG_PATH=/usr/lib/pkgconfig
        cd $SOURCEDIR
        ./configure --prefix=/usr \
                --sysconfdir=/etc \
                --without-xproto
}

build(){
        make
}

package(){
        make install DESTDIR=$DESTDIR
}
```

Not: Burada verilen derleme talimatı(script) **kly Paket Sistemi**'ni kullanarak paketi derler ve oluştur. Oluşan paket(**.kly uzantılı dosya**) **kly Paket Sistemi** kullanılarak siteme yüklenebilir. **kly Paket Sistemiyle Paket Yapma** konusunu okumak için tıklayınız.

libXdmcp

X11 kapsamında, uzak istemcilerle oturum yönetimini sağlayan bir kütüphanedir

Paketi Derleme :

```
#!/usr/bin/env bash
name="libXdmcp"
version="1.1.5"
description="X.Org X Display Manager Control Protocol library"
source="https://www.x.org/archive/individual/lib/libXdmcp-$version.tar.xz"
depends=""
builddepend="xorgproto util-macros xmlto"
group="x11.libs"


setup(){
        cd $SOURCEDIR
        export PKG_CONFIG_PATH=/usr/lib/pkgconfig
        ./configure --prefix=/usr \
                --sysconfdir=/etc
}

build(){
        make
}

package(){
        make install DESTDIR=$DESTDIR
}
```

Not: Burada verilen derleme talimatı(script) **kly Paket Sistemi**'ni kullanarak paketi derler ve oluştur. Oluşan paket(**.kly uzantılı dosya**) **kly Paket Sistemi** kullanılarak siteme yüklenebilir. **kly Paket Sistemiyle Paket Yapma** konusunu okumak için tıklayınız.

libXfont2

X sunucusunun font kaynaklarına erişmesini ve bunları yönetmesini sağlayan bir kütüphanedir.

Paketi Derleme :

```
#!/usr/bin/env bash
name="libXfont2"
version="2.0.6"
description="X.Org Xfont library"
source="https://www.x.org/archive/individual/lib/libXfont2-$version.tar.xz"
depends=""
group="x11.libs"


setup(){
        $SOURCEDIR/configure --prefix=/usr \
            --libdir=/usr/lib64/
}

build(){
        make
}

package(){
        make install DESTDIR=$DESTDIR
}
```

Not: Burada verilen derleme talimatı(script) **kly Paket Sistemi**'ni kullanarak paketi derler ve oluştur. Oluşan paket(**.kly uzantılı dosya**) **kly Paket Sistemi** kullanılarak siteme yüklenebilir. **kly Paket Sistemiyle Paket Yapma** konusunu okumak için tıklayınız.

libxshmfence

X11 için geliştirilmiş, paylaşılan bellek üzerinden eşzamanlama sağlayan küçük ve özel amaçlı bir kütüphanedir.

Paketi Derleme :

```
#!/usr/bin/env bash
name="libxshmfence"
version="1.3.2"
description="Shared memory fences using futexes"
source="https://www.x.org/archive/individual/lib/libxshmfence-$version.tar.xz"
depends="xorgproto"
group="x11.libs"


setup(){
        $SOURCEDIR/configure --prefix=/usr \
            --libdir=/usr/lib64/
}

build(){
        make
}

package(){
        make install DESTDIR=$DESTDIR
}
```

Not: Burada verilen derleme talimatı(script) **kly Paket Sistemi**'ni kullanarak paketi derler ve oluştur. Oluşan paket(**.kly uzantılı dosya**) **kly Paket Sistemi** kullanılarak siteme yüklenebilir. **kly Paket Sistemiyle Paket Yapma** konusunu okumak için tıklayınız.

libdrm

Linux'ta grafik donanımına güvenli ve doğrudan erişim imkânı sunan bir ara kütüphanedir.

Paketi Derleme :

```
#!/usr/bin/env bash
name="libdrm"
version="2.4.120"
description="X.Org libdrm library"
source="https://dri.freedesktop.org/libdrm/libdrm-$version.tar.xz"
depends="libpciaccess"
group="x11.libs"


setup(){
        cd $SOURCEDIR
        meson setup $BUILDDIR --prefix=/usr \
                -D default_library=both \
                -D udev=false \
                -D etnaviv=disabled \
                -D freedreno=disabled \
                -D vc4=disabled \
                -D valgrind=disabled \
                -D install-test-programs=true
}

build(){
        ninja -C $BUILDDIR
}

package(){
        DESTDIR=$DESTDIR ninja -C $BUILDDIR install
}
```

Not: Burada verilen derleme talimatı(script) **kly Paket Sistemi**'ni kullanarak paketi derler ve oluştur. Oluşan paket(**.kly uzantılı dosya**) **kly Paket Sistemi** kullanılarak siteme yüklenebilir. **kly Paket Sistemiyle Paket Yapma** konusunu okumak için tıklayınız.

libxcvt

VESA CVT ekran çözünürlükleri için kullanılan bir kütüphanedir.

Paketi Derleme :

```
#!/usr/bin/env bash
name="libxcvt"
version="0.1.2"
description="X.Org xcvt library and cvt program"
source="https://gitlab.freedesktop.org/xorg/lib/libxcvt/-\
/archive/libxcvt-$version/libxcvt-libxcvt-$version.tar.gz"
depends=""
group="x11.libs"


setup(){
cd $SOURCEDIR
        meson setup $BUILDDIR --prefix=/usr \
            --libdir=/usr/lib64/
}

build(){
        ninja -C $BUILDDIR
}

package(){
        DESTDIR=$DESTDIR ninja -C $BUILDDIR install
}
```

Not: Burada verilen derleme talimatı(script) **kly Paket Sistemi**'ni kullanarak paketi derler ve oluştur. Oluşan paket(**.kly uzantılı dosya**) **kly Paket Sistemi** kullanılarak siteme yüklenebilir. **kly Paket Sistemiyle Paket Yapma** konusunu okumak için tıklayınız.

libfontenc

X11 sistemlerinde yazı tipi kodlama bilgilerini işleyen bir kütüphanedir.

Paketi Derleme :

```
#!/usr/bin/env bash
name="libfontenc"
version="1.1.7"
description="PX.Org fontenc library"
source="https://www.x.org/archive/individual/lib/libfontenc-$version.tar.xz"
depends=""
group="x11.libs"


setup(){
        $SOURCEDIR/configure --prefix=/usr \
            --libdir=/usr/lib64/
}

build(){
        make
}

package(){
        make install DESTDIR=$DESTDIR
}
```

Not: Burada verilen derleme talimatı(script) **kly Paket Sistemi**'ni kullanarak paketi derler ve oluştur. Oluşan paket(**.kly uzantılı dosya**) **kly Paket Sistemi** kullanılarak siteme yüklenebilir. **kly Paket Sistemiyle Paket Yapma** konusunu okumak için tıklayınız.

freetype

Yazı tipi motorudur.

Paketi Derleme :

Debian'da paketi derlemek için aşağıdaki paketlerin kurulu olması gerekir.

```
sudo apt install libharfbuzz-dev
```

```
#!/usr/bin/env bash
name="freetype"
version="2.13.1"
description="Package freetype"
source="https://download.savannah.gnu.org/releases/freetype/freetype-$version.tar.gz"
depends="zlib,bzip2,glib,libpng,harfbuzz"
group="media.libs"

setup(){
        $SOURCEDIR/configure --prefix=/usr \
            --libdir=/usr/lib64/ \
                --with-harfbuzz=yes
}

build(){
        make
}

package(){
        make install DESTDIR=$DESTDIR
}
```

Not: Burada verilen derleme talimatı(script) **kly Paket Sistemi**'ni kullanarak paketi derler ve oluştur. Oluşan paket(**.kly uzantılı dosya**) **kly Paket Sistemi** kullanılarak siteme yüklenebilir. **kly Paket Sistemiyle Paket Yapma** konusunu okumak için tıklayınız.

libpng

Portable Network Graphics (PNG) formatındaki resim için kütüphanedir.

Paketi Derleme :

```
#!/usr/bin/env bash
name="libpng"
version="1.6.39"
description="Portable Network Graphics library"
source="https://salsa.debian.org/debian/libpng1.6/-\
/archive/debian/$version-2/libpng1.6-debian-$version-2.tar.gz"
depends="zlib"
group="media.libs"


setup(){
        cd $SOURCEDIR
        ./configure --prefix=/usr \
            --libdir=/usr/lib64/
}

build(){
        make
}

package(){
        make install DESTDIR=$DESTDIR
}
```

Not: Burada verilen derleme talimatı(script) **kly Paket Sistemi**'ni kullanarak paketi derler ve oluştur. Oluşan paket(**.kly uzantılı dosya**) **kly Paket Sistemi** kullanılarak siteme yüklenebilir. **kly Paket Sistemiyle Paket Yapma** konusunu okumak için tıklayınız.

harfbuzz

Metin şekillendirme (text shaping) kütüphanesi.

Paketi Derleme :

Debian'da paketi derlemek için aşağıdaki paketlerin kurulu olması gerekir.

```
sudo apt install libcairo2-dev gtk-doc-tools libgirepository1.0-dev libchafa-dev
```

```
#!/usr/bin/env bash
name="harfbuzz"
version="8.3.0"
description="HarfBuzz text shaping engine"
source="https://github.com/harfbuzz/harfbuzz/archive/refs/tags/$version.tar.gz"
depends="cairo,glib"
group="media.libs"

setup(){
        cd $SOURCEDIR
        meson setup $BUILDDIR --prefix=/usr \
            --libdir=/usr/lib64/ \
            -Dglib=enabled \
                -Dgobject=enabled \
                -Dicu=enabled \
                -Dfreetype=enabled \
                -Dtests=disabled \
                -Dcairo=enabled \
                -Ddocs=enabled
}

build(){
        ninja -C $BUILDDIR
}

package(){
        DESTDIR=$DESTDIR ninja -C $BUILDDIR install
}
```

Not: Burada verilen derleme talimatı(script) **kly Paket Sistemi**'ni kullanarak paketi derler ve oluştur. Oluşan paket(**.kly uzantılı dosya**) **kly Paket Sistemi** kullanılarak siteme yüklenebilir. **kly Paket Sistemiyle Paket Yapma** konusunu okumak için tıklayınız.

glib

GNOME tarafından geliştirilen, C programlama dili için temel kütüphanedir.

Paketi Derleme :

```
#!/usr/bin/env bash
name="glib"
version="2.78.0"
description="Low-level core library that forms the basis for projects\
 such as GTK+ and GNOME."
source="https://download.gnome.org/sources/glib/2.78/glib-${version}.tar.xz"
depends="bzip2,gettext,python,py3-packaging"
builddepend="bison,flex,libffi,meson,pcre2,py3-setuptools,py3-docutils,util-linux"
group="dev.libs"


setup(){
cd $SOURCEDIR
        cp $PACKAGEDIR/files/* $SOURCEDIR
        meson setup $BUILDDIR --prefix=/usr \
            --libdir=/usr/lib64 \
            --default-library both \
            -D glib_debug=disabled \
            -D selinux=disabled \
            -D sysprof=disabled
}

build(){
        ninja -C $BUILDDIR
}

package(){
        DESTDIR=$DESTDIR ninja -C $BUILDDIR install

}
```

Ek dosyaları indirmek için tıklayınız..

Not: Burada verilen derleme talimatı(script) **kly Paket Sistemi**'ni kullanarak paketi derler ve oluştur. Oluşan paket(**.kly uzantılı dosya**) **kly Paket Sistemi** kullanılarak siteme yüklenebilir. **kly Paket Sistemiyle Paket Yapma** konusunu okumak için tıklayınız.

xterm

Terminal emülatörüdür.

Paketi Derleme :

Debian'da paketi derlemek için aşağıdaki paketlerin kurulu olması gerekir.

```
sudo apt install  libxt-dev libxinerama-dev libxaw7-dev
```

```
#!/usr/bin/env bash
name="xterm"
version="388"
description="Terminal Emulator for X Windows"
depends="libX11,ncurses,util-linux,libXt,libXau,libXinerama,libXaw,libXpm,libXft"
source="https://invisible-island.net/archives/xterm/xterm-$version.tgz"
group="x11.terms"


setup(){
        cp -prfv $PACKAGEDIR/files/* $SOURCEDIR

        $SOURCEDIR/configure --prefix=/usr --libdir=/usr/lib64/ --exec-prefix=/usr \
                --with-app-defaults=/etc/X11/app-defaults --with-icondir=/usr/share/icons \
                --with-icon-theme=yes --with-tty-group=tty --enable-warnings --enable-logging \
                --enable-wide-chars --enable-luit --enable-256-color --disable-imake \
                --enable-narrowproto --enable-exec-xterm --enable-dabbrev --enable-backarrow-is-erase \
                --enable-sixel-graphics --with-utempter --with-desktop-category=System,TerminalEmulator
}

build(){
        make
}

package(){
        make install DESTDIR=$DESTDIR
        make install-ti $jobs DESTDIR=${DESTDIR}
        mkdir -p "${DESTDIR}"/usr/share/applications
        mkdir -p "${DESTDIR}"/usr/share/xgreeters
        install $SOURCEDIR/{xterm,uxterm}.desktop "${DESTDIR}"/usr/share/applications/
        install $SOURCEDIR/xterm.desktop "${DESTDIR}"/usr/share/xgreeters/
}
```

Ek dosyaları indirmek için tıklayınız..

Not: Burada verilen derleme talimatı(script) **kly Paket Sistemi**'ni kullanarak paketi derler ve oluştur. Oluşan paket(**.kly uzantılı dosya**) **kly Paket Sistemi** kullanılarak siteme yüklenebilir. **kly Paket Sistemiyle Paket Yapma** konusunu okumak için tıklayınız.

libXft

Yazı tipi motorunu kullannan bir kütüphanedir.

Paketi Derleme :

```
#!/usr/bin/env bash
name="libXft"
version="2.3.7"
description="XFreeType-based font drawing library for X"
source="https://www.x.org/archive/individual/lib/libXft-$version.tar.xz"
depends="fontconfig,libXrender"
group="x11.libs"


setup(){
    $SOURCEDIR/configure --prefix=/usr \
        --libdir=/usr/lib64/
}

build(){
    make
}

package(){
    make install DESTDIR=$DESTDIR
}
```

Not: Burada verilen derleme talimatı(script) **kly Paket Sistemi**'ni kullanarak paketi derler ve oluştur. Oluşan paket(**.kly uzantılı dosya**) **kly Paket Sistemi** kullanılarak siteme yüklenebilir. **kly Paket Sistemiyle Paket Yapma** konusunu okumak için tıklayınız.

fontconfig

Sistemdeki tüm yazı tiplerini yönetimini ve kullanımını kolaylaştıran paket.

Paketi Derleme :

```
#!/usr/bin/env bash
name="fontconfig"
version="2.15.0"
description="Fontconfig is a library for configuring and customizing font access."
source="https://www.freedesktop.org/software/fontconfig/release/fontconfig-$version.tar.gz"
depends="freetype,expat"
group="media.libs"


setup(){
cd $SOURCEDIR
        cp -prfv $PACKAGEDIR/files/* $SOURCEDIR
        meson setup $BUILDDIR --prefix=/usr \
            --libdir=/usr/lib64/
}

build(){
        ninja -C $BUILDDIR
}

package(){
        DESTDIR=$DESTDIR ninja -C $BUILDDIR install
        # fix symlinks
        for link in $(ls ${DESTDIR}/etc/fonts/conf.d/) ; do
            if [[ -L ${DESTDIR}/etc/fonts/conf.d/$link ]] ; then
            rm -f ${DESTDIR}/etc/fonts/conf.d/$link
            ln -s ../../../usr/share/fontconfig/conf.avail/$link ${DESTDIR}/etc/fonts/conf.d/$link
            fi
        done

}
```

Not: Burada verilen derleme talimatı(script) **kly Paket Sistemi**'ni kullanarak paketi derler ve oluştur. Oluşan paket(**.kly uzantılı dosya**) **kly Paket Sistemi** kullanılarak siteme yüklenebilir. **kly Paket Sistemiyle Paket Yapma** konusunu okumak için tıklayınız.

dejavu

Yazı tipi.

Paketi Derleme :

```
#!/usr/bin/env bash
name="dejavu"
version="2.37"
description="DejaVu fonts, bitstream vera with ISO-8859-2 characters"
source="https://github.com/dejavu-fonts/dejavu-fonts/releases/download/\
version_${version/./_}/dejavu-lgc-fonts-ttf-$version.zip"
depends=""
group="media.fonts"
setup()
{
        /bin/busybox wget "https://github.com/dejavu-fonts/dejavu-fonts/releases/\
        download/version_${version/./_}/dejavu-fonts-ttf-$version.zip"

        /bin/busybox unzip dejavu-fonts-ttf-$version.zip
        /bin/busybox rm dejavu-fonts-ttf-$version.zip
        /bin/busybox cp -prfv $SOURCEDIR ./
}
build()
{
echo ""
}
package(){
        mkdir -p ${DESTDIR}/usr/share/fonts
        for font in dejavu dejavu-fonts-ttf; do
            cp -prfv $BUILDDIR/$font-$version/ttf ${DESTDIR}/usr/share/fonts/$font
        done
}
```

Not: Burada verilen derleme talimatı(script) **kly Paket Sistemi**'ni kullanarak paketi derler ve oluştur. Oluşan paket(**.kly uzantılı dosya**) **kly Paket Sistemi** kullanılarak siteme yüklenebilir. **kly Paket Sistemiyle Paket Yapma** konusunu okumak için tıklayınız.

libXext

Xorg için ek kütüphanedir.

Paketi Derleme :

```
#!/usr/bin/env bash
name="libXext"
version="1.3.5"
description="X.Org Xext library"
source="https://www.x.org/archive/individual/lib/libXext-$version.tar.xz"
depends="libX11,xorgproto"
group="x11.libs"


setup(){
        $SOURCEDIR/configure --prefix=/usr \
            --libdir=/usr/lib64/
}

build(){
        make
}

package(){
        make install DESTDIR=$DESTDIR
}
```

Not: Burada verilen derleme talimatı(script) **kly Paket Sistemi**'ni kullanarak paketi derler ve oluştur. Oluşan paket(**.kly uzantılı dosya**) **kly Paket Sistemi** kullanılarak siteme yüklenebilir. **kly Paket Sistemiyle Paket Yapma** konusunu okumak için tıklayınız.

libXaw

Grafik kullanıcı arayüzü (GUI) kütüphanesidir.

Paketi Derleme :

```
#!/usr/bin/env bash
name="libXaw"
version="1.0.14"
description="Package libXaw"
source="https://www.x.org/archive/individual/lib/libXaw-$version.tar.gz"
depends="libXext,libXt,libXmu,libXpm"
group="x11.libs"


setup(){
        $SOURCEDIR/configure --prefix=/usr \
            --libdir=/usr/lib64/
}

build(){
        make
}

package(){
        make install DESTDIR=$DESTDIR
}
```

Not: Burada verilen derleme talimatı(script) **kly Paket Sistemi**'ni kullanarak paketi derler ve oluştur. Oluşan paket(**.kly uzantılı dosya**) **kly Paket Sistemi** kullanılarak siteme yüklenebilir. **kly Paket Sistemiyle Paket Yapma** konusunu okumak için tıklayınız.

libXmu

X11 pencere sistemi için gerekli ek kütüphanedir.

Paketi Derleme :

```
#!/usr/bin/env bash
name="libXmu"
version="1.1.4"
description="X.Org Xmu library"
source="https://www.x.org/archive/individual/lib/libXmu-$version.tar.xz"
depends="libXt,libXext,libX11"
group="x11.libs"

setup(){
    $SOURCEDIR/configure --prefix=/usr \
        --libdir=/usr/lib64/
}

build(){
    make
}

package(){
    make install DESTDIR=$DESTDIR
}
```

Not: Burada verilen derleme talimatı(script) **kly Paket Sistemi**'ni kullanarak paketi derler ve oluştur. Oluşan paket(**.kly uzantılı dosya**) **kly Paket Sistemi** kullanılarak siteme yüklenebilir. **kly Paket Sistemiyle Paket Yapma** konusunu okumak için tıklayınız.

libXinerama

Moniör için gerekli kütüphanedir.

Paketi Derleme :

```
#!/usr/bin/env bash
name="libXinerama"
version="1.1.5"
description="X.Org Xinerama library"
source="https://www.x.org/archive/individual/lib/libXinerama-$version.tar.xz"
depends=""
group="x11.libs"

setup(){
    $SOURCEDIR/configure --prefix=/usr \
        --libdir=/usr/lib64/
}

build(){
    make
}

package(){
    make install DESTDIR=$DESTDIR
}
```

Not: Burada verilen derleme talimatı(script) **kly Paket Sistemi**'ni kullanarak paketi derler ve oluştur. Oluşan paket(**.kly uzantılı dosya**) **kly Paket Sistemi** kullanılarak siteme yüklenebilir. **kly Paket Sistemiyle Paket Yapma** konusunu okumak için tıklayınız.

libXpm

XPM (X PixMap) formatındaki dosyalarını işlemek üzere kullanılan bir kütüphanedir.

Paketi Derleme :

```
#!/usr/bin/env bash
name="libXpm"
version="3.5.14"
description="X.Org Xpm library"
source="https://www.x.org/archive/individual/lib/libXpm-$version.tar.xz"
depends="libX11"
group="x11.libs"

setup(){
    $SOURCEDIR/configure --prefix=/usr \
        --libdir=/usr/lib64/
}

build(){
    make
}

package(){
    make install DESTDIR=$DESTDIR
}
```

Not: Burada verilen derleme talimatı(script) **kly Paket Sistemi**'ni kullanarak paketi derler ve oluştur. Oluşan paket(**.kly uzantılı dosya**) **kly Paket Sistemi** kullanılarak siteme yüklenebilir. **kly Paket Sistemiyle Paket Yapma** konusunu okumak için tıklayınız.

libXt

GUI (grafik kullanıcı arayüzü) uygulamaları için gerekli bir kütüphanedir.

Paketi Derleme :

```
#!/usr/bin/env bash
name="libXt"
version="1.3.0"
description="X.Org X Toolkit Intrinsics library"
source="https://www.x.org/archive/individual/lib/libXt-$version.tar.gz"
depends="libSM,libX11"
group="x11.libs"

setup(){
    $SOURCEDIR/configure --prefix=/usr \
        --libdir=/usr/lib64/
}

build(){
    make
}

package(){
    make install DESTDIR=$DESTDIR
}
```

Not: Burada verilen derleme talimatı(script) **kly Paket Sistemi**'ni kullanarak paketi derler ve oluştur. Oluşan paket(**.kly uzantılı dosya**) **kly Paket Sistemi** kullanılarak siteme yüklenebilir. **kly Paket Sistemiyle Paket Yapma** konusunu okumak için tıklayınız.

libX11

X11 pencere sistemi için temel kütüphanedir.

Paketi Derleme :

```
#!/usr/bin/env bash
name="libX11"
version="1.8"
description="X.Org X11 library"
source="https://www.x.org/archive/individual/lib/libX11-$version.tar.xz"
depends="libxcb,libXau,libXdmcp,xtrans"
group="x11.libs"

setup(){
    $SOURCEDIR/configure --prefix=/usr \
        --libdir=/usr/lib64/
}

build(){
    make
}

package(){
    make install DESTDIR=$DESTDIR
}
```

Not: Burada verilen derleme talimatı(script) **kly Paket Sistemi**'ni kullanarak paketi derler ve oluştur. Oluşan paket(**.kly uzantılı dosya**) **kly Paket Sistemi** kullanılarak siteme yüklenebilir. **kly Paket Sistemiyle Paket Yapma** konusunu okumak için tıklayınız.

libICE

Uygulamalar arasında protokol tabanlı iletişimi sağlayan bir kütüphanedir.

Paketi Derleme :

```
#!/usr/bin/env bash
name="libICE"
version="1.1.1"
description="X.Org Inter-Client Exchange library"
source="https://www.x.org/archive/individual/lib/libICE-$version.tar.xz"
depends="xorgproto"
group="x11.libs"

setup(){
    $SOURCEDIR/configure --prefix=/usr \
        --libdir=/usr/lib64/
}

build(){
    make
}

package(){
    make install DESTDIR=$DESTDIR
}
```

Not: Burada verilen derleme talimatı(script) **kly Paket Sistemi**'ni kullanarak paketi derler ve oluştur. Oluşan paket(**.kly uzantılı dosya**) **kly Paket Sistemi** kullanılarak siteme yüklenebilir. **kly Paket Sistemiyle Paket Yapma** konusunu okumak için tıklayınız.

libXrender

2D grafikler için gerekli bir kütüphanedir.

Paketi Derleme :

```
#!/usr/bin/env bash
name="libXrender"
version="0.9.11"
description="X.Org Xrender library"
source="https://www.x.org/archive/individual/lib/libXrender-$version.tar.xz"
depends=""
group="x11.libs"

setup(){
    $SOURCEDIR/configure --prefix=/usr \
        --libdir=/usr/lib64/
}

build(){
    make
}

package(){
    make install DESTDIR=$DESTDIR
}
```

Not: Burada verilen derleme talimatı(script) **kly Paket Sistemi**'ni kullanarak paketi derler ve oluştur. Oluşan paket(**.kly uzantılı dosya**) **kly Paket Sistemi** kullanılarak siteme yüklenebilir. **kly Paket Sistemiyle Paket Yapma** konusunu okumak için tıklayınız.

libxcb

X11 için geerewkli temel kütüphanedir.

Paketi Derleme :

Debian'da paketi derlemek için aşağıdaki paketlerin kurulu olması gerekir.

```
sudo apt install xutils-dev doxygen xcb-proto_1.17.0 python3-xcbgen
```

```
#!/usr/bin/env bash
name="libxcb"
version="1.17.0"
description="X C-language Bindings library"
#source="https://www.x.org/releases/individual/lib/libxcb-$version.tar.xz"
source="https://gitlab.freedesktop.org/xorg/lib/libxcb/-\
/archive/libxcb-$version/libxcb-libxcb-$version.tar.gz"
depends="libXau,libXdmcp,xcb-proto"
builddepend="libxslt,python3,util-macros"
group="x11.libs"

setup(){
    cd $SOURCEDIR
    #export PKG_CONFIG_PATH=/usr/lib/pkgconfig
    autoreconf -fvi
    ./configure --prefix=/usr \
            --libdir=/usr/lib64 \
            --enable-xinput \
            --enable-xkb \
            --disable-static
}

build(){
    make
}

package(){
    make install DESTDIR=$DESTDIR
}
```

Not: Burada verilen derleme talimatı(script) **kly Paket Sistemi**'ni kullanarak paketi derler ve oluştur. Oluşan paket(**.kly uzantılı dosya**) **kly Paket Sistemi** kullanılarak siteme yüklenebilir. **kly Paket Sistemiyle Paket Yapma** konusunu okumak için tıklayınız.

libSM

X11 pencere sisteminde oturum yönetimi için gerekli kütüphanedir.

Paketi Derleme :

Debian'da paketi derlemek için aşağıdaki paketlerin kurulu olması gerekir.

```
sudo apt install libice-dev_1.1.1
```

```
#!/usr/bin/env bash
name="libSM"
version="1.2.4"
description="libSM X.Org Session Management library"
source="https://www.x.org/archive/individual/lib/libSM-$version.tar.xz"
depends="xorgproto,libICE"
group="x11.libs"

setup(){
    $SOURCEDIR/configure --prefix=/usr \
        --libdir=/usr/lib64
}

build(){
    make
}

package(){
    make install DESTDIR=$DESTDIR
}
```

Not: Burada verilen derleme talimatı(script) **kly Paket Sistemi**'ni kullanarak paketi derler ve oluştur. Oluşan paket(**.kly uzantılı dosya**) **kly Paket Sistemi** kullanılarak siteme yüklenebilir. **kly Paket Sistemiyle Paket Yapma** konusunu okumak için tıklayınız.

xf86-input-libinput

X11 için giriş aygıtlarının yönetimini sağlayan bir kütüphanedir.

Paketi Derleme :

```
#!/usr/bin/env bash
name="xf86-input-libinput"
version="1.4.0"
description="X.org input driver based on libinput"
source="https://gitlab.freedesktop.org/xorg/driver/xf86-input-libinput/-/archive/\
xf86-input-libinput-$version/xf86-input-libinput-xf86-input-libinput-$version.tar.gz"
depends="libinput"
group="x11.drivers"

setup(){
    cd $SOURCEDIR
    ./autogen.sh
    ./configure --prefix=/usr \
        --libdir=/usr/lib64/
}

build(){
    make
}

package(){
    make install DESTDIR=$DESTDIR
}
```

Not: Burada verilen derleme talimatı(script) **kly Paket Sistemi**'ni kullanarak paketi derler ve oluştur. Oluşan paket(**.kly uzantılı dosya**) **kly Paket Sistemi** kullanılarak siteme yüklenebilir. **kly Paket Sistemiyle Paket Yapma** konusunu okumak için tıklayınız.

xf86-input-vmmouse

Sanallaştırılmış ortamlarda fareyi yönetmek için kullanılır.

Paketi Derleme :

Debian'da paketi derlemek için aşağıdaki paketlerin kurulu olması gerekir.

```
sudo apt install xserver-xorg-dev
```

```
#!/usr/bin/env bash
name="xf86-input-vmmouse"
version="13.2.0"
description="VMWare mouse input driver"
source="https://gitlab.freedesktop.org/xorg/driver/xf86-input-vmmouse/-/archive/\
xf86-input-vmmouse-$version/xf86-input-vmmouse-xf86-input-vmmouse-$version.tar.gz"
depends=""
group="x11.drivers"

setup(){
    cd $SOURCEDIR
    ./autogen.sh
    ./configure --prefix=/usr \
        --libdir=/usr/lib64/
}

build(){
    make
}

package(){
    make install DESTDIR=$DESTDIR
}
```

Not: Burada verilen derleme talimatı(script) **kly Paket Sistemi**'ni kullanarak paketi derler ve oluştur. Oluşan paket(**.kly uzantılı dosya**) **kly Paket Sistemi** kullanılarak siteme yüklenebilir. **kly Paket Sistemiyle Paket Yapma** konusunu okumak için tıklayınız.

xf86-video-amdgpu

AMD grafik kartları için X11 video sürücüsüdür.

Paketi Derleme :

```
#!/usr/bin/env bash
name="xf86-video-amdgpu"
version="23.0.0"
description="Accelerated Open Source driver for AMDGPU cards"
source="https://gitlab.freedesktop.org/xorg/driver/xf86-video-amdgpu/-/archive/\
xf86-video-amdgpu-$version/xf86-video-amdgpu-xf86-video-amdgpu-$version.tar.gz"
depends=""
group="x11.drivers"

setup(){
    cd $SOURCEDIR
    ./autogen.sh
    ./configure --prefix=/usr \
        --libdir=/usr/lib64/
}

build(){
    make
}

package(){
    make install DESTDIR=$DESTDIR
}
```

Not: Burada verilen derleme talimatı(script) **kly Paket Sistemi**'ni kullanarak paketi derler ve oluştur. Oluşan paket(**.kly uzantılı dosya**) **kly Paket Sistemi** kullanılarak siteme yüklenebilir. **kly Paket Sistemiyle Paket Yapma** konusunu okumak için tıklayınız.

xf86-video-ast

AST (Advanced Systems Technology) grafik kartları için video sürücüsüdür.

Paketi Derleme :

```
#!/usr/bin/env bash
name="xf86-video-ast"
version="1.1.6"
description="X.Org driver for ASpeedTech cards"
source="https://gitlab.freedesktop.org/xorg/driver/xf86-video-ast/-/archive/\
xf86-video-ast-$version/xf86-video-ast-xf86-video-ast-$version.tar.gz"
depends=""
group="x11.drivers"

setup(){
    cd $SOURCEDIR
    ./autogen.sh
    ./configure --prefix=/usr \
        --libdir=/usr/lib64/
}

build(){
    make
}

package(){
    make install DESTDIR=$DESTDIR
}
```

Not: Burada verilen derleme talimatı(script) **kly Paket Sistemi**'ni kullanarak paketi derler ve oluştur. Oluşan paket(**.kly uzantılı dosya**) **kly Paket Sistemi** kullanılarak siteme yüklenebilir. **kly Paket Sistemiyle Paket Yapma** konusunu okumak için tıklayınız.

xf86-video-ati

AMD/ATI grafik kartları için video sürücüsüdür.

Paketi Derleme :

```
#!/usr/bin/env bash
name="xf86-video-ati"
version="22.0.0"
description="ATI video driver"
source="https://gitlab.freedesktop.org/xorg/driver/xf86-video-ati/-/archive/\
xf86-video-ati-$version/xf86-video-ati-xf86-video-ati-$version.tar.gz"
depends=""
group="x11.drivers"

setup(){
        cd $SOURCEDIR
        ./autogen.sh
        ./configure --prefix=/usr \
        --libdir=/usr/lib64/
}

build(){
        make
}

package(){
        make install DESTDIR=$DESTDIR
}
```

Not: Burada verilen derleme talimatı(script) **kly Paket Sistemi**'ni kullanarak paketi derler ve oluştur. Oluşan paket(**.kly uzantılı dosya**) **kly Paket Sistemi** kullanılarak siteme yüklenebilir. **kly Paket Sistemiyle Paket Yapma** konusunu okumak için tıklayınız.

xf86-video-dummy

Dummy video sürücüsüdür.

Paketi Derleme :

```
#!/usr/bin/env bash
name="xf86-video-dummy"
version="0.4.0"
description="X.Org driver for dummy cards"
source="https://gitlab.freedesktop.org/xorg/driver/xf86-video-dummy/-/archive/\
xf86-video-dummy-$version/xf86-video-dummy-xf86-video-dummy-$version.tar.gz"
depends=""
group="x11.drivers"

setup(){
    cd $SOURCEDIR
    ./autogen.sh
    ./configure --prefix=/usr \
        --libdir=/usr/lib64/
}

build(){
    make
}

package(){
    make install DESTDIR=$DESTDIR
}
```

Not: Burada verilen derleme talimatı(script) **kly Paket Sistemi**'ni kullanarak paketi derler ve oluştur. Oluşan paket(**.kly uzantılı dosya**) **kly Paket Sistemi** kullanılarak siteme yüklenebilir. **kly Paket Sistemiyle Paket Yapma** konusunu okumak için tıklayınız.

xf86-video-fbdev

framebuffer (fbdev) video sürücüsüdür.

Paketi Derleme :

```
#!/usr/bin/env bash
name="xf86-video-fbdev"
version="0.5.0"
description="video driver for framebuffer devic"
source="https://gitlab.freedesktop.org/xorg/driver/xf86-video-fbdev/-/archive/\
xf86-video-fbdev-$version/xf86-video-fbdev-xf86-video-fbdev-$version.tar.gz"
depends=""
group="x11.drivers"

setup(){
    cd $SOURCEDIR
    ./autogen.sh
    ./configure --prefix=/usr \
        --libdir=/usr/lib64/
}

build(){
    make
}

package(){
    make install DESTDIR=$DESTDIR
}
```

Not: Burada verilen derleme talimatı(script) **kly Paket Sistemi**'ni kullanarak paketi derler ve oluştur. Oluşan paket(**.kly uzantılı dosya**) **kly Paket Sistemi** kullanılarak siteme yüklenebilir. **kly Paket Sistemiyle Paket Yapma** konusunu okumak için tıklayınız.

xf86-video-intel

Intel grafik kartları için video sürücüsüdür.

Paketi Derleme :

```
#!/usr/bin/env bash
name="xf86-video-intel"
version="2.99.917"
description="X.Org driver for Intel cards"
source="https://salsa.debian.org/xorg-team/driver/xserver-xorg-video-intel/-/\
archive/xserver-xorg-video-intel-2_$version+git20210115-1/\
xserver-xorg-video-intel-xserver-xorg-video-intel-2_$version+git20210115-1.tar.gz"
depends=""
group=x11.drivers

setup(){
    cd $SOURCEDIR
    ./autogen.sh
    ./configure --prefix=/usr \
        --libdir=/usr/lib64/ \
        --with-default-dri=3
}

build(){
    make
}

package(){
    make install DESTDIR=$DESTDIR
}
```

Not: Burada verilen derleme talimatı(script) **kly Paket Sistemi**'ni kullanarak paketi derler ve oluştur. Oluşan paket(**.kly uzantılı dosya**) **kly Paket Sistemi** kullanılarak siteme yüklenebilir. **kly Paket Sistemiyle Paket Yapma** konusunu okumak için tıklayınız.

xf86-video-mga

Matrox grafik kartları için geliştirilmiş bir video sürücüsüdür.

Paketi Derleme :

```
#!/usr/bin/env bash
name="xf86-video-mga"
version="2.0.1"
description="Matrox video driver"
source="https://gitlab.freedesktop.org/xorg/driver/xf86-video-mga/-/archive/\
xf86-video-mga-$version/xf86-video-mga-xf86-video-mga-$version.tar.gz"
depends=""
group="x11.drivers"

setup(){
    cd $SOURCEDIR
    ./autogen.sh
    ./configure --prefix=/usr \
        --libdir=/usr/lib64/
}

build(){
    make
}

package(){
    make install DESTDIR=$DESTDIR
}
```

xf86-video-nouveau

NVIDIA grafik kartları için video sürücüsüdür.

Paketi Derleme :

```
#!/usr/bin/env bash
name="xf86-video-nouveau"
version="1.0.17"
description="Accelerated Open Source driver for nVidia cards"
source="https://gitlab.freedesktop.org/xorg/driver/xf86-video-nouveau/-/archive/\
xf86-video-nouveau-$version/xf86-video-nouveau-xf86-video-nouveau-$version.tar.gz"
depends=""
group="x11.drivers"

setup(){
        cp -prfv $PACKAGEDIR/files/* $SOURCEDIR/
        cd $SOURCEDIR
        patch -Np1 < ./xorg-server-21.1.diff
        autoreconf -fvi
        ./configure --prefix=/usr \
        --libdir=/usr/lib64/
}

build(){
        make
}

package(){
        make install DESTDIR=$DESTDIR
}
```

Ek dosyaları indirmek için tıklayınız..

Not: Burada verilen derleme talimatı(script) **kly Paket Sistemi**'ni kullanarak paketi derler ve oluştur. Oluşan paket(**.kly uzantılı dosya**) **kly Paket Sistemi** kullanılarak siteme yüklenebilir. **kly Paket Sistemiyle Paket Yapma** konusunu okumak için tıklayınız.

xf86-video-r128

ATI R128 serisi grafik kartları için video sürücüsüdür.

Paketi Derleme :

```
#!/usr/bin/env bash
name="xf86-video-r128"
version="6.12.1"
description="ATI Rage128 video drive"
source="https://gitlab.freedesktop.org/xorg/driver/xf86-video-r128/-/archive/\
xf86-video-r128-$version/xf86-video-r128-xf86-video-r128-$version.tar.gz"
depends=""
group="x11.drivers"

setup(){
    cd $SOURCEDIR
    ./autogen.sh
    ./configure --prefix=/usr \
        --libdir=/usr/lib64/
}

build(){
    make
}

package(){
    make install DESTDIR=$DESTDIR
}
```

Not: Burada verilen derleme talimatı(script) **kly Paket Sistemi**'ni kullanarak paketi derler ve oluştur. Oluşan paket(**.kly uzantılı dosya**) **kly Paket Sistemi** kullanılarak siteme yüklenebilir. **kly Paket Sistemiyle Paket Yapma** konusunu okumak için tıklayınız.

xf86-video-siliconmotion

Silicon Motion grafik kartları için video sürücüsüdür.

Paketi Derleme :

```
#!/usr/bin/env bash
name="xf86-video-siliconmotion"
version="1.7.9"
description="Silicon Motion video driver"
source="https://gitlab.freedesktop.org/xorg/driver/xf86-video-siliconmotion/-\
/archive/xf86-video-siliconmotion-$version/\
xf86-video-siliconmotion-xf86-video-siliconmotion-$version.tar.gz"
depends=""
group="x11.drivers"

setup(){
    cd $SOURCEDIR
    ./autogen.sh
    ./configure --prefix=/usr \
        --libdir=/usr/lib64/
}

build(){
    make
}

package(){
    make install DESTDIR=$DESTDIR
}
```

Not: Burada verilen derleme talimatı(script) **kly Paket Sistemi**'ni kullanarak paketi derler ve oluştur. Oluşan paket(**.kly uzantılı dosya**) **kly Paket Sistemi** kullanılarak siteme yüklenebilir. **kly Paket Sistemiyle Paket Yapma** konusunu okumak için tıklayınız.

xf86-video-vboxvideo

VirtualBox sanal makinesi video sürücüsüdür.

Paketi Derleme :

```
#!/usr/bin/env bash
name="xf86-video-vboxvideo"
version="1.0.0"
description="VirtualBox guest video driver"
source="https://gitlab.freedesktop.org/xorg/driver/xf86-video-vbox/-/archive/\
xf86-video-vboxvideo-$version/\
xf86-video-vbox-xf86-video-vboxvideo-$version.tar.gz"
depends=""
group="x11.drivers"

setup(){
    cd $SOURCEDIR
    ./autogen.sh
    ./configure --prefix=/usr \
        --libdir=/usr/lib64/
}

build(){
    make
}

package(){
    make install DESTDIR=$DESTDIR
}
```

Not: Burada verilen derleme talimatı(script) **kly Paket Sistemi**'ni kullanarak paketi derler ve oluştur. Oluşan paket(**.kly uzantılı dosya**) **kly Paket Sistemi** kullanılarak siteme yüklenebilir. **kly Paket Sistemiyle Paket Yapma** konusunu okumak için tıklayınız.

xf86-video-vesa

Genel amaçlı bir video sürücüsü.

Paketi Derleme :

```
#!/usr/bin/env bash
name="xf86-video-vesa"
version="2.6.0"
description="Generic VESA video driver"
source="https://gitlab.freedesktop.org/xorg/driver/xf86-video-vesa/-/archive/\
xf86-video-vesa-$version/xf86-video-vesa-xf86-video-vesa-$version.tar.gz"
depends=""
group="x11.drivers"

setup(){
    cd $SOURCEDIR
    ./autogen.sh
    ./configure --prefix=/usr \
        --libdir=/usr/lib64/
}

build(){
    make
}

package(){
    make install DESTDIR=$DESTDIR
}
```

Not: Burada verilen derleme talimatı(script) **kly Paket Sistemi**'ni kullanarak paketi derler ve oluştur. Oluşan paket(**.kly uzantılı dosya**) **kly Paket Sistemi** kullanılarak siteme yüklenebilir. **kly Paket Sistemiyle Paket Yapma** konusunu okumak için tıklayınız.

xf86-video-vmware

VMware sanal makineleri için video sürücüsüdür.

Paketi Derleme :

```
#!/usr/bin/env bash
name="xf86-video-vmware"
version="13.4.0"
description="VMware SVGA video driver"
source="https://gitlab.freedesktop.org/xorg/driver/xf86-video-vmware/-/archive/\
xf86-video-vmware-$version/xf86-video-vmware-xf86-video-vmware-$version.tar.gz"
depends=""
group="x11.drivers"

setup(){
    cd $SOURCEDIR
    ./autogen.sh
    ./configure --prefix=/usr \
        --libdir=/usr/lib64/
}

build(){
    make
}

package(){
    make install DESTDIR=$DESTDIR
}
```

xkbcomp

Klavye yapılandırma için kullanılan bir komut satırı aracıdır.

Paketi Derleme :

```
#!/usr/bin/env bash
name="xkbcomp"
version="1.4.6"
description="XKB keyboard description compiler"
source="https://gitlab.freedesktop.org/xorg/app/xkbcomp/-/archive/\
xkbcomp-$version/xkbcomp-xkbcomp-$version.tar.gz"
depends="libxkbfile,libX11"
group="x11.apps"

setup(){
    cd $SOURCEDIR
    autoreconf -fvi
    ./configure --prefix=/usr \
        --libdir=/usr/lib64/
}

build(){
    make
}

package(){
    make install DESTDIR=$DESTDIR
}
```

Not: Burada verilen derleme talimatı(script) **kly Paket Sistemi**'ni kullanarak paketi derler ve oluştur. Oluşan paket(**.kly uzantılı dosya**) **kly Paket Sistemi** kullanılarak siteme yüklenebilir. **kly Paket Sistemiyle Paket Yapma** konusunu okumak için tıklayınız.

libxkbfile

Klavye yapılandırması için kullanılan kütüphanedir.

Paketi Derleme :

```
#!/usr/bin/env bash
name="libxkbfile"
version="1.1.2"
description="X.Org xkbfile library"
source="https://www.x.org/archive/individual/lib/libxkbfile-$version.tar.gz"
depends="libX11,xorgproto"
group="x11.libs"

setup(){
    $SOURCEDIR/configure --prefix=/usr \
        --libdir=/usr/lib64/
}

build(){
    make
}

package(){
    make install DESTDIR=$DESTDIR
}
```

Not: Burada verilen derleme talimatı(script) **kly Paket Sistemi**'ni kullanarak paketi derler ve oluştur. Oluşan paket(**.kly uzantılı dosya**) **kly Paket Sistemi** kullanılarak siteme yüklenebilir. **kly Paket Sistemiyle Paket Yapma** konusunu okumak için tıklayınız.

libglvnd

OpenGL (GL), grafik sürücü yönetimini için kütüphanedir.

Paketi Derleme :

```
#!/usr/bin/env bash
name="libglvnd"
version="1.7.0"
description="The GL Vendor-Neutral Dispatch library"
source="https://gitlab.freedesktop.org/glvnd/libglvnd/-/archive/\
v$version/libglvnd-v$version.tar.gz"
depends="libXext"
group="media.libs"

setup(){
    cd $SOURCEDIR
    meson setup $BUILDDIR --prefix=/usr \
        --libdir=/usr/lib64/
}

build(){
    ninja -C $BUILDDIR
}

package(){
    DESTDIR=$DESTDIR ninja -C $BUILDDIR install
}
```

Not: Burada verilen derleme talimatı(script) **kly Paket Sistemi**'ni kullanarak paketi derler ve oluştur. Oluşan paket(**.kly uzantılı dosya**) **kly Paket Sistemi** kullanılarak siteme yüklenebilir. **kly Paket Sistemiyle Paket Yapma** konusunu okumak için tıklayınız.

startup-notification

Bilgilendirme(uygulama mesajları) için kullanılan bir pakettir.

Paketi Derleme :

```
#!/usr/bin/env bash
name="startup-notification"
version="0.12"
description="Application startup notification and feedback library "
source="http://www.freedesktop.org/software/startup-notification/\
releases/startup-notification-${version}.tar.gz"
depends="libX11,xcb-util"
group="x11.libs"

setup(){
    $SOURCEDIR/configure --prefix=/usr \
        --libdir=/usr/lib64/
}

build(){
    make
}

package(){
    make install DESTDIR=$DESTDIR
}
```

Not: Burada verilen derleme talimatı(script) **kly Paket Sistemi**'ni kullanarak paketi derler ve oluştur. Oluşan paket(**.kly uzantılı dosya**) **kly Paket Sistemi** kullanılarak siteme yüklenebilir. **kly Paket Sistemiyle Paket Yapma** konusunu okumak için tıklayınız.

xkeyboard-config

X11 sistemlerinde klavye yapılandırması ve klavye seçeneklerinin yönetilmesini sağlar.

Paketi Derleme :

```
#!/usr/bin/env bash
name="xkeyboard-config"
version="2.40"
description="X keyboard configuration database"
source="https://gitlab.freedesktop.org/xkeyboard-config/xkeyboard-config/-/\
archive/xkeyboard-config-$version/xkeyboard-config-xkeyboard-config-$version.tar.gz"
depends=""
group="x11.misc"

setup(){
    cd $SOURCEDIR
    meson setup $BUILDDIR --prefix=/usr \
        --libdir=/usr/lib64/
}

build(){
    ninja -C $BUILDDIR
}

package(){
    DESTDIR=$DESTDIR ninja -C $BUILDDIR install
}
```

Not: Burada verilen derleme talimatı(script) **kly Paket Sistemi**'ni kullanarak paketi derler ve oluştur. Oluşan paket(**.kly uzantılı dosya**) **kly Paket Sistemi** kullanılarak siteme yüklenebilir. **kly Paket Sistemiyle Paket Yapma** konusunu okumak için tıklayınız.

libinput

Giriş aygıtlarını yönetmek için bir kütüphanedir.

Paketi Derleme :

```
#!/usr/bin/env bash
name="libinput"
version="1.25.0"
url="https://gitlab.freedesktop.org/libinput/libinput"
description="Input device management and event handling library"
source="https://gitlab.freedesktop.org/libinput/libinput/-/archive/\
$version/libinput-$version.tar.gz"
depends="eudev,mtdev,libevdev"

setup(){
    cd $SOURCEDIR
    meson setup $BUILDDIR --prefix=/usr \
        --libdir=/usr/lib64/ \
        -Dudev-dir=/lib64/udev \
        -Dlibwacom=true  \
        -Ddebug-gui=false \
        -Dtests=false
}

build(){
    ninja -C $BUILDDIR
}

package(){
       DESTDIR=$DESTDIR  ninja -C  $BUILDDIR install
}
```

Not: Burada verilen derleme talimatı(script) **kly Paket Sistemi**'ni kullanarak paketi derler ve oluştur. Oluşan paket(**.kly uzantılı dosya**) **kly Paket Sistemi** kullanılarak siteme yüklenebilir. **kly Paket Sistemiyle Paket Yapma** konusunu okumak için tıklayınız.

mtdev

multitouch (çoklu dokunma) aygıtlarını için sürücüdür.

Paketi Derleme :

```
#!/usr/bin/env bash
name="mtdev"
version="1.1.6"
url="https://bitmath.org/code/mtdev/"
description="Multitouch Protocol Translation Library"
source="https://bitmath.org/code/mtdev/mtdev-$version.tar.gz"
group="dev.libs"

setup(){
    $SOURCEDIR/configure --prefix=/usr \
        --libdir=/usr/lib64/
}

build(){
    make
}

package(){
    make install DESTDIR=$DESTDIR
}
```

Not: Burada verilen derleme talimatı(script) **kly Paket Sistemi**'ni kullanarak paketi derler ve oluştur. Oluşan paket(**.kly uzantılı dosya**) **kly Paket Sistemi** kullanılarak siteme yüklenebilir. **kly Paket Sistemiyle Paket Yapma** konusunu okumak için tıklayınız.

libevdev

Giriş aygıtlarını yönetmek için kullanılan bir C kütüphanesidir.

Paketi Derleme :

Debian'da paketi derlemek için aşağıdaki paketlerin kurulu olması gerekir.

```
sudo apt install doxygen
```

```
#!/usr/bin/env bash
name="libevdev"
version="1.13.1"
description="Wrapper library for evdev devices"
source="https://gitlab.freedesktop.org/libevdev/libevdev/-/archive/\
libevdev-$version/libevdev-libevdev-$version.tar.gz"
depends=""
builddepend="doxygen,meson,python3"
group="sys.libs"

setup(){
    cd $SOURCEDIR
    meson setup $BUILDDIR \
            --prefix=/usr \
        --libdir=/usr/lib \
            -Db_lto=true \
            -Dtests=disabled \
            -Ddocumentation=enabled \
            -Dcoverity=false
}

build(){
    ninja -C $BUILDDIR
}

package(){
    DESTDIR=$DESTDIR ninja -C $BUILDDIR install
}
```

Not: Burada verilen derleme talimatı(script) **kly Paket Sistemi**'ni kullanarak paketi derler ve oluştur. Oluşan paket(**.kly uzantılı dosya**) **kly Paket Sistemi** kullanılarak siteme yüklenebilir. **kly Paket Sistemiyle Paket Yapma** konusunu okumak için tıklayınız.

libwacom

Dokunmatik grafik aygıtları için geliştirilmiş bir kütüphanedir.

Paketi Derleme :

```
#!/usr/bin/env bash
name="libwacom"
version="2.10.0"
url="https://github.com/linuxwacom/libwacom/wiki"
description="Library to help implement Wacom tablet settings"
source="https://github.com/linuxwacom/libwacom/releases/download/\
libwacom-${version}/libwacom-${version}.tar.xz"
depends="eudev"
builddepend="meson"
group="dev.libs"
setup(){
        cd $SOURCEDIR
        meson setup $BUILDDIR \
        --prefix=/usr \
        --buildtype=release \
        -Dtests=disabled
}
build(){
        ninja -C $BUILDDIR $jobs
}

package(){
        DESTDIR=$DESTDIR ninja -C $BUILDDIR install $jobs
        install -D -m644 COPYING "${DESTDIR}/usr/share/licenses/${name}/LICENSE"
}
```

Not: Burada verilen derleme talimatı(script) **kly Paket Sistemi**'ni kullanarak paketi derler ve oluştur. Oluşan paket(**.kly uzantılı dosya**) **kly Paket Sistemi** kullanılarak siteme yüklenebilir. **kly Paket Sistemiyle Paket Yapma** konusunu okumak için tıklayınız.

libgudev

Donanım aygıtlarını yönetmek için kullanılan bir C kütüphanesidir.

Paketi Derleme :

Debian'da paketi derlemek için aşağıdaki paketlerin kurulu olması gerekir.

```
sudo apt install gobject-introspection libumockdev-dev valac libgirepository1.0-dev
```

```
#!/usr/bin/env bash
name="libgudev"
version="238"
url="https://example.org"
description="GObject bindings for libudev"
source="https://gitlab.gnome.org/GNOME/libgudev/-/archive/$version/\
libgudev-$version.tar.gz"
depends="glib,eudev"
builddepend=""
group="dev.libs"

setup(){
cd $SOURCEDIR
    meson setup $BUILDDIR --prefix=/usr \
        --libdir=/usr/lib64/ \
        -Dintrospection=enabled \
            -Dvapi=enabled \
            -Dgtk_doc=false
}

build(){
    meson compile -C $BUILDDIR
}

package(){
    DESTDIR="$DESTDIR" meson install -C $BUILDDIR
}
```

Not: Burada verilen derleme talimatı(script) **kly Paket Sistemi**'ni kullanarak paketi derler ve oluştur. Oluşan paket(**.kly uzantılı dosya**) **kly Paket Sistemi** kullanılarak siteme yüklenebilir. **kly Paket Sistemiyle Paket Yapma** konusunu okumak için tıklayınız.

libffi

Programlama kütüphanesidir.

Paketi Derleme :

```
#!/bin/bash
name="libffi"
version="3.4.6"
description="A portable foreign-function interface library. "
source="https://github.com/libffi/libffi/releases/download/v${version}/\
libffi-${version}.tar.gz"
depends=""
builddepend=""
group="dev.libs"

setup(){
    cd $SOURCEDIR
    ./configure --prefix=/usr \
        --libdir=/usr/lib64/ \
        --enable-pax_emutramp \
            --enable-portable-binary \
            --disable-exec-static-tramp
}

build(){
    make
}

package(){
    make install DESTDIR=$DESTDIR
}
```

Not: Burada verilen derleme talimatı(script) **kly Paket Sistemi**'ni kullanarak paketi derler ve oluştur. Oluşan paket(**.kly uzantılı dosya**) **kly Paket Sistemi** kullanılarak siteme yüklenebilir. **kly Paket Sistemiyle Paket Yapma** konusunu okumak için tıklayınız.

xinit

X11 için kullanılan, X sunucusunu başlatıp ardından belirtilen istemciyi çalıştıran bir komuttur. Grafik arayüzü olmayan sistemlerde veya özel oturumlarda kullanılır ve çalıştırılacak istemciler genellikle *~/.xinitrc* ile belirlenir.

Paketi Derleme :

```
#!/usr/bin/env bash
name="xinit"
version="1.4.2"
description="X Window System initializer xinit"
source="https://gitlab.freedesktop.org/xorg/app/xinit/-/archive/\
xinit-$version/xinit-xinit-1.4.2.tar.gz"
depends="util-macros,xorgproto,libX11,xorg-server,xterm,xhost,xrdb"
group="x11.apps"


setup(){
        cp -prfv $PACKAGEDIR/files $SOURCEDIR
        cd $SOURCEDIR

        autoreconf -vfi
        ./configure --prefix=/usr \
            --libdir=/usr/lib64/ \
            -sysconfdir=/etc
}

build(){
        make
}

package(){
        make install DESTDIR=$DESTDIR
        install -m755 -D $SOURCEDIR/files/Xwrapper.config "$DESTDIR"/etc/X11/Xwrapper.config
        install -m755 -D $SOURCEDIR/files/xinitrc "$DESTDIR"/etc/X11/xinit/xinitrc
        install -m755 -D $SOURCEDIR/files/Xsession "$DESTDIR"/etc/X11/xinit/Xsession
        install -m755 $SOURCEDIR/files/xserverrc "$DESTDIR"/etc/X11/xinit/xserverrc
        install -m755 -D $SOURCEDIR/files/xinit.sysconf "$DESTDIR"/etc/sysconf.d/xinit
        install -m755 -D $SOURCEDIR/files/xinit.initd "$DESTDIR"/etc/init.d/xinit
        install -m755 -D $SOURCEDIR/files/xinit.confd "$DESTDIR"/etc/conf.d/xinit
        install $SOURCEDIR/files/Xresources "$DESTDIR"/etc/X11/Xresources
        # allow local connections config
        mkdir -p "$DESTDIR"/etc/X11/xinit/xinitrc.d
        echo "#!/bin/sh" > "$DESTDIR"/etc/X11/xinit/xinitrc.d/10-local.sh
        echo "xhost +local:" >> "$DESTDIR"/etc/X11/xinit/xinitrc.d/10-local.sh
        chmod 754 "$DESTDIR"/etc/X11/xinit/xinitrc.d/10-local.sh
}
```

Ek dosyaları indirmek için tıklayınız..

Not: Burada verilen derleme talimatı(script) **kly Paket Sistemi**'ni kullanarak paketi derler ve oluştur. Oluşan paket(**.kly uzantılı dosya**) **kly Paket Sistemi** kullanılarak siteme yüklenebilir. **kly Paket Sistemiyle Paket Yapma** konusunu okumak için tıklayınız.

xcalc

X11 üzerinde çalışan hesap makinesi uygulamasıdır.

Paketi Derleme :

Debian'da paketi derlemek için aşağıdaki paketlerin kurulu olması gerekir.

```
sudo apt install libxaw7-dev libxt-dev  libsm-dev \
libxau-dev libxext-dev libxi-dev libxmu-dev libxt-dev
```

```
#!/usr/bin/env bash
name="xcalc"
version="1.1.1"
description="scientific calculator for X"
source="https://gitlab.freedesktop.org/xorg/app/xcalc/-/archive/\
xcalc-$version/xcalc-xcalc-$version.tar.gz"
depends="libXaw,libICE,libSM,libXau,libXau,libXext,libXi,libXmu,libXrender,libXt,libxcb"
group="x11.apps"

setup(){
    cd $SOURCEDIR
    autoreconf -fvi
    ./configure --prefix=/usr \
        --libdir=/usr/lib64/
}

build(){
    make
}

package(){
    make install DESTDIR=$DESTDIR
}
```

Not: Burada verilen derleme talimatı(script) **kly Paket Sistemi**'ni kullanarak paketi derler ve oluştur. Oluşan paket(**.kly uzantılı dosya**) **kly Paket Sistemi** kullanılarak siteme yüklenebilir. **kly Paket Sistemiyle Paket Yapma** konusunu okumak için tıklayınız.

libXi

X uygulamarında giriş aygıtlarına erişmesini sağlar.

Paketi Derleme :

Debian'da paketi derlemek için aşağıdaki paketlerin kurulu olması gerekir.

```
sudo apt install libxfixes-dev
```

```
#!/usr/bin/env bash
name="libXi"
version="1.8"
description="X.Org Xi library"
source="https://www.x.org/archive/individual/lib/libXi-$version.tar.gz"
depends="libXfixes"
group="x11.libs"

setup(){
    $SOURCEDIR/configure --prefix=/usr \
        --libdir=/usr/lib64/
}

build(){
    make
}

package(){
    make install DESTDIR=$DESTDIR
}
```

Not: Burada verilen derleme talimatı(script) **kly Paket Sistemi**'ni kullanarak paketi derler ve oluştur. Oluşan paket(**.kly uzantılı dosya**) **kly Paket Sistemi** kullanılarak siteme yüklenebilir. **kly Paket Sistemiyle Paket Yapma** konusunu okumak için tıklayınız.

openbox

X11 için bir pencere yöneticisidir.

Paketi Derleme :

Debian'da paketi derlemek için aşağıdaki paketlerin kurulu olması gerekir.

```
sudo apt install libpango1.0-dev
```

```
#!/usr/bin/env bash
name="openbox"
version="3.6.1"
description="Highly configurable and lightweight X11 window manager"
source="http://openbox.org/dist/openbox/openbox-$version.tar.xz"
depends="libSM,libxml2,pango,startup-notification,libXrandr,librsvg,libXinerama"
group="x11.wm"

setup(){
   $SOURCEDIR/configure --prefix=/usr \
        --sysconfdir=/etc \
        --libdir=/usr/lib64/
}

build(){
    make
}

package(){
    make install DESTDIR=$DESTDIR
}
```

Not: Burada verilen derleme talimatı(script) **kly Paket Sistemi**'ni kullanarak paketi derler ve oluştur. Oluşan paket(**.kly uzantılı dosya**) **kly Paket Sistemi** kullanılarak siteme yüklenebilir. **kly Paket Sistemiyle Paket Yapma** konusunu okumak için tıklayınız.

libXcursor

Fare imleci (cursor) yönetimini geliştiren bir kütüphanedir.

Paketi Derleme :

```
#!/usr/bin/env bash
name="libXcursor"
version="1.2.1"
description="X.Org Xcursor library"
source="https://www.x.org/archive/individual/lib/libXcursor-$version.tar.xz"
depends="libXrender,libXfixes,libX11"
group="x11.libs"

setup(){
    $SOURCEDIR/configure --prefix=/usr \
        --libdir=/usr/lib64/
}

build(){
    make
}

package(){
    make install DESTDIR=$DESTDIR
}
```

Not: Burada verilen derleme talimatı(script) **kly Paket Sistemi**'ni kullanarak paketi derler ve oluştur. Oluşan paket(**.kly uzantılı dosya**) **kly Paket Sistemi** kullanılarak siteme yüklenebilir. **kly Paket Sistemiyle Paket Yapma** konusunu okumak için tıklayınız.

libXfixes

Xfixes uzantısı için gerekli kütüphanedir.

Paketi Derleme :

```
#!/usr/bin/env bash
name="libXfixes"
version="6.0.0"
description="X.Org Xfixes library"
source="https://www.x.org/archive/individual/lib/libXfixes-$version.tar.gz"
depends=""
group="x11.libs"

setup(){
    $SOURCEDIR/configure --prefix=/usr \
        --libdir=/usr/lib64/
}

build(){
    make
}

package(){
    make install DESTDIR=$DESTDIR
}
```

Not: Burada verilen derleme talimatı(script) **kly Paket Sistemi**'ni kullanarak paketi derler ve oluştur. Oluşan paket(**.kly uzantılı dosya**) **kly Paket Sistemi** kullanılarak siteme yüklenebilir. **kly Paket Sistemiyle Paket Yapma** konusunu okumak için tıklayınız.

pango

Metin işleme kütüphanesidir.

Paketi Derleme :

```
#!/usr/bin/env bash
name="pango"
version="1.51.0"
description="Layout and render international text"
source="https://download.gnome.org/sources/pango/1.51/pango-$version.tar.xz"
depends="fribidi,libthai,harfbuzz,cairo,libXft"
makedepend="gobject-introspection"
group="x11.libs"

setup(){
cd $SOURCEDIR
    meson setup $BUILDDIR --prefix=/usr \
        --libdir=/usr/lib64/ \
        -Dintrospection=enabled \
        -Dxft=enabled \
        -Dcairo=enabled \
        -Dlibthai=enabled \
        -Dfreetype=enabled \
        -Dgtk_doc=false
}

build(){
    ninja -C $BUILDDIR
}

package(){
    DESTDIR=$DESTDIR ninja -C $BUILDDIR install
}
```

Not: Burada verilen derleme talimatı(script) **kly Paket Sistemi**'ni kullanarak paketi derler ve oluştur. Oluşan paket(**.kly uzantılı dosya**) **kly Paket Sistemi** kullanılarak siteme yüklenebilir. **kly Paket Sistemiyle Paket Yapma** konusunu okumak için tıklayınız.

libXrandr

Ekran çözünürlüğü için gerekli kütüphanedir.

Paketi Derleme :

```
#!/usr/bin/env bash
name="libXrandr"
version="1.5.3"
description="X.Org Xrandr library"
source="https://www.x.org/archive/individual/lib/libXrandr-$version.tar.xz"
depends="libXrender"
group="x11.libs"

setup(){
    $SOURCEDIR/configure --prefix=/usr \
        --libdir=/usr/lib64/
}

build(){
    make
}

package(){
    make install DESTDIR=$DESTDIR
}
```

Not: Burada verilen derleme talimatı(script) **kly Paket Sistemi**'ni kullanarak paketi derler ve oluştur. Oluşan paket(**.kly uzantılı dosya**) **kly Paket Sistemi** kullanılarak siteme yüklenebilir. **kly Paket Sistemiyle Paket Yapma** konusunu okumak için tıklayınız.

fribidi

Unicode metinlerde metin işleme kütüphanesidir.

Paketi Derleme :

```
#!/usr/bin/env bash
name="fribidi"
version="1.0.12"
description="A free implementation of the unicode bidirectional algorithm"
source="https://github.com/fribidi/fribidi/releases/download/\
v$version/fribidi-$version.tar.xz"
depends=""
builddepend="meson"
group="dev.libs"

setup(){
    cd $SOURCEDIR
    ./configure --prefix=/usr \
        --libdir=/usr/lib64/
}

build(){
    make
}

package(){
    make install DESTDIR=$DESTDIR
}
```

Not: Burada verilen derleme talimatı(script) **kly Paket Sistemi**'ni kullanarak paketi derler ve oluştur. Oluşan paket(**.kly uzantılı dosya**) **kly Paket Sistemi** kullanılarak siteme yüklenebilir. **kly Paket Sistemiyle Paket Yapma** konusunu okumak için tıklayınız.

xcb-util

X11 için yardımcı kütüphanesidir.

Paketi Derleme :

```
#!/usr/bin/env bash
name="xcb-util"
version="0.4.0"
description="X C-language Bindings sample implementations"
source="https://www.x.org/releases/individual/xcb/xcb-util-${version}.tar.gz"
depends="libxcb,util-macros,xorgproto"
group="x11.libs"

setup(){
    $SOURCEDIR/configure --prefix=/usr \
            --libdir=/usr/lib64 \
            --enable-shared \
            --enable-static \
            --disable-devel-docs \
            --without-doxygen
}

build(){
    make
}

package(){
    make install DESTDIR=$DESTDIR
}
```

Not: Burada verilen derleme talimatı(script) **kly Paket Sistemi**'ni kullanarak paketi derler ve oluştur. Oluşan paket(**.kly uzantılı dosya**) **kly Paket Sistemi** kullanılarak siteme yüklenebilir. **kly Paket Sistemiyle Paket Yapma** konusunu okumak için tıklayınız.

libthai

Tayland yazı sistemini için gerekli C kütüphanesidir.

Paketi Derleme :

```
#!/usr/bin/env bash
name="libthai"
version="0.1.29"
description="Package libthai"
source="https://github.com/tlwg/libthai/releases/download/\
v$version/libthai-$version.tar.xz"
depends="libdatrie"
group="x11.libs"

setup(){
    $SOURCEDIR/configure --prefix=/usr \
        --libdir=/usr/lib64 --disable-dict
}

build(){
    make
}

package(){
    make install DESTDIR=$DESTDIR
}
```

Not: Burada verilen derleme talimatı(script) **kly Paket Sistemi**'ni kullanarak paketi derler ve oluştur. Oluşan paket(**.kly uzantılı dosya**) **kly Paket Sistemi** kullanılarak siteme yüklenebilir. **kly Paket Sistemiyle Paket Yapma** konusunu okumak için tıklayınız.

libdatrie

Veri yapısı için kullanılan bir C kütüphanesidir.

Paketi Derleme :

```
#!/usr/bin/env bash
name="libdatrie"
version="0.2.13"
description="Implementation of double-array structure for\
 representing trie, as proposed by Junichi Aoe."
source="https://github.com/tlwg/libdatrie/releases/download/\
v$version/libdatrie-$version.tar.xz"
depends=""
group="x11.libs"

setup(){
    $SOURCEDIR/configure --prefix=/usr \
        --libdir=/usr/lib64/
}

build(){
    make
}

package(){
    make install DESTDIR=$DESTDIR
}
```

Not: Burada verilen derleme talimatı(script) **kly Paket Sistemi**'ni kullanarak paketi derler ve oluştur. Oluşan paket(**.kly uzantılı dosya**) **kly Paket Sistemi** kullanılarak siteme yüklenebilir. **kly Paket Sistemiyle Paket Yapma** konusunu okumak için tıklayınız.

dbus

D-Bus, süreçlerin birbiriyle haberleşmesini sağlayan mesajlaşma sistemidir.

Paketi Derleme :

```
#-----------------------------------------------------------------------------------------
#!/usr/bin/env bash
name="dbus"
version="1.15.2"
description="A message bus system, a simple way for applications to talk to each other"
source="https://dbus.freedesktop.org/releases/dbus/dbus-$version.tar.xz"
depends="audit,expat,libcap-ng,elogind,libX11,libunwind"
group="sys.apps"

setup(){
    cp -r ${dizin}/${paket}/files/ /tmp/kly/build/
    cd $SOURCEDIR
    ./configure --prefix=/usr --libdir=/usr/lib64/ --sysconfdir=/etc  --localstatedir=/var \
        --runstatedir=/run --disable-doxygen-docs --disable-xml-docs --disable-static  \
        --disable-systemd --with-system-pid-file=/run/dbus/dbus.pid --with-x \
        --with-systemduserunitdir=no --with-systemdsystemunitdir=no  \
        --with-system-socket=/run/dbus/system_bus_socket
}

build(){
    make
}

package(){
    make install DESTDIR=$DESTDIR
    mkdir -p "$DESTDIR"/etc/init.d "$DESTDIR"/etc/local.d "$DESTDIR"/etc/X11/xinit/xinitrc.d/
    install $SOURCEDIR/files/dbus.initd "$DESTDIR"/etc/init.d/dbus
    install $SOURCEDIR/files/dbus.xinit "$DESTDIR"/etc/X11/xinit/xinitrc.d/30-dbus-launch.sh

    for level in boot default nonetwork shutdown sysinit ; do
    mkdir -p ${DESTDIR}/etc/runlevels/$level
    done
    cd ${DESTDIR}/etc/runlevels/default
    ln -s ../../init.d/dbus dbus
}
```

Ek dosyaları indirmek için tıklayınız..

Not: Burada verilen derleme talimatı(script) **kly Paket Sistemi**'ni kullanarak paketi derler ve oluştur. Oluşan paket(**.kly uzantılı dosya**) **kly Paket Sistemi** kullanılarak siteme yüklenebilir. **kly Paket Sistemiyle Paket Yapma** konusunu okumak için tıklayınız.

elogind

Linux'ta oturum (session) yönetimi sağlayan pakettir.

Paketi Derleme :

```
#-----------------------------------------------------------------------------------------------------------------
#!/usr/bin/env bash
name="elogind"
version="252.9"
description="Elogind is the systemd projects logind, extracted to a standalone package"
source="https://github.com/elogind/elogind/archive/refs/tags/v$version.tar.gz"
depends="acl,attr,audit,libcap-ng,libcap,dbus,pam,py3-jinja"
group="sys.auth"

setup(){
    cd $SOURCEDIR
    cp -prfv $PACKAGEDIR/files /tmp/kly/build/
    meson setup $BUILDDIR --prefix=/usr --libdir=/lib64/ -Drootlibdir=/lib64 -Dudevrulesdir=/lib64/udev/rules.d \
   -Dcgroup-controller=elogind -Dhalt-path=/sbin/halt -Dreboot-path=/sbin/reboot -Dpoweroff-path=/sbin/poweroff \
   -Drootlibexecdir=/usr/libexec/elogind -Ddefault-hierarchy=hybrid -Ddefault-kill-user-processes=true \
   -Dpam=true -Dpamconfdir=/etc/pam.d -Dpamlibdir=/usr/lib64/security -Dselinux=false -Daudit=true \
   -Defi=false -Dpolkit=true
}

build(){
    ninja -C $BUILDDIR
}

package(){
    cd $DESTDIR
    mkdir -p $DESTDIR/lib64
    ln -s lib64 lib
    mkdir -p "$DESTDIR"/usr/lib64/pkgconfig/
    DESTDIR=$DESTDIR ninja -C $BUILDDIR install
    # Claim compatiblity with systemd and systemd-logind (thanks Alpine linux)
    ln -s libelogind.pc "$DESTDIR"/lib64/pkgconfig/libsystemd.pc
    ln -s libelogind.pc "$DESTDIR"/lib64/pkgconfig/libsystemd-login.pc
    ln -s elogind "$DESTDIR"/usr/include/systemd
    # Extra compatiblity support
    ln -s libelogind.so.0 "$DESTDIR"/lib64/libsystemd.so.0
    ln -s libelogind.so.0 "$DESTDIR"/lib64/libsystemd.so.0.35.0
    ln -s pam_elogind.so "$DESTDIR"/usr/lib64/security/pam_systemd.so
    install ../files/systemd.pc "$DESTDIR"/lib64/pkgconfig/
    # Install headers from elogind
    install -Dm644 $SOURCEDIR/src/systemd/sd-id128.h "$DESTDIR"/usr/include/sd-id128.h
    install -Dm644 $SOURCEDIR/src/systemd/_sd-common.h "$DESTDIR"/usr/include/_sd-common.h
    # openrc service
    mkdir -p "${DESTDIR}"/etc/init.d
    install ../files/elogind.initd "${DESTDIR}"/etc/init.d/elogind
    # shadow system-auth.d file
    mkdir -p "${DESTDIR}"/etc/pam.d/system-auth.d/
    echo "session    include    elogind-user" > "${DESTDIR}"/etc/pam.d/system-auth.d/99-elogind
    # ld.so.conf
    mkdir -p "${DESTDIR}"/etc/ld.so.conf.d/
    echo "/lib64/elogind" > "${DESTDIR}"/etc/ld.so.conf.d/elogind.conf
    cd "$DESTDIR"/usr/lib64/pkgconfig/
    ln -s ../../../lib64/pkgconfig/libelogind.pc libelogind.pc
}
```

Ek dosyaları indirmek için tıklayınız..

Not: Burada verilen derleme talimatı(script) **kly Paket Sistemi**'ni kullanarak paketi derler ve oluştur. Oluşan paket(**.kly uzantılı dosya**) **kly Paket Sistemi** kullanılarak siteme yüklenebilir. **kly Paket Sistemiyle Paket Yapma** konusunu okumak için tıklayınız.

libunwind

Bir programın çağrı yığınını yönetmek için kullanılan bir kütüphanedir.

Paketi Derleme :

```
#!/usr/bin/env bash
    name="libunwind"
    version="1.8.1"
    description="Portable and efficient API to determine the call-chain of a program"
    source="https://github.com/libunwind/libunwind/archive/refs/tags/v$version.tar.gz"
    depends=""
    builddepend="autoconf,automake,libtool"
    group="sys.libs"

    setup(){
            cd $SOURCEDIR
            autoreconf -fvi
            ./configure --prefix=/usr \
            --libdir=/usr/lib64/ \
            --sysconfdir=/etc \
            --localstatedir=/var


    }

    build(){
            make
    }

    package(){
            make install DESTDIR=$DESTDIR
    }
```

Not: Burada verilen derleme talimatı(script) **kly Paket Sistemi**'ni kullanarak paketi derler ve oluştur. Oluşan paket(**.kly uzantılı dosya**) **kly Paket Sistemi** kullanılarak siteme yüklenebilir. **kly Paket Sistemiyle Paket Yapma** konusunu okumak için tıklayınız.

ISO Hazırlama

Dağıtıma özgü ISO oluşturmak için o dağıtımın paket sistemi kullanılır. Bu dokümanda anlatılan **kly** paket sistemi, önceki bölümde detaylıca açıklanmıştır.

Bu bölümde, **kly** paket sistemini kullanarak ISO oluşturma adımları basit ve anlaşılır şekilde verilecektir. Aşağıda payşlaşılan script ile hazırlanan iso **/tmp/distro/distro.iso** konumunda olacaktır. Sistemi oluşturan paketleri **github.com/kendilinuxunuyap/kly-base-packages** ve **github.com/kendilinuxunuyap/kly-x11-packages** konumundan indirerek oluşturmaktadır.

Not: Anlatılan yöntem genel Linux dağıtımlarının ISO oluşturma mantığına dayanır. Araçlar, dizinler ve komutlarda küçük farklılıklar olabilir, ancak iş akışı aynıdır.

Aşağıdaki scriptle oluşturulmuş iso https://github.com/kendilinuxunuyap/kly-x11-distro/releases/download/current/kly-x11-distro.iso adresinde bulunmaktadır.

kly Paket Sistemiyle ISO Oluşturma Scripti

```
#-------------------------------------------------------------------------------------------------------------------
#!/bin/bash
set -x
if which apt &>/dev/null && [[ -d /var/lib/dpkg && -d /etc/apt ]] ; then
        apt-get update
        echo "işlem başladı....."
        apt install xorriso grub-pc-bin grub-efi mtools make python3 dosfstools e2fsprogs squashfs-tools \
        python3-yaml gcc wget curl unzip xz-utils debootstrap git erofs-utils zstd -y
fi
apt-get install git devscripts equivs -y
#-------------------------------------------------------------------------------------------------------------------
rootfs="/tmp/distro/rootfs"
distro="/tmp/distro"
rm -rf $distro/iso
#rm -rf $rootfs
mkdir -p /tmp/distro/rootfs
mkdir -p $rootfs/bin
#mkdir -p distro/rootfs
#export PATH=/bbin:$PATH
cp busybox $rootfs/bin/busybox
cp kly $rootfs/bin/kly
cd $rootfs/bin/
ln -s busybox cpio
#busybox --install -s ./
cd $rootfs/
#cd distro/rootfs/
mkdir  -p var run dev sys proc etc tmp/kly/kur
bash -c "echo '127.0.0.1 kly' >> $rootfs/etc/hosts"
bash -c "echo 'kly' > $rootfs/etc/hostname"
bash -c "echo 'nameserver 8.8.8.8' > $rootfs/etc/resolv.conf"

### system chroot  bind/mount
for i in dev dev/pts proc sys; do mount -o bind /$i $rootfs/$i; done
### manuel kly-tools install
$rootfs/bin/busybox wget -O $rootfs/tmp/base-file-1.0.kly https://github.com/bpslinux/\
kly-temel-paketler/raw/refs/heads/master/base-file/base-file-1.0.kly 1>$rootfs/dev/null 2>$rootfs/dev/null
$rootfs/bin/busybox tar -xf $rootfs/tmp/base-file-1.0.kly -C $rootfs/tmp/kly/kur
$rootfs/bin/busybox tar -xf $rootfs/tmp/kly/kur/rootfs.tar.xz -C $rootfs

#paket adresi ekleniyor
$rootfs/bin/busybox mkdir -p $rootfs/etc/kly
echo "kendilinuxunuyap/kly-base-packages">$rootfs/etc/kly/sources.list
echo "kendilinuxunuyap/kly-x11-packages">>$rootfs/etc/kly/sources.list

### installing kly package in rootfs
$rootfs/bin/kly -u $rootfs
#**********************************************************************
echo root:x:0:0:root:/root:/bin/sh > $rootfs/etc/passwd
chmod 755 $rootfs/etc/passwd
#**********************************************************************

for paket in glibc readline ncurses bash openssl acl attr libcap libpcre2 gmp coreutils sqlite  \
util-linux grep sed mpfr gawk findutils libgcc libcap-ng gzip xz-utils zstd bzip2 elfutils libselinux tar zlib \
brotli curl
do
$rootfs/bin/kly -ri $paket $rootfs
#chroot $rootfs /bin/kly -ri $paket;
done
```

```
#-------------------------------------------------------------------------------------------------------------------
# scriptin devamı

for paket in shadow file eudev cpio libsepol kmod audit libxcrypt libnsl libbsd libtirpc \
e2fsprogs dosfstools initramfs-tools libxml2 expat libmd libaio lvm2 popt icu iproute2 \
net-tools dhcp openrc rsync kbd kernel dialog live-boot live-config parted busybox nano grub \
efibootmgr efivar libssh openssh pam
do
#$rootfs/bin/kly -i $paket $rootfs
chroot $rootfs /bin/kly -ri $paket;
done

# burası x11 için olan paketlerdi. arasınsa ek paket varmı kontrol etmedim. edeceğim
for paket in xorg-server pixman libpciaccess libXau libXdmcp libXfont2 libxshmfence libdrm libxcvt libfontenc freetype \
libpng harfbuzz glib \
xterm libXft fontconfig libXext libXaw libXmu libXinerama libXpm libXt libX11 libICE libXrender libxcb libSM \
xf86-input-libinput xf86-input-vmmouse xf86-video-amdgpu xf86-video-ast xf86-video-ati xf86-video-dummy \
xf86-video-fbdev xf86-video-intel \
xf86-video-mga xf86-video-nouveau xf86-video-qxl xf86-video-r128 xf86-video-siliconmotion xf86-video-vboxvideo \
xf86-video-vesa xkbcomp libxkbfile libglvnd mesa xkeyboard-config libinput mtdev libevdev libwacom libgudev libffi \
xinit xcalc libXi openbox libXcursor libXfixes pango libXrandr fribidi xcb-util libthai libdatrie startup-notification \
dejavu libunwind dbus polkit elogind
do
chroot $rootfs /bin/kly -ri $paket;
done

### user add and passwd
chroot $rootfs echo -e "1\n1\n"|chroot $rootfs passwd root
chroot $rootfs useradd live -m -s /bin/sh -d /home/live
chroot $rootfs echo -e "live\nlive\n"|chroot $rootfs passwd live
for grp in users tty wheel cdrom audio dip video plugdev netdev; do
chroot $rootfs usermod -aG $grp live || true
done

### update-initrd
fname=$(basename $rootfs/boot/config*)
kversion=${fname:7}
mv $rootfs/boot/config* $rootfs/boot/config-$kversion
cp $rootfs/boot/config-$kversion $rootfs/etc/kernel-config

chroot $rootfs update-initramfs -u -k $kversion

#### system chroot umount
for dir in dev dev/pts proc sys ; do    while umount -lf -R $rootfs/$dir 2>/dev/null ; do true; done done

#***********************iso ********************************
mkdir -p $distro/iso
mkdir -p $distro/iso/boot
mkdir -p $distro/iso/boot/grub
mkdir -p $distro/iso/live || true

#### Copy kernel and initramfs (Debian/Devuan)
cp -pf $rootfs/boot/initrd.img-* $distro/iso/boot/initrd.img
cp -pf $rootfs/boot/vmlinuz-* $distro/iso/boot/vmlinuz
rm -rf $rootfs/boot
#### Create squashfs
mksquashfs $rootfs $distro/filesystem.squashfs -comp xz -wildcards
mv $distro/filesystem.squashfs $distro/iso/live/filesystem.squashfs

# grub.cfg dosyasını yaz
cat << EOF > "$distro/iso/boot/grub/grub.cfg"
set timeout=6; set default=1; terminal_input console;
menuentry "Canli(live) GNU/Linux 64-bit" --class liveiso {
linux /boot/vmlinuz boot=live init=/sbin/openrc-init net.ifnames=0 \
biosdevname=0
initrd /boot/initrd.img
}
menuentry "Kur GNU/Linux 64-bit" --class liveiso {
linux /boot/vmlinuz boot=live init=/bin/kur quiet
initrd /boot/initrd.img
}
EOF

#iso oluşturuluyor
grub-mkrescue $distro/iso/ -o $distro/distro.iso
```

Kaynaklar:

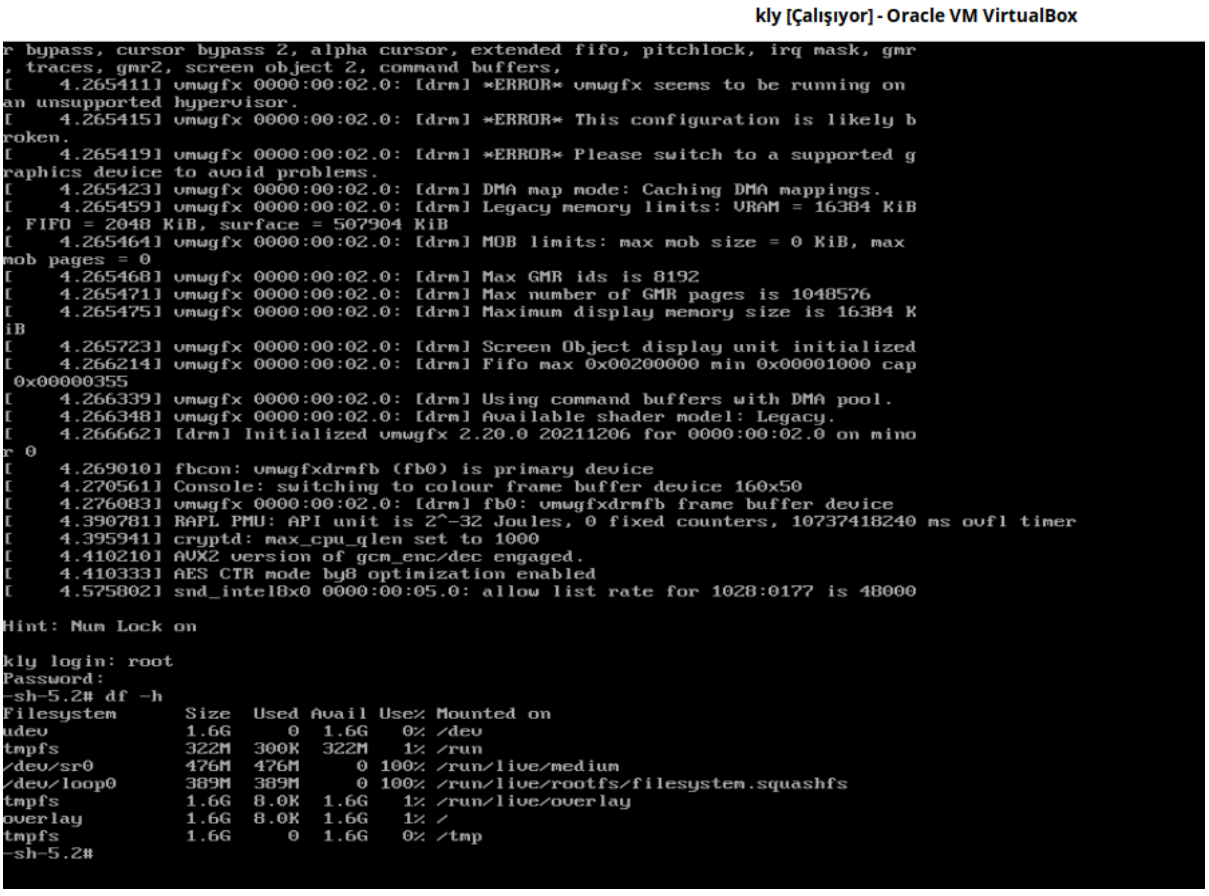

```
- https://www.subrat.info/build-kernel-and-userspace/ - 08/07/2025
- https://medium.com/@chienhaotan/compiling-and-running-a-minimal-kernel-with-busybox-bfc45a991017 - 08/07/2025
- https://wiki.archlinux.org/title/GRUB_(T%C3%BCrk%C3%A7e)  - 08/07/2025
- https://chatgpt.com/share/6875084c-6050-8012-9229-a37b47351aa2 - 14/07/2025
```

Oluşan Sistemin Çalıştırılması İncelenmesi

Bu dokümanda anlatılan paketlerin **kly Paket Sistemiyle** hazırlanmış **X Pencere Sistemi** isosu hazırlandı. İsoyu https://github.com/kendilinuxunuyap/kly-x11-distro/releases/download/current/kly-x11-distro.iso adresinden indirebilirsiniz.

Şimdi hazırlanan ISO'yu **QEMU** veya **VirtualBox** kullanarak çalıştıralım. Ekran görüntüleri aşağıda verilmiştir.

Canlı(live) Sistem Kullanımı

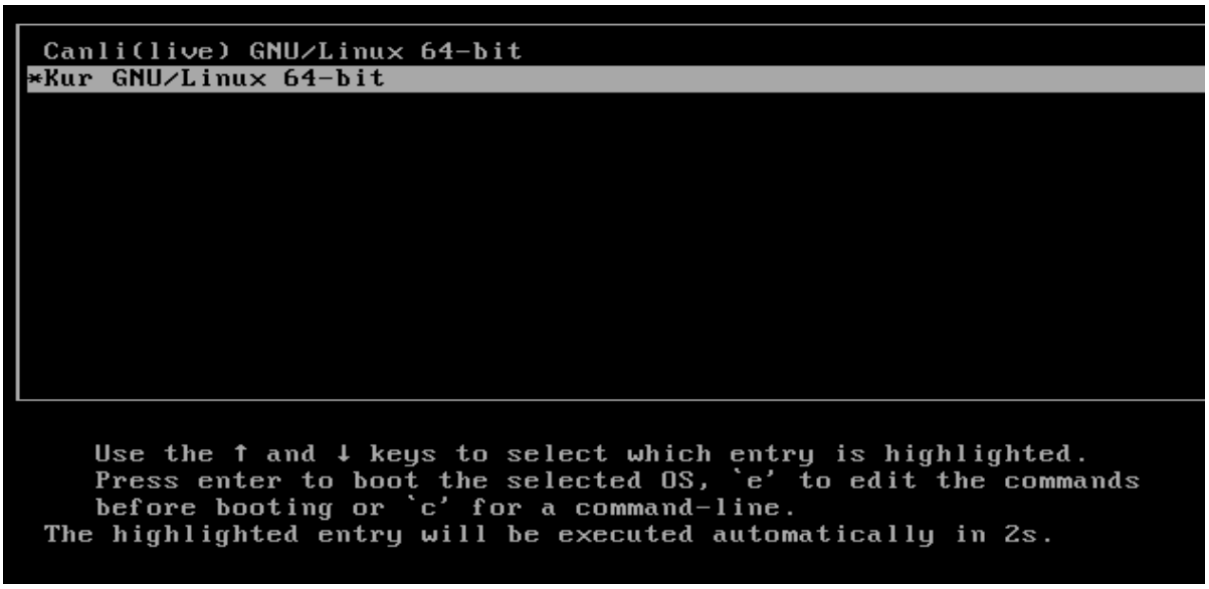


Canlı(live) sistem çalıştırıldığında overlay live bir sistemin açıldığını görmekteyiz. Canlı(live) sistemde kullanıcı adları ve parolaları;

- Kullanıcı: root Parola: 1
- Kullanıcı: live Parola: live

Sistem Kurulumu

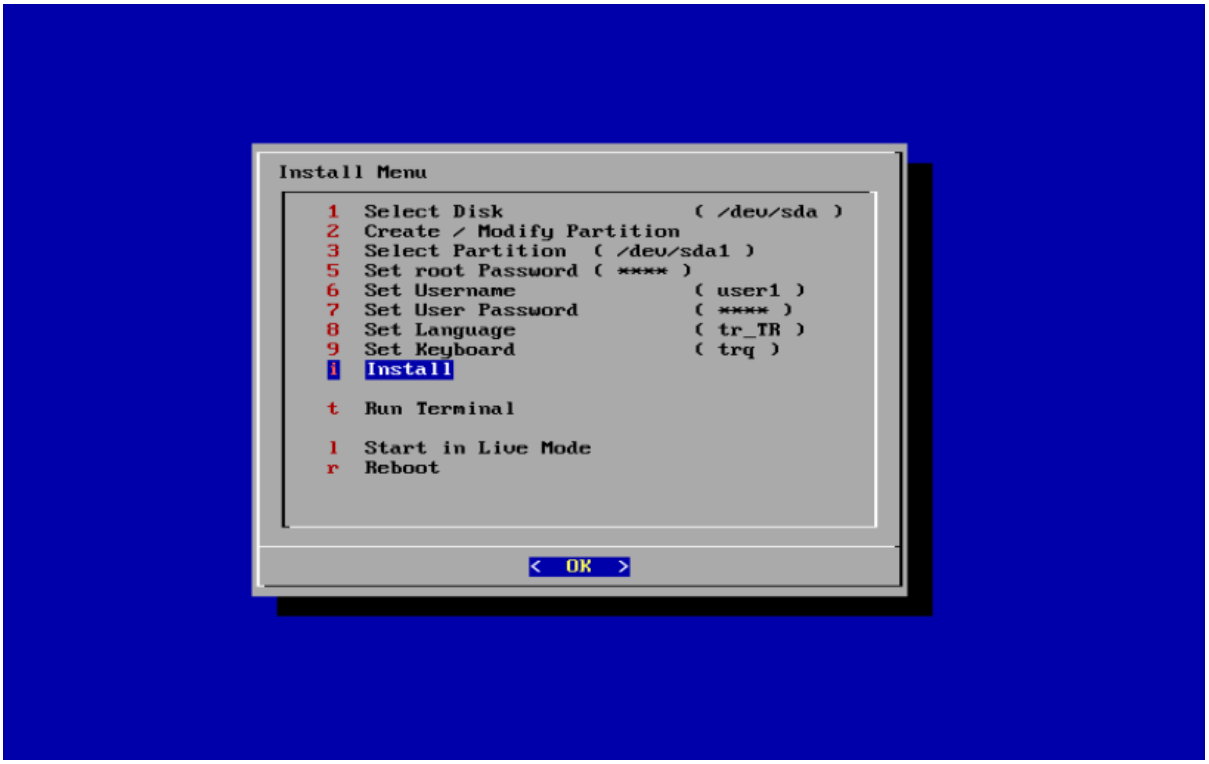
Sistemin kurulumu için resimlerde görünen sıraya göre seçimler yapmalıyız.



Kurulum menüsünde kullanıcı adları ve parolaları, klayve varsayılan olarak;

- Kullanıcı: root Parola: 1
- Kullanıcı: user1 Parola: 1
- Dil : tr_TR
- Klavye : trq

menüden değişiklik yapabilirsiniz. Değişiklik yapmadan sadece kurulum diskini ve disk bölümünü seçip Install(Yükle) işlemi yapabilirsiniz.

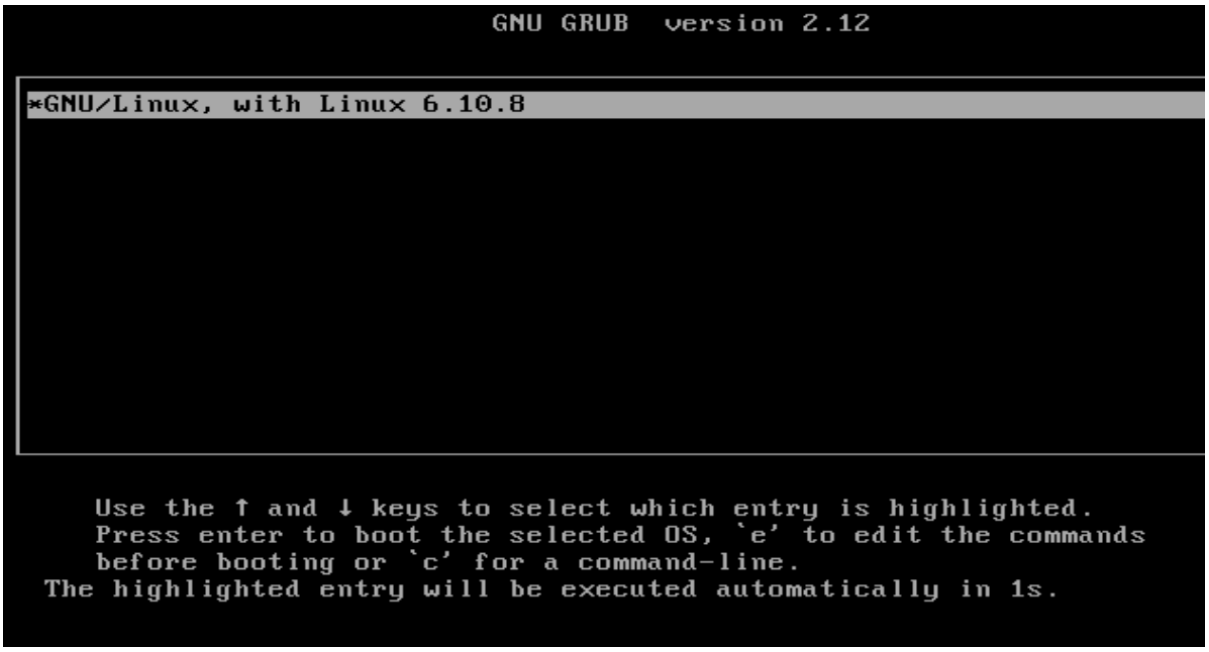


kly [Çalışıyor] - Oracle VM VirtualBox

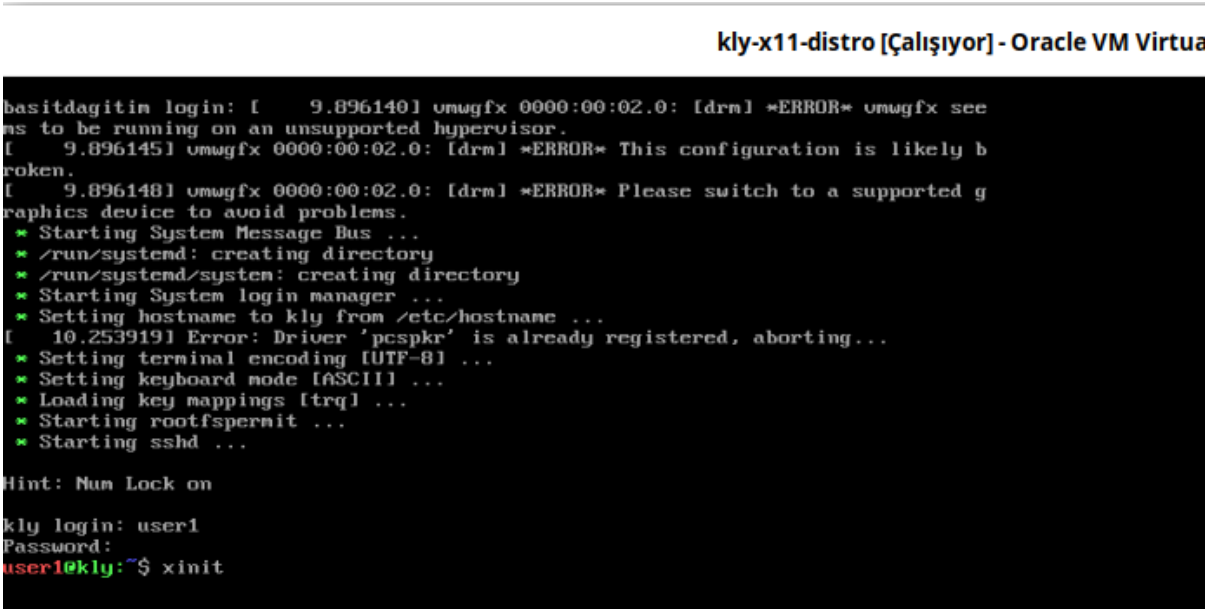
```
kuruluma geçildi........
legacy kurulum ........
Kurulum Yap■lacak Disk sda
mount: /kaynak: WARNING: source write-protected, mounted read-only.
**********disk bölümleri biçimlendiriliyor ******************************
e2fsck 1.47.0 (5-Feb-2023)
e2fsck: need terminal for interactive repairs
tune2fs 1.47.0 (5-Feb-2023)
Recovering journal.

This operation requires a freshly checked filesystem.

Please run e2fsck -f on the filesystem.

mke2fs 1.47.0 (5-Feb-2023)
Creating filesystem with 2096896 4k blocks and 524288 inodes
Filesystem UUID: 9ea082d0-a034-4dab-8e2f-564e68c99c9c
Superblock backups stored on blocks:
        32768, 98304, 163840, 229376, 294912, 819200, 884736, 1605632

Allocating group tables: done
Writing inode tables: done
Creating journal (16384 blocks): done
Writing superblocks and filesystem accounting information: done

Information: You may need to update /etc/fstab.

********** kurulum ba■lad■**********
Sistem yuklenmeye baslandi. Tahmini 3-5 dakika surecektir.. Lutfen bekleyiniz.....
.....
_
```

Sistemin Çalışması

Sistem kurulumu gerçekleştiğinde sistem resimde görüldüğü gibi açılmalıdır.



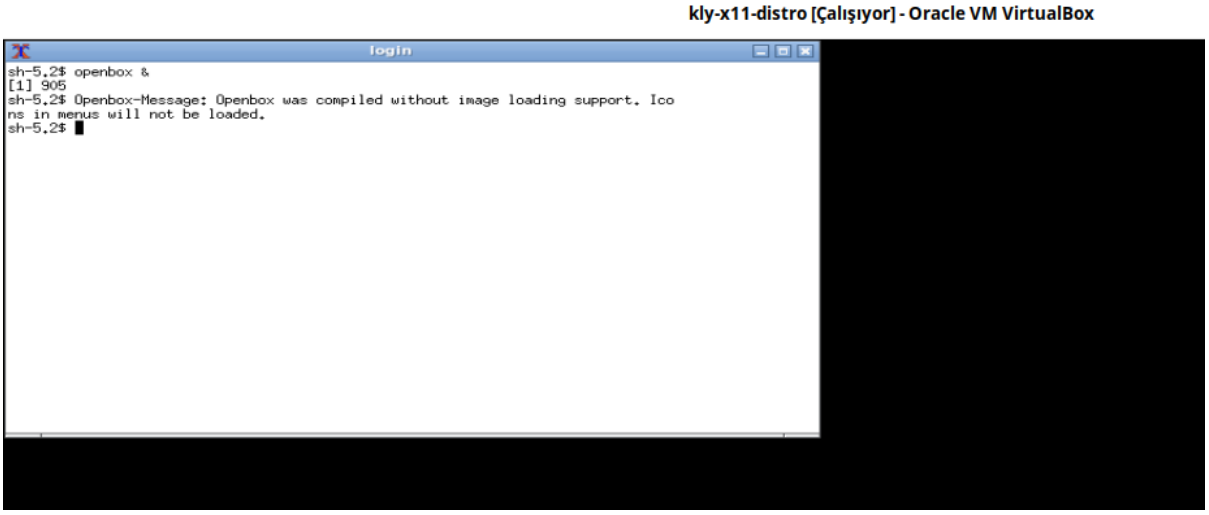
Sisteme **user1** (parola=1) kullanıcısı olarak giriş yapıldığı görülmektedir.



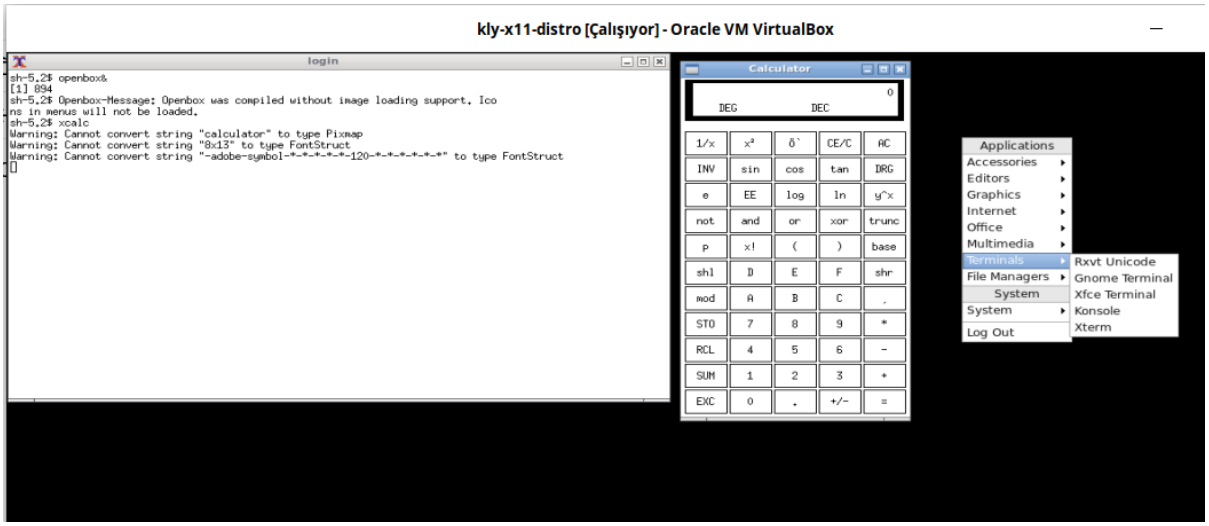
xinit çalışınca **X Pencere Sistemine** geçiş yapılacak.

xinit çalışınca **X Pencere Sistemi** aşağıdaki gibi açılacaktır.

X Pencere Sistemi çalıştırılmış ve pencere yönetimini ve masaüstü ortamı için **openbox** çalıştırılıyor.



xcalc uygulamasının çalıştırılması aşağıda gösterilmiştir.



X Pencere Sistemi

Xorg Çalıştırma

Xorg Yapılandırma Dosyası

Xorg, genellikle /etc/X11/xorg.conf dosyasını kullanır. Eğer bu dosya yoksa, Xorg varsayılan ayarlarla çalışacaktır. Ancak, özel ayarlar yapmak istiyorsanız, bu dosyayı oluşturmanız gerekebilir. Aşağıdaki komut ile bir yapılandırma dosyası oluşturabilirsiniz:

```
sudo X -configure
```

Bu komut, xorg.conf.new adında bir dosya oluşturacaktır. Bu dosyayı /etc/X11/xorg.conf olarak kopyalayabilirsiniz:

```
sudo cp ~/xorg.conf.new /etc/X11/xorg.conf
```

Xwrapper.config

```
echo needs_root_rights=yes>/etc/X11/Xwrapper.config
echo allowed_users=anybody>>/etc/X11/Xwrapper.config
```

Kullanıncı Ayarı

Kullanıcının tty ve wheel grubunda olması lazım ayrıca **/dev/tty*** dosyalarının grub ve izinleri ayarlanmalıdır.

```
chmod 620 /dev/tty*
chgrp tty /dev/tty*
usermod -a -G tty by
addgroup wheel
usermod -a -G wheel by
```

Xorg'u Elle Başlatma

Xorg'u elle başlatmak için terminalde aşağıdaki komutu kullanabilirsiniz:

```
export DISPLAY=:0
Xorg:0
```

Bu şekilde çalıştırmak yerine;

```
startx
```

Bu komut, varsayılan kullanıcı arayüzünü başlatacaktır. Eğer belirli bir pencere yöneticisi veya masaüstü ortamı ile başlatmak istiyorsanız, ~/.xinitrc dosyasını düzenlemeniz gerekebilir. Örneğin, openbox kullanmak istiyorsanız, dosyanın içeriği şöyle olmalıdır:

```
exec openbox-session
```

Xorg'u Durdurma

Xorg'u durdurmak için, terminalde Ctrl + Alt + Backspace tuş kombinasyonunu kullanabilirsiniz. Bu, X sunucusunu kapatacaktır.

Xorg Hata ayıklama

Hata Ayıklama

Eğer Xorg başlatılamıyorsa, hata ayıklamak için aşağıdaki komutu kullanarak log dosyalarını kontrol edebilirsiniz:

```
cat /var/log/Xorg.0.log
```

Bu dosya, Xorg'un başlatılması sırasında oluşan hataları ve uyarıları içerecektir. Hataları inceleyerek sorunu çözmeye çalışabilirsiniz.

Yardımcı Konular

apt Paket Sistemi

apt paket sistemi debian tabanlı sistemlerde paket yönetimi için kullanılan bir uygulamadır. Temel işlemler şunlardır.

Yerel Paket İndexini Güncelleme

```
sudo apt  update
```

Paket Yükleme

```
sudo apt install paket_adi
```

Paket Silme

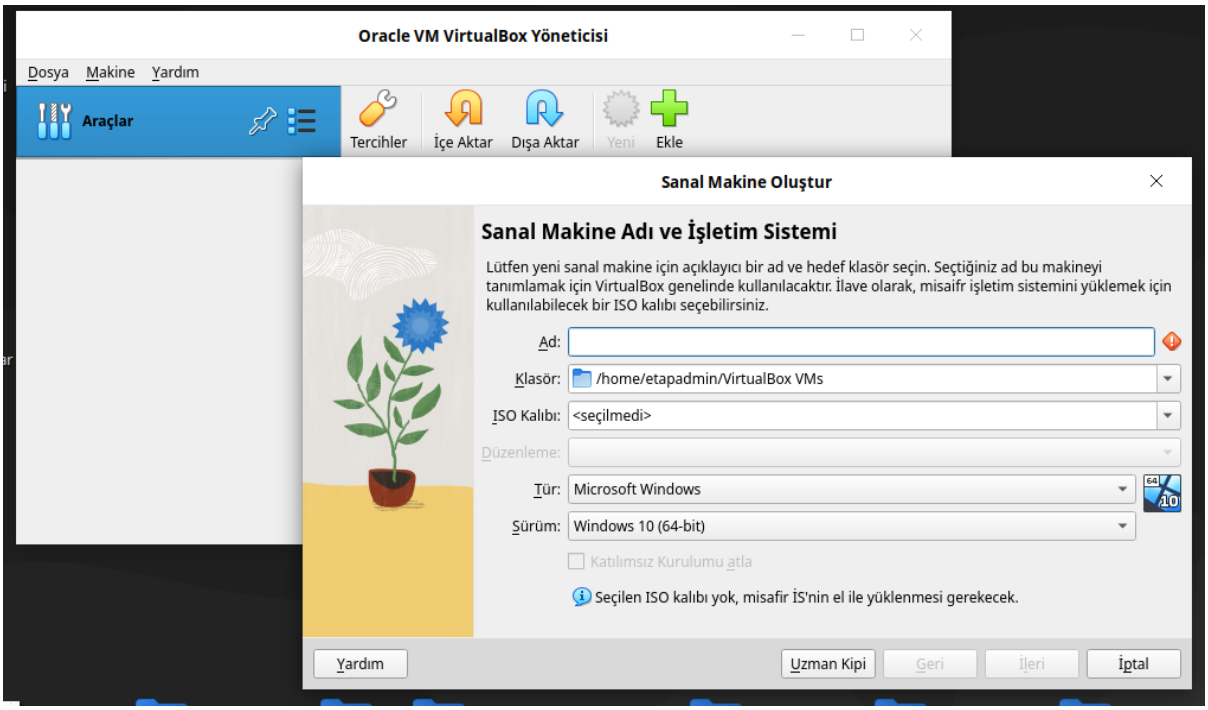
```
sudo apt remove paket_adi
```

VirtualBox

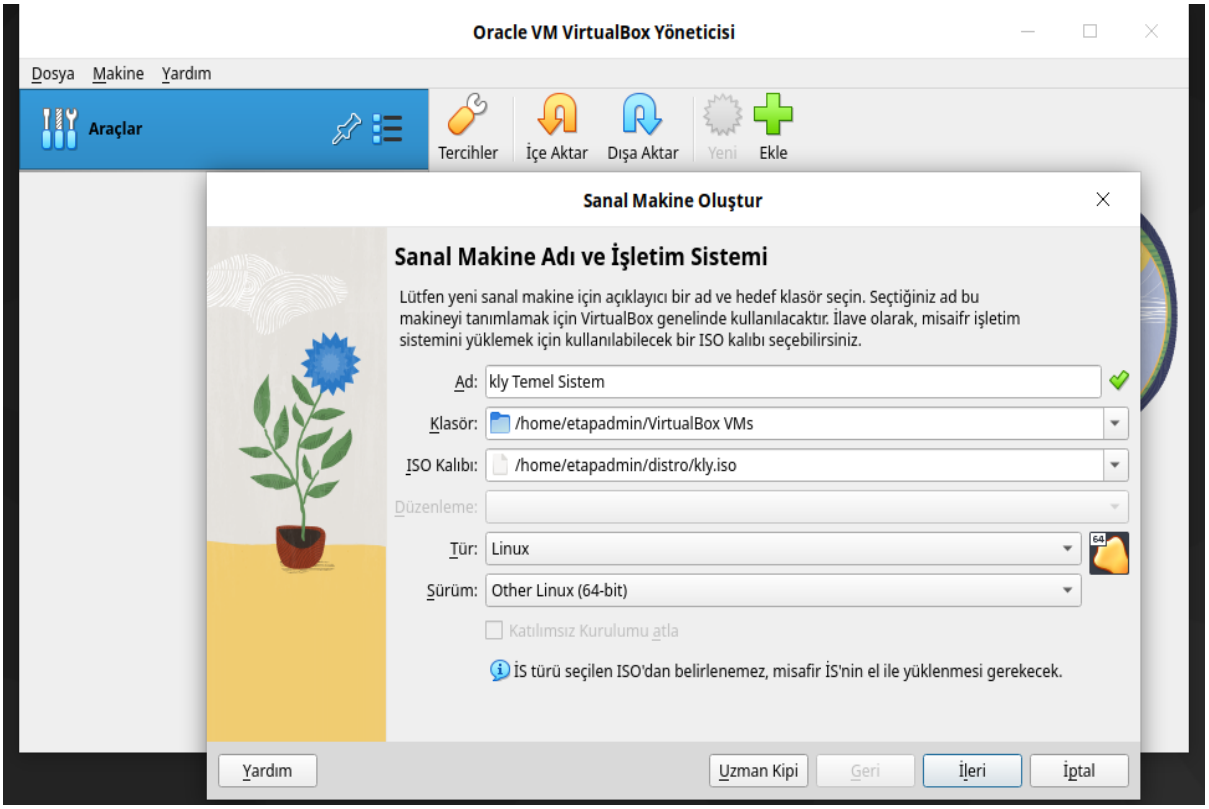
VirtualBox, Oracle tarafından geliştirilen bir açık kaynaklı sanallaştırma yazılımıdır. Bilgisayarınızda çalışan bir işletim sisteminin (örneğin Windows, Linux, macOS) içinde başka bir işletim sistemi çalıştırmanıza izin verir. Bu çalıştırdığınız ikinci işletim sistemi (konuk/guest) kendi sanal donanımına sahipmiş gibi davranır.

```
sudo apt update                          # index güncellenir
sudo apt install virtualbox-7.0          # Sisteme virtualbox kurulur
```

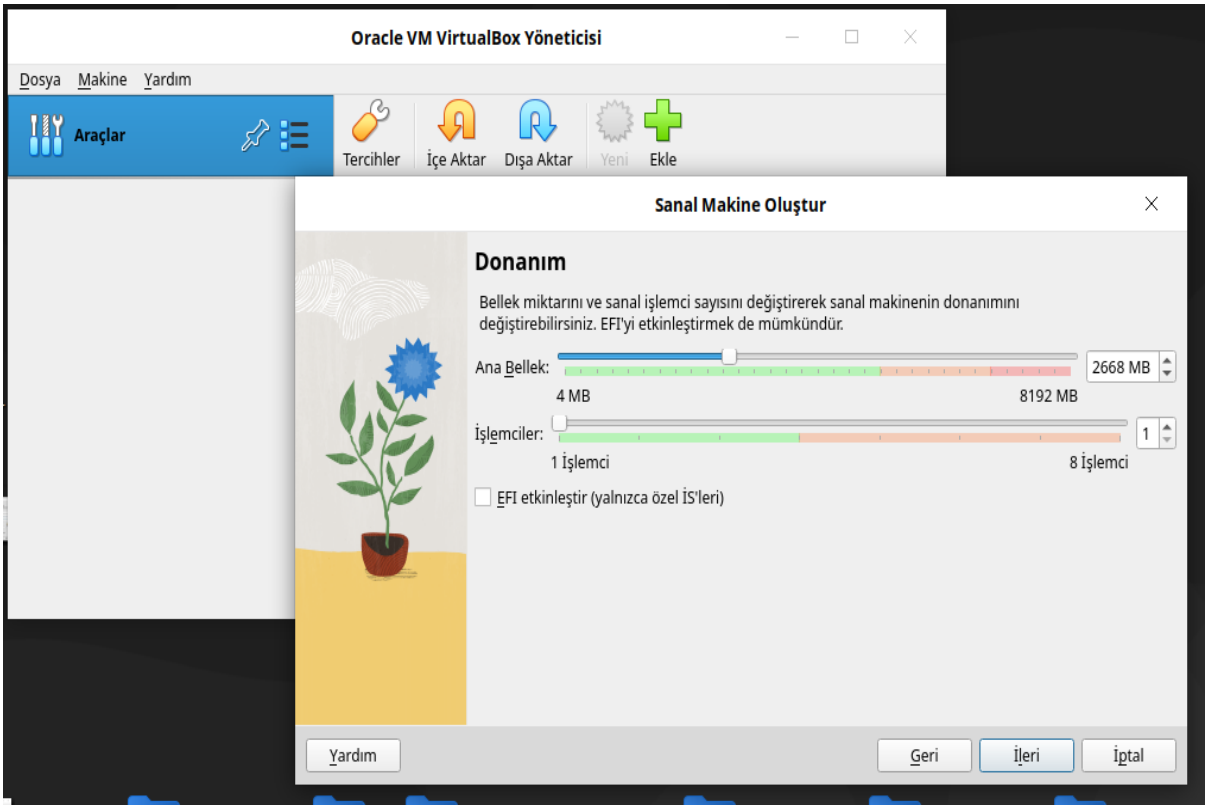
Ekle seçeneğini seçin. Aşağıdaki gibi bir ekran gelecektir.



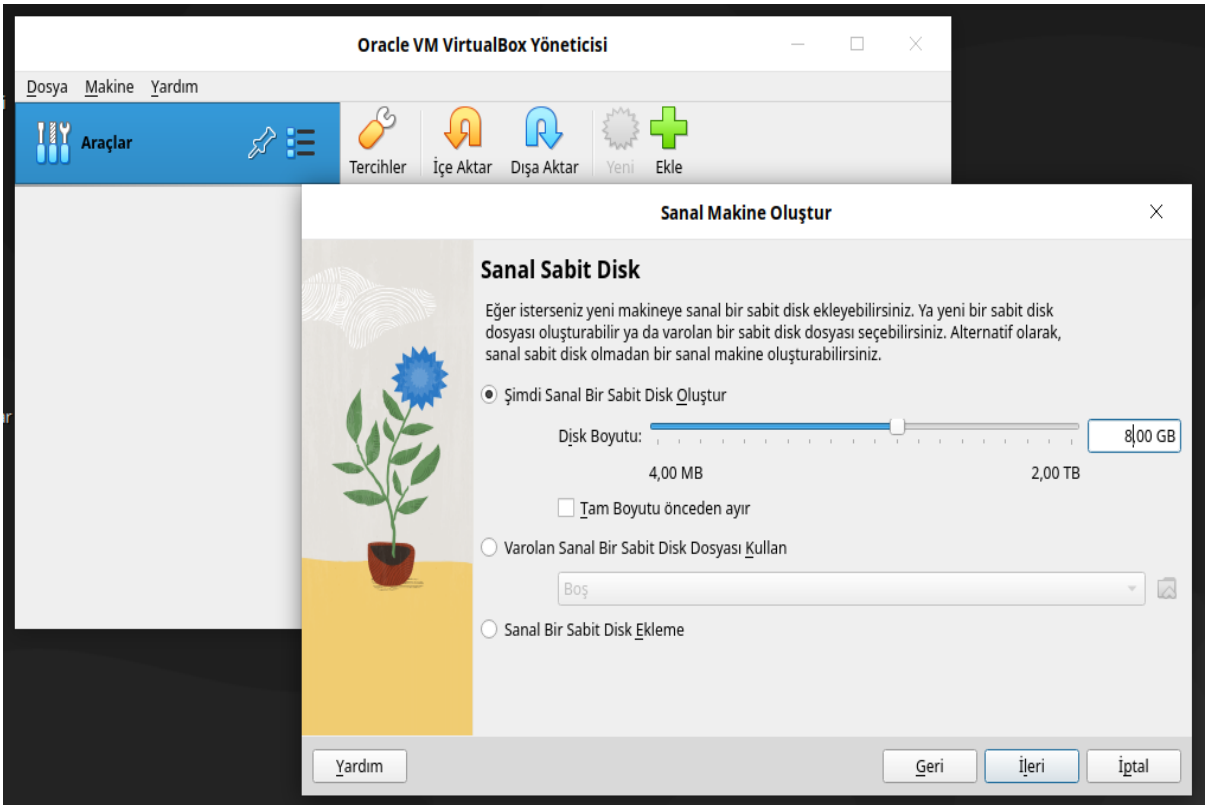
Aşağıdaki gibi seçenekleri doldurunuz.



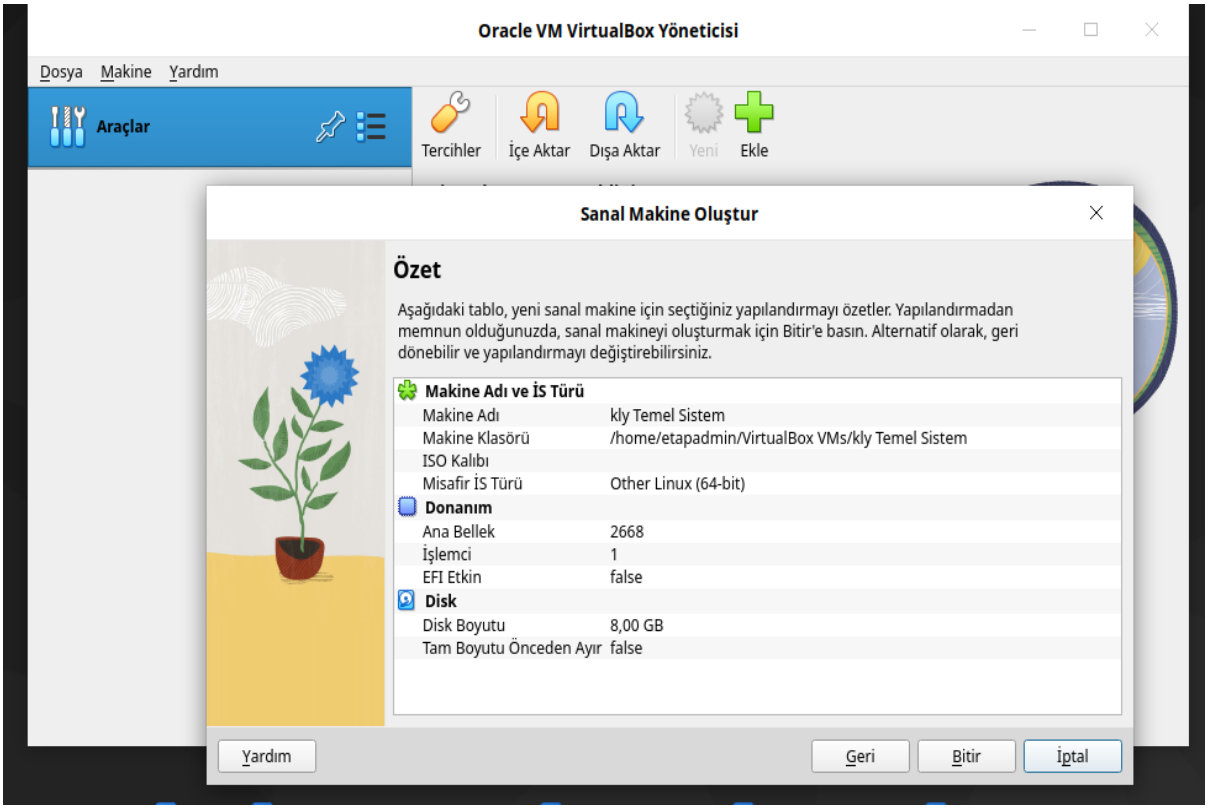
İleri seçeneği seçilir ve aşağıdaki gibi ekrana gelirsiniz. Bellek miktarını aşağıdaki gibi belirleyiniz. **İleri** seçeneği seçilir.



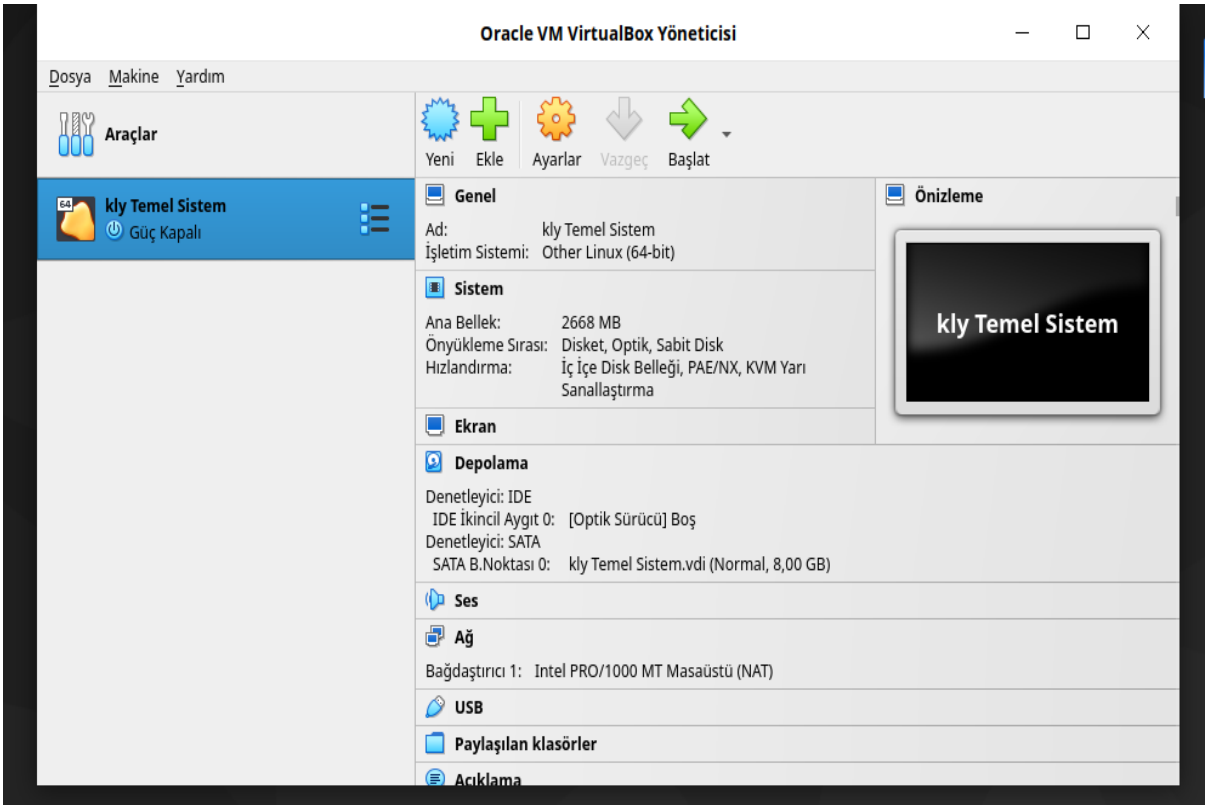
Disk boyutu belirlenir. 8GB bu sistem için yeterli. **İleri** seçeneği seçilir.



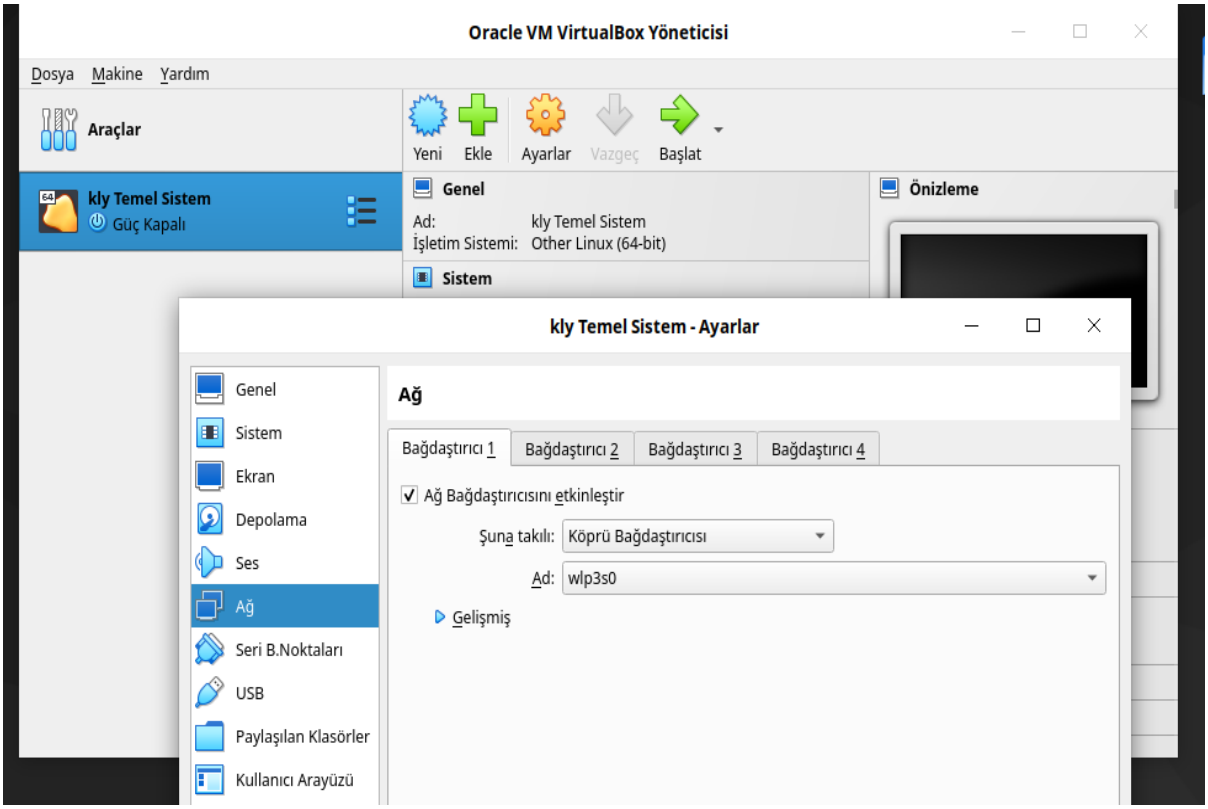
Tüm işlemler bitince aşağıdaki gibi pencere gelir. Bitir'i seçerek işlem tamamlanır.



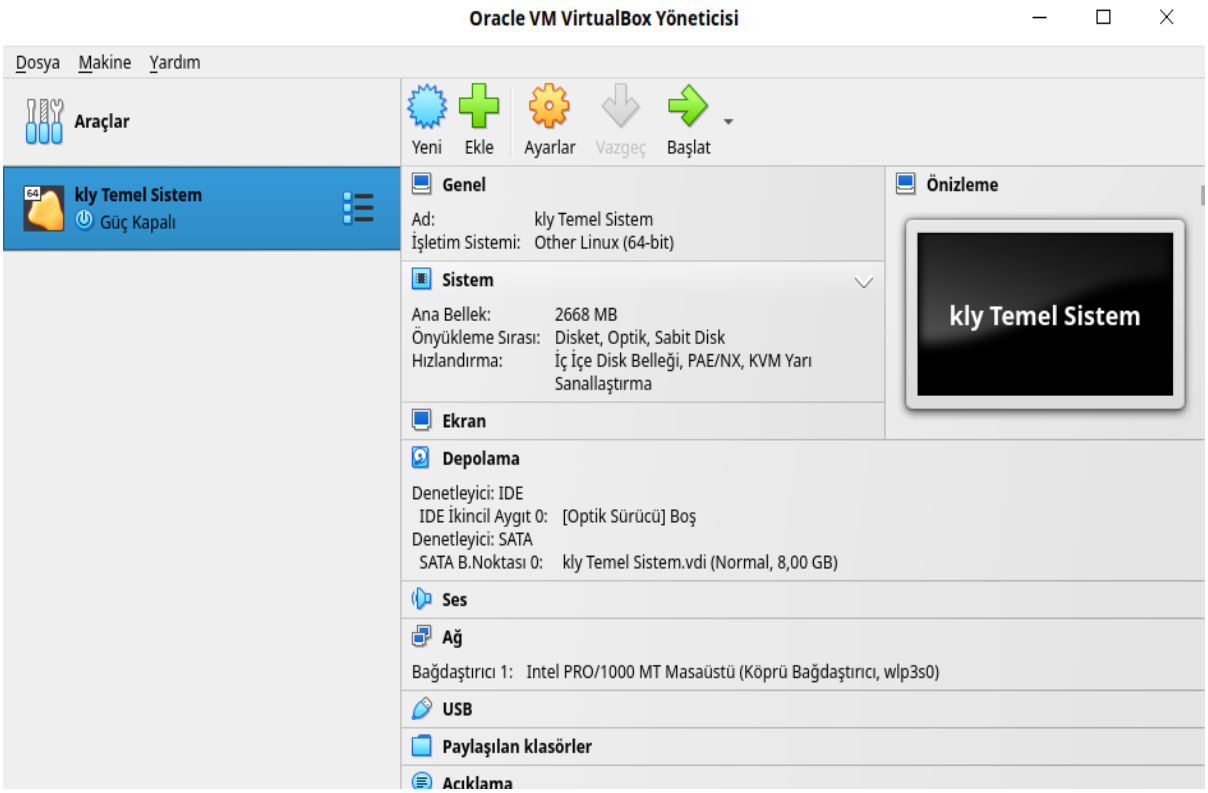
Bitir işlemi seçildikten sonra aşağıdaki gibi görünecektir.



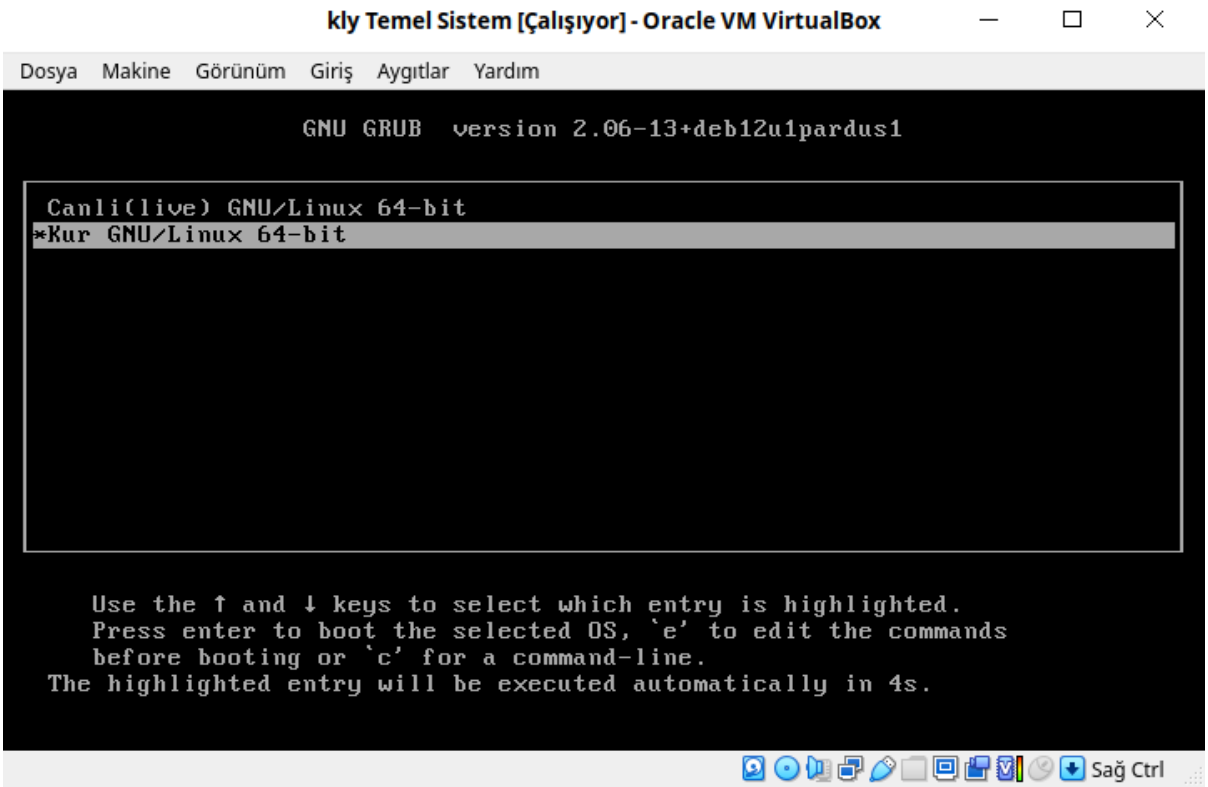
Ayarlar bölümünde aşağıdaki gibi **Ağ** ayarında değişiklik yapılır.



Başlat seçilerek sistem çalıştırılır.



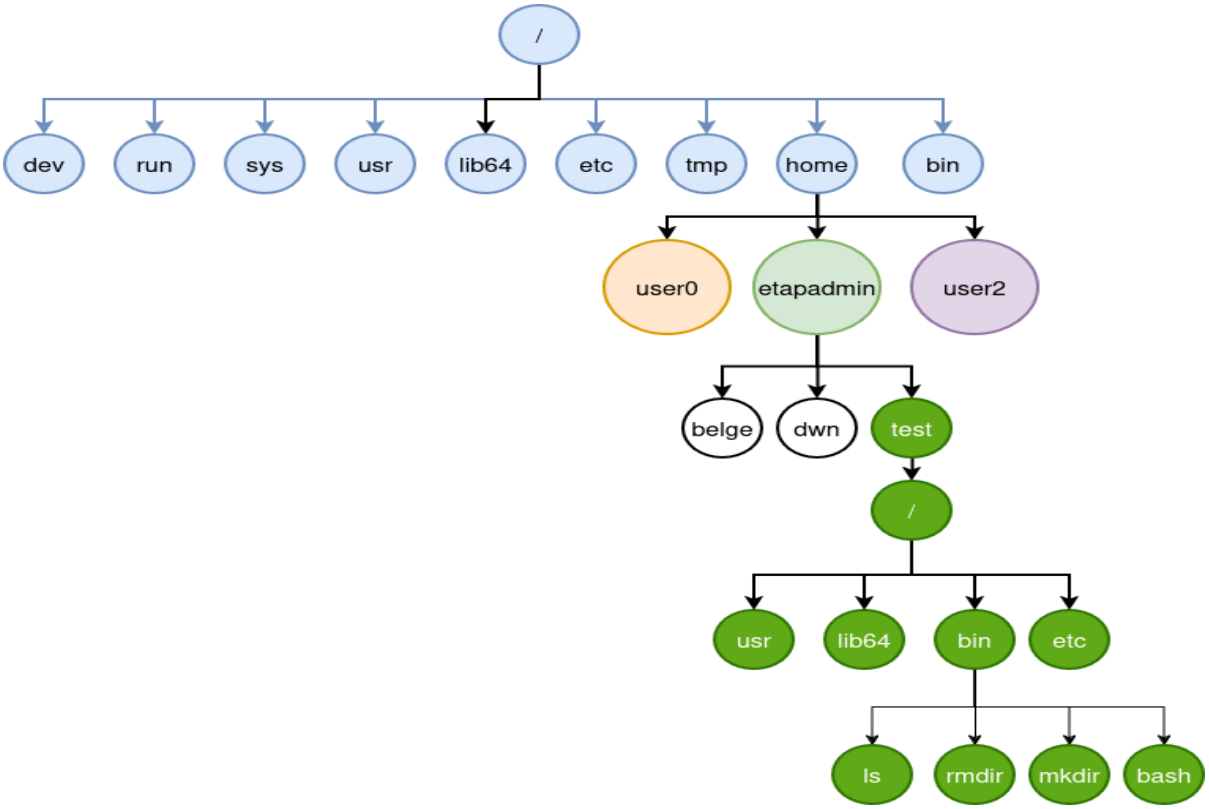
İso açıldığıda aşağıdaki gibi ekran gelecektir. Burada seçili olan iso **Temel Sistem** isosudur.



Chroot Nedir?

chroot komutu çalışan sistem üzerinde belirli bir klasöre root yetkisi verip sadece o klasörü sanki linux sistemi gibi çalıştıran bir komuttur. Sağladığı avantajlar çok fazladır. Bunlar;

- Sistem tasarlama
- Sitem üzerinde yeni dağıtımlara müdahale etme ve sorun çözme
- Kullanıcı kendine özel geliştirme ortamı oluşturabilir.
- Yazılım bağımlıkları sorunlarına çözüm olabilir.
- Kullanıcıya sadece kendisine verilen alanda sınırsız yetki verme vb.



Yukarıdaki resimde user1 altında wrk dizini altına yeni bir sistem kurulmuş gibi yapılandırmayı gerçekleştirmiş.

/home/etapadmin/test dizinindeki sistem üzerinde sisteme erişmek için;

```
# sisteme erişim yapıldı.
sudo chroot /home/etapadmin/test
```

/home/etapadmin/test dizinindeki sistem üzerinde sistemi silmek için;

```
# sistem silindi
sudo rm -rf /home/etapadmin/test
```

Yeni sistem tasarlamak ve erişmek için temel komutları ve komut yorumlayıcının olması gerekmektedir. Bunun için bize gerekli olan komutları bu yapının içine koymamız gerekmektedir. Örneğin ls komutu için doğrudan çalışıp çalışmadığını ldd komutu ile kontrol edelim.

Görüldüğü gibi ls komutunun çalışması için bağımlı olduğu kütüphane dosyaları bulunmaktadır. Bağımlı olduğu dosyaları yeni oluşturduğumuz sistem dizinine aynı dizin yapısında kopyalamamız gerekmektedir. Bu dosyalar eksiksiz olursa ls komutu çalışacaktır. Fakat bu işlemi tek tek yapmamız çok zahmetli bir işlemdir. Bu işi yapacak script dosyası aşağıda verilmiştir.

Bağımlılık Scripti

lddscript.sh

```
#!/bin/bash

# Betik iki parametre bekler: kopyalanacak dosya ve hedef klasör
if [ ${#} != 2 ]; then
        echo "Kullanım: $0 PATH_TO_BINARY hedef_klasor"
        exit 1
fi

path_to_binary="$1"
target_folder="$2"

# Dosya yoksa işlem durur
if [ ! -f "${path_to_binary}" ]; then
        echo "Dosya '${path_to_binary}' bulunamadı. İşlem iptal ediliyor!"
        exit 1
fi

# Dosyayı kopyala
echo "Dosya kopyalanıyor..."
cp --parents -v "${path_to_binary}" "${target_folder}"

# Bağımlı kütüphaneleri kopyala
echo "Kütüphaneler kopyalanıyor..."
ldd "${path_to_binary}" | awk -F'[> ]' '{print $(NF-1)}' | while read -r lib; do
        [ -f "$lib" ] && cp -v --parents "$lib" "${target_folder}"
done
```

Basit Sistem Oluşturma

Bu örnekte kullanıcının(etapadmin) ev dizinine(/home/etapadmin) test dizini oluşturuldu ve işlemler yapıldı. ls, rmdir, mkdir ve bash komutlarından oluşan sistem hazırlama.

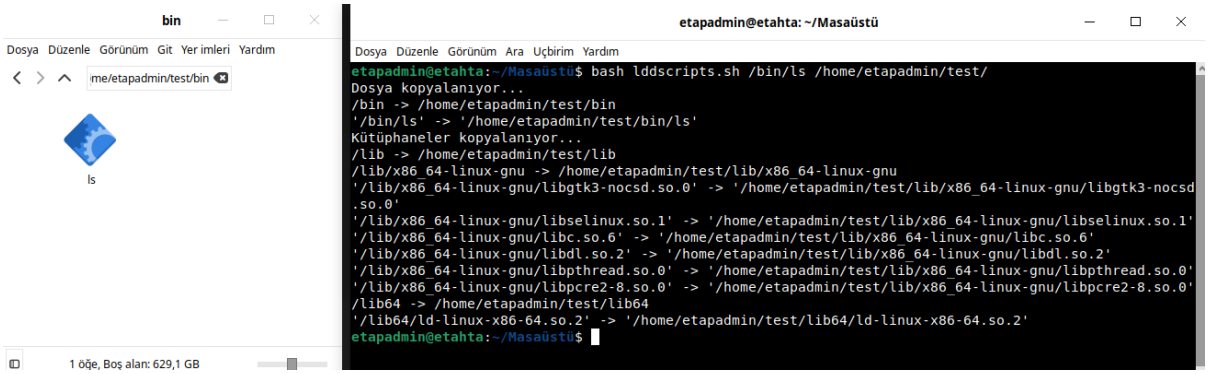
Sistem Dizinin Oluşturulması

```
# ev dizinine test dizini oluşturuldu.
mkdir /home/etapadmin/test/
```

/home/etapadmin/ dizinine **Bağımlılık Scripti** kodunu **lddscripts.sh** oluşturalım.

ls Komutu

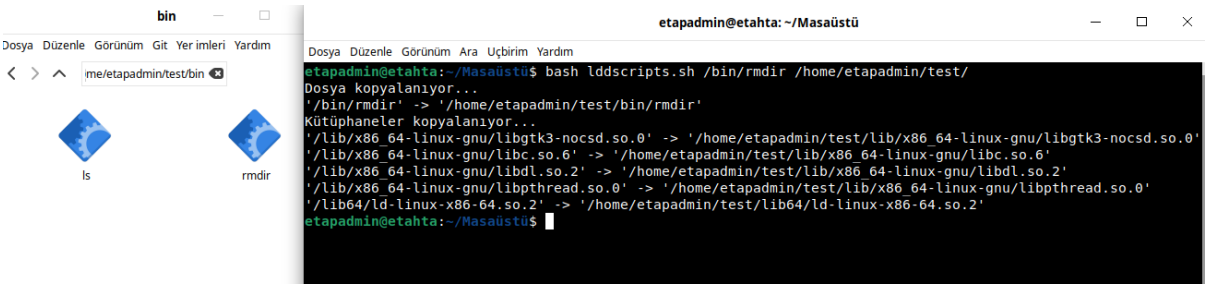
```
bash lddscripts.sh /bin/ls /home/etapadmin/test/
```



Bu işlemi diğer komutlar içinde sırasıyla yapmamız gerekmektedir.

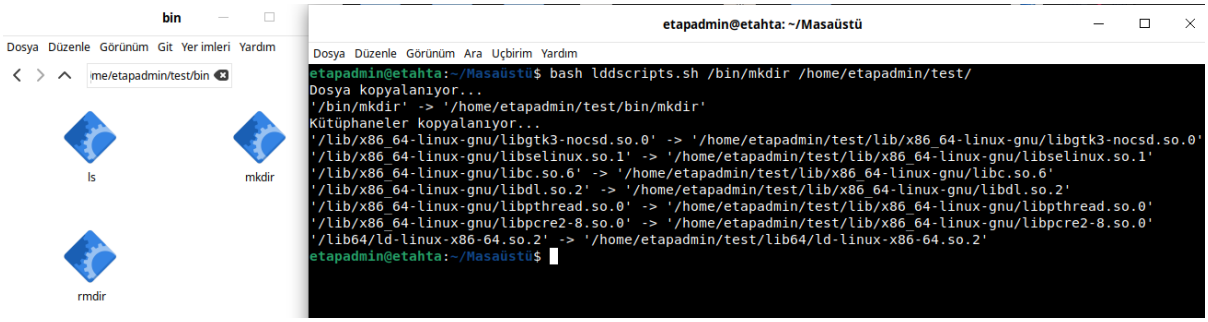
rmdir Komutu

```
bash lddscripts.sh /bin/rmdir /home/etapadmin/test/
```



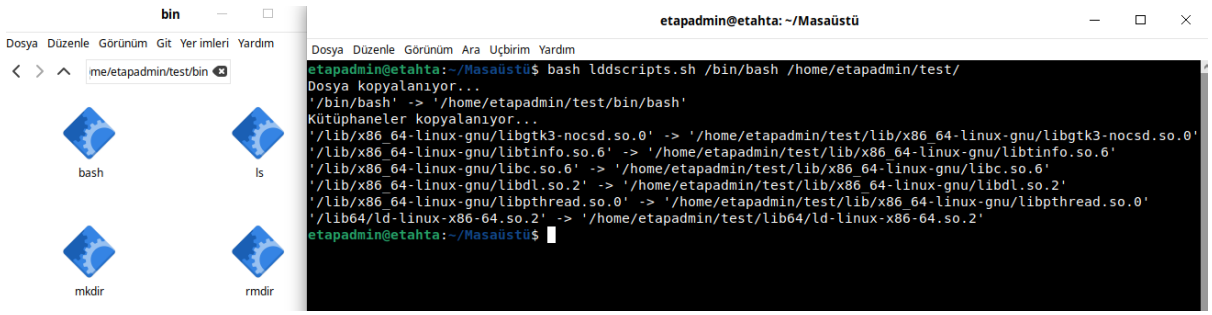
mkdir Komutu

```
bash lddscripts.sh /bin/mkdir /home/etapadmin/test/
```



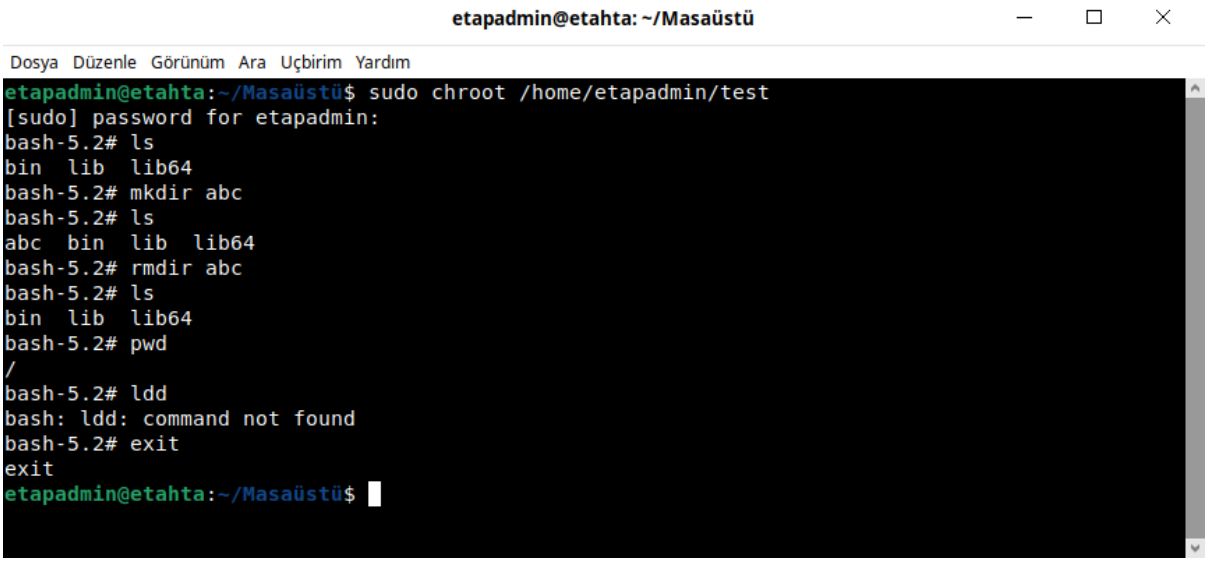
bash Komutu

```
# bash komutu ve bağımlılığı kopyalandı.
bash lddscripts.sh /bin/bash /home/etapadmin/test/
```



chroot Sistemde Çalışma

```
sudo chroot /home/etapadmin/test komutunu kullanmalıyız.
```



- **abc** dizini oluşturuldu, **abc** dizini silindi, **pwd** komutuyla konum öğrenildi, **ldd** komutu sistemimizde olmadığından hata verdi.
- Çıkış için ise **exit** komutu kullanılarak sistemden çıkıldı.

Kaynak:
https://stackoverflow.com/questions/64838052/how-to-delete-n-characters-appended-to-ldd-list
https://app.diagrams.net/

Kaynak Kod Derleme

Bir uygulamanın kodları genellikle çalışmaz(python benzeri kodlar istisna). Bu kodlardan sistemlerin çalışması için çalışabilir dosyalar üretilir(linuxta ikili dosya, elf, windowsta exe, com vb.). Bu çalışabilir dosyaları koddan oluştururken iki faklı şekilde oluşturabiliriz.

- **Paylaşımlı Derleme(dynamic):** Kendine lazım olan kütüphaneleri sistem üzerindeki başka uygulamalarla ortak kullanır.
- **Paylaşımsız, gömülü(static):** Kendisine lazım olan kütütphaneleri kendi içinde barındırır(porable uygulama gibi).

Şimdi aşağıdaki kaynak kodumuzu iki farklı yöntemle derleyelim.

```
//main.c dosyamız
#include <stdio.h>
void main(){
    printf("Merhaba\n");
}
```

1-Paylaşımlı Derleme(dynamic):

Derlenen uygulama sistemde bulunan kütüphaneleri kullanacak şeklide derlenmesidir. Uygulama boyutu küçüktür, taşınabirliği sınırlanabilir. Aşağıdaki gibi derlenir.

```
gcc -o main main.c
```

main.c kodumuzu **main** adında ikili çalışabilir dosyaya dönüşütürdük. **ldd** komutuyla **main** dosyasının kullandığı kütüphaneler öğrenilir.

```
unset LD_PRELOAD
ldd ./main
        linux-vdso.so.1 (0x00007ffdb3bb9000)
        libc.so.6 => /lib/x86_64-linux-gnu/libc.so.6 (0x00007f2c53fe0000)
        /lib64/ld-linux-x86-64.so.2 (0x00007f2c541e7000)
```

Burada **libc.so.6** ve **ld-linux-x86_64.so.2** dosyaları **glibc** tarafından sağlanır. Derlenmiş dosyanın çalışması için tüm bağımlılıklarının sistemde bulunması gereklidir.Sadece gerekli olan kütüphaneleri görmek için **readelf -d** komutu kullanılabilir. Aşağıda gerekli olan kütüphaneler listeleniyor.

```
readelf -d ./main | grep -i needed
       0x0000000000000001 (NEEDED)             Paylaşımlı kitaplık: [libc.so.6]
```

ldconfig

Sistemdeki kütüphanelerin konumlarını **/etc/ld.so.conf** dosyasına bakarak belirler.

```
#/etc/ld.so.conf dosyası
/usr/lib64
/usr/lib
/lib64
/lib
```

Kütüphanelerde değişiklik yapılmışsa ve hemen bu değişikliği sistemin görmesini istersek **ldconfig** komutu kullanılmalıdır.

2-Paylaşımsız, gömülü(static):

Derlenen uygulama sistemde bulunan ve çalışması için gerekli olan kütüphaneleri uygulama içine dahil eden bir derleme yöntemidir. Uygulamamızı static derlemek için **-static** parametresi ekleyerek derlenir.

```
gcc -o main main.c -static
```

Paylaşımlı(dynamic) derleme işleminde bağımlı olduğu dosyaları **ldd** komutunu kullanarak öğrenmiştik. Şimdi paylaşımsız derlediğimiz **main** dosyasında bağımğımlı olduğu kütüphaneleri kontrol ediyoruz.

```
ldd main
    not a dynamic executable
```

Bağımlı kütüphaneler yerine **not a dynamic executable** mesajı gördük. Bunun anlamı çalışması için hiçbir kütüphaneye ihtiyaç duymaz. Bu bir avantaj ve taşınabirliği artırır. Devevantajı ise boyutu büyük olur. İhtiyaca göre paylaşımlı veya paylaşımsız derleme tercih edilir.

Derleme Araçları

Linux sistemlerinde kodlarımızı derlemek kullanılan uygulamalara derleme araçları denir. Birden fazla araç olsada en temel derleme araçları **make**, **cmake**, **meson**, **python**.

1-make

make, yazılım geliştirme süreçlerinde sıkça kullanılan bir araçtır. Özellikle C ve C++ gibi dillerde projelerin derlenmesi ve yönetilmesi için kullanılır.

Aşağıdaki C kodumuzu(merhaba.c dosyası) make ile nasıl derleyeceğimizi anlatalım;

```
#include <stdio.h>
int main(){
    puts("Merhaba");
    return 0;
}
```

Makefile Nedir?

Makefile, make aracının nasıl çalışacağını belirten bir dosyadır. İçinde hedefler, bağımlılıklar ve komutlar bulunur. Örneğin, bir C programını derlemek için aşağıdaki gibi basit bir Makefile oluşturabilirsiniz. Makefile ve merhaba.c dosyası aynı konumda olmalıdır.

```
CC=gcc
CFLAGS=-I.
all: program
program: merhaba.o
        $(CC) -o program merhaba.o
merhaba.o: merhaba.c
        $(CC) -c merhaba.c $(CFLAGS)
clean:
        rm -f *.o program
```

merhaba dosyasından **program** adında bir ikili dosya oluşturmak için aşağıdaki komutlar Makefile ve merhaba.c dosyasının olduğu konumda çalıştırılır.

```
make
make install
```

Genellikle Makefile dosyası olmaz. Onun yerine **configure.ac** dosyası olur. **configure.ac** dosyasından **Makefile** dosyası elde etmek için, öncelikle **autoconf** ve **automake** araçlarının sisteminizde kurulu olması gerekmektedir.

autoreconf -fvi komutu çalıştırılıarak configure dosyamızı üretir.Bu araçlar, yapılandırma dosyalarınızı işleyerek gerekli **Makefile** dosyalarını oluşturmanıza yardımcı olur.

configure komutunun devamında **--prefix** dışında başka parametreleride olabilir. Bu paramatreleri "Read.me" dosyası içerisinde bulunabilir veya **configure --help** komutu kaynak kodların olduğu konumda kullanılarak görülebilir. Bu kaynak kod aşağıdaki gibi derlenir.

```
$ autoreconf -fvi
$ ./configure --prefix=/usr
$ make
$ make install
```

2-cmake

CMake, projelerin derlenmesi ve yapılandırılması için kullanılan bir araçtır. Bir paketi CMake ile derlemek için genellikle aşağıdaki adımları izleriz:

İlk olarak, projenin kök dizininde bir CMakeLists.txt dosyası oluşturun. Ardından, CMake komutunu kullanarak projeyi yapılandırın ve derleyin. Örneğin:

```
    mkdir compile_packages
    cd compile_packages
    cmake ..
    make

Bu adımları takip ederek, CMake ile paketinizi başarıyla derleyebilirsiniz.
```

CMake'in esnekliği ve kullanım kolaylığı sayesinde paketlerin derlenmesi ve yapılandırılması oldukça kolay hale gelir.

Bu kaynak kod aşağıdaki gibi derlenir:

```
mkdir compile_packages
cd compile_packages
cmake ..
make
make install
```

3-meson

Meson, modern bir yapılandırma betik dili ve Ninja ise hızlı bir derleyici aracıdır. Bir paketi derlemek için öncelikle Meson ile yapılandırma dosyalarını oluşturmanız gerekir. Ardından, Ninja derleyici aracını kullanarak bu yapılandırmayı derleyebilirsiniz.

İlk olarak, projenizin dizinine gidin ve Meson ile yapılandırma dosyalarını oluşturun:

```
meson setup builddir --prefix=/usr
```

Sonra, oluşturulan builddir dizinine geçin ve Ninja ile derlemeyi başlatın:

```
ninja -C builddir
```

Bu adımları takip ederek Meson ve Ninja kullanarak paketinizi başarılı bir şekilde derleyebilirsiniz.

Bu kaynak kod aşağıdaki gibi derlenir:

```
# paketin yükleneceği konum
INSTALL_ROOT=/
meson setup builddir --prefix=/usr
ninja -C builddir
INSTALL_ROOT=$INSTALL_ROOT ninja -C builddir install
```

4-python

Python ile paket derlemek için genellikle **setuptools** ve **distutils** gibi kütüphaneler kullanılır. Öncelikle, projenizin kök dizininde bir **setup.py** dosyası oluşturmanız gerekir. Bu dosya, paketin yapılandırmasını ve gereksinimlerini tanımlar.

Örnek bir setup.py dosyası aşağıdaki gibi olabilir:

```
from setuptools import setup

setup(
    name='paket_adi',
    version='1.0',
    packages=['paket'],
    install_requires=[
        'bağımlılık_paketi',
    ],
)
```

Ardından, terminalde projenizin bulunduğu dizine giderek aşağıdaki komutu çalıştırarak paketi derleyebilirsiniz:

```
# yükleme konumu
INSTALL_ROOT=/
python3 -m compile_packages
python3 -m installer --destdir="$INSTALL_ROOT" dist/*.whl
```

Bu komut, paketi dist klasörüne derleyecektir. Artık paketiniz hazır ve dağıtıma uygun hale gelmiştir.

OpenRC

Openrc sistem açılışında çalışacak uygulamaları çalıştıran servis yöneticisidir.

Çalıştırılması

Openrc servis yönetiminin çalışması için boot parametrelerine yazılması gerekmektedir. **/boot/grub.cfg** içindeki **linux /vmlinuz init=/usr/sbin/openrc-init root=/dev/sdax** olan satırda **init=/usr/sbin/openrc-init** yazılması gerekmektedir. Artık sistem openrc servis yöneticisi tarafından uygulamalar çalıştırılacak ve sistem hazır hale getirilecek.

openrc Kullanımı

Servisleri etkinleştirip devre dışı hale getirmek için **rc-update** komutu kullanılır. Aşağıda **udhcpc** internet servisi örnek olarak gösterilmiştir. **/etc/init.d/** konumunda **udhcpc** dosyamızın olması gerekmektedir.

```
# servis etkinleştirmek için
rc-update add udhcpc boot
# servisi devre dışı yapmak için
rc-update del udhcpc boot
# Burada udhcpc servis adı, boot ise runlevel adıdır.
```

Elle **/etc/runlevels/** altında bulunan **boot**, **default**, **nonetwork**, **shutdown**, **sysinit** dizinlerine **/etc/init.d/** dizininin altındaki dosyaları **ln** komutuyla kısayol yaparsak **rc-update add** komutunun yaptığı görevi yapmış oluruz. Kısayolu silersek **rc-update del** komutunun görevini yapmış oluruz.

/etc/runlevels/ altında bulunan **boot**, **default**, **nonetwork**, **shutdown**, **sysinit** dizinler servis dosyalarımızın hangi sırayla çalışacağını belirleyen dizinlerdir. Mesela **boot** dizini ilk açılış sırasında çalışacak olan servis dosyalarının konulacağı dizindir.

Servisleri başlatıp durdurmak için ise **rc-service** komutu kullanılır.

```
rc-service udhcpc start
# veya şu şekilde de çalıştırılabilir.
/etc/init.d/udhcpc start
```

Servislerin durumunu öğrenmek için **rc-status** komutu kullanılır. Ayrıca sistemdeki servislerin sonraki açılışta hangisinin başlatılacağını öğrenmek için parametresiz olarak **rc-update** kullanabilirsiniz.

```
# şu an hangi servislerin çalıştığını gösterir
rc-status
# sonraki açılışta hangi servislerin çalışacağını gösterir
rc-update
```

Sistemi kapatmak veya yeniden başlatmak için **openrc-shutdown** komutunu kullanabilirsiniz.

```
openrc-shutdown -p 0     # kapatmak için
openrc-shutdown -r 0     # yeniden başlatmak için
```

Servis Dosyası

Openrc servis dosyaları basit birer **bash** betiğidir. Bu betikler **openrc-run** komutu ile çalıştırılır ve çeşitli fonksiyonlardan oluşabilir. Servis dosyaları **/etc/init.d** içerisinde bulunur. Servisleri ayarlamak için ise **/etc/conf.d** içerisine aynı isimle ayar dosyası oluşturabiliriz.

Çalıştırılacak komut, komut parametreleri ve **pidfile** dosyamızı aşağıdaki gibi belirtebiliriz.

```
description="Ornek servis"
command=/usr/bin/ornek-servis
command_args=--parametre
pidfile=/run/ornek-servis.pid
```

Bununla birlikte **start**, **stop**, **status**, **reload**, **start_pre**, **stop_pre** gibi fonksiyonlar da yazabiliriz.

```
start(){
    ebegin "Starting ${RC_SVCNAME}"
    start-stop-daemon --start --pidfile "/run/servis.pid" --exec /usr/bin/ornek-servis --parametre
}
```

Servis bağımlılıklarını belirtmek için ise **depend** fonksiyonu kullanılır.

```
depend() {
  need localmount
  after dbus
}
```

Qemu Kullanımı

qemu açık kaynaklı Virtual Box, Vmware benzeri sanallaştırma aracıdır.

Sisteme Kurulum

```
sudo apt update
sudo apt install qemu-system-x86  qemu-utils
```

qemu Kullanımı

```
# 30GB disk oluşturuldu.
qemu-img create disk.img 30G
qemu-system-x86_64 --enable-kvm -hda disk.img -m 2G -cdrom etahta.iso
# qemu-system-x86_64 --enable-kvm -hda disk.img -m 2G #sadece disk ile çalıştırılıyor
# qemu-system-x86_64 -m 2G -cdrom etahta.iso          #sadece iso dosyası ile çalıştırma
```

Sistem Hızlandırılması

--enable-kvm eğer sistem disk ile çalıştırıldığında bu parametre eklenmezse yavaş çalışacaktır.

Boot Menu Açma

Sistemin diskten mi imajdan mı başlayacağını başlangıçta belirlemek için boot menu gelmesini istersek aşağıdaki gibi komut satırına seçenek eklemeliyiz.

```
qemu-system-x86_64 --enable-kvm -cdrom distro.iso -hda disk.img -m 4G -boot menu=on
```

Uefi kurulum için:

sudo apt-get install ovmf

qemu-system-x86_64 --enable-kvm -bios /usr/share/ovmf/OVMF.fd -cdrom distro.iso -hda disk.img -m 4G -boot menu=on

qemu Host Erişimi:

İstemci bilgisayar ip'si:10.0.2.15 ve ana bilgisayar 10.0.0.2 olarak ayarlıyor.

vmlinuz ve initrd

qemu ile vmlinuz ve initrd.img dosyaları iso olmadan test edilebilir.

```
qemu-system-x86_64 --enable-kvm -kernel /boot/vmlinuz-5.17 -initrd /home/deneme/initrd.img -append "quiet" -m 512m
```

qemu Terminal Yönlendirmesi

```
qemu-system-x86_64 --enable-kvm -kernel vmlinuz -initrd initrd.img -m 3G -serial stdio -append "console=ttyS0"
```

Diskteki Sistemin Açılışını Terminale Yönlendirme

```
qemu-system-x86_64 -nographic -kernel boot/vmlinuz -hda disk.img -append console=ttyS0
```

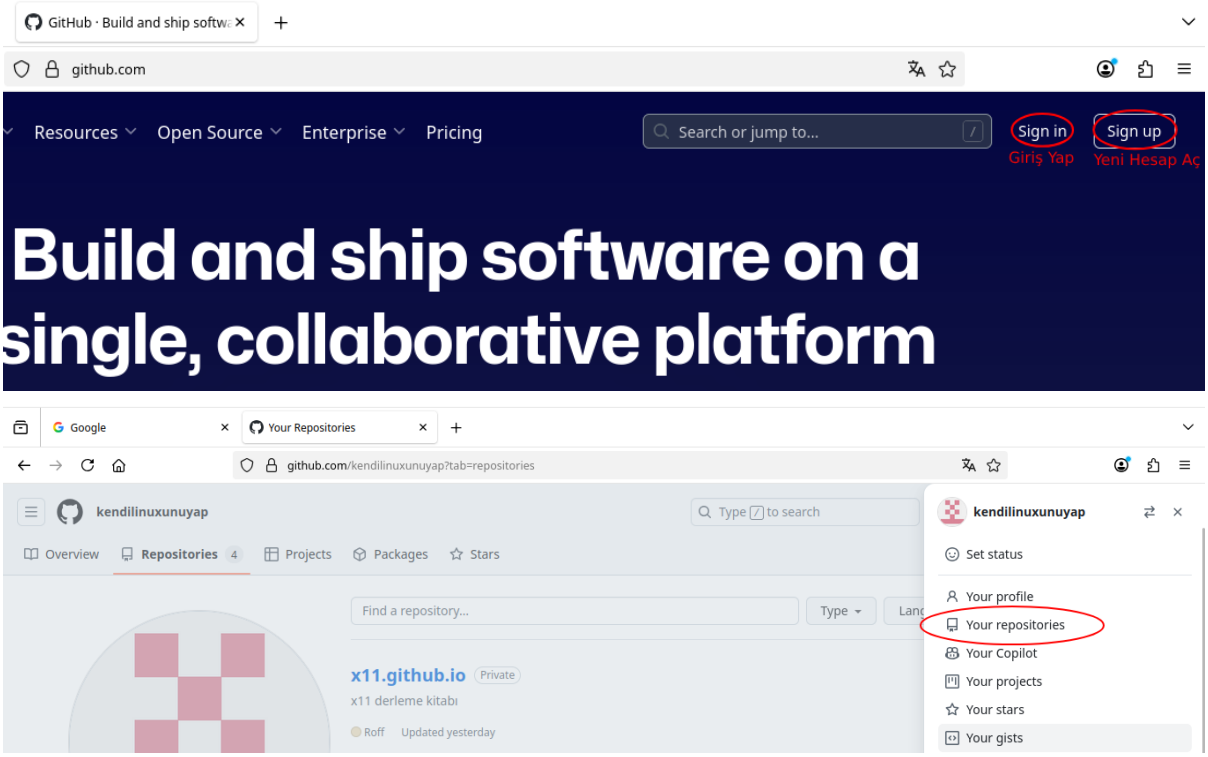
Kaynak: | https://www.ubuntubuzz.com/2021/04/how-to-boot-uefi-on-qemu.html

github

GitHub üzerinde yeni bir depo açmak oldukça basit bir işlemdir. Aşağıdaki adımları takip ederek hızlıca kendi deponuzu oluşturabilirsiniz:

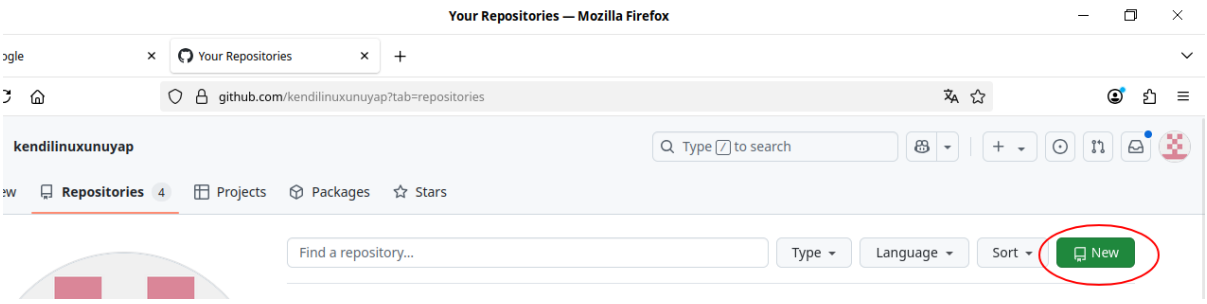
GitHub Hesabınıza Giriş Yapın

GitHub ana sayfasına gidin ve hesabınıza giriş yapın. Eğer bir hesabınız yoksa, öncelikle bir hesap oluşturmalısınız.



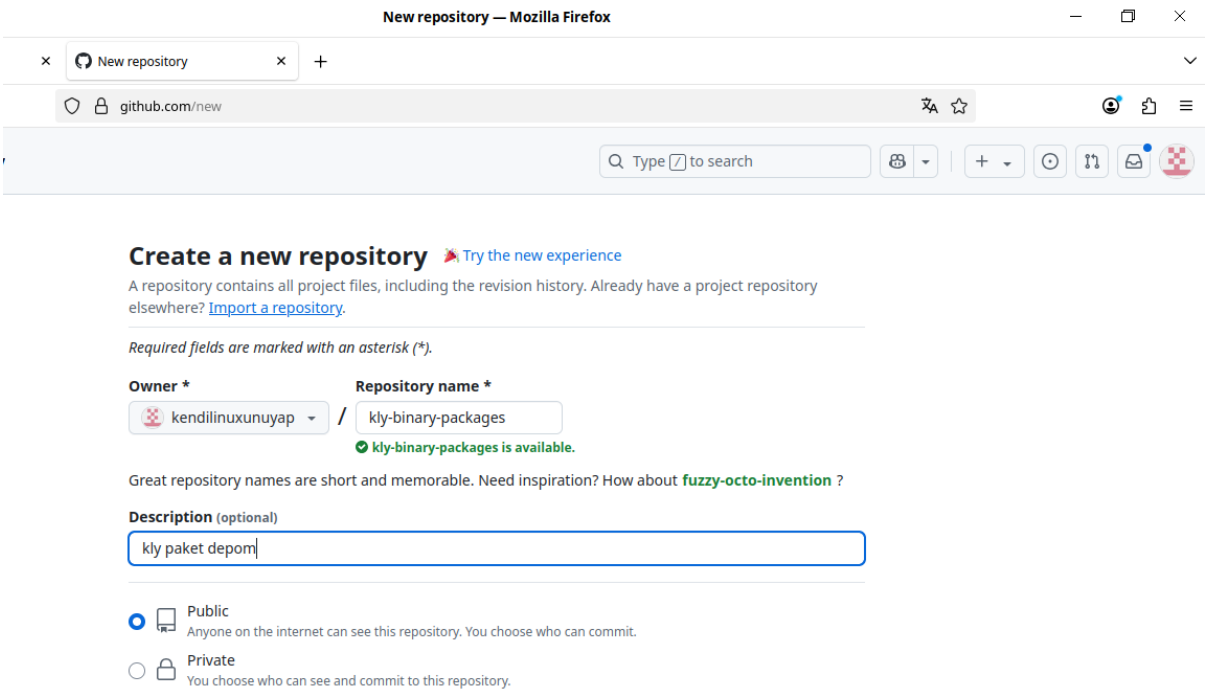
Yeni Depo Oluşturma

Sağ üst köşede bulunan "+" simgesine tıklayın ve "New repository" seçeneğini seçin.



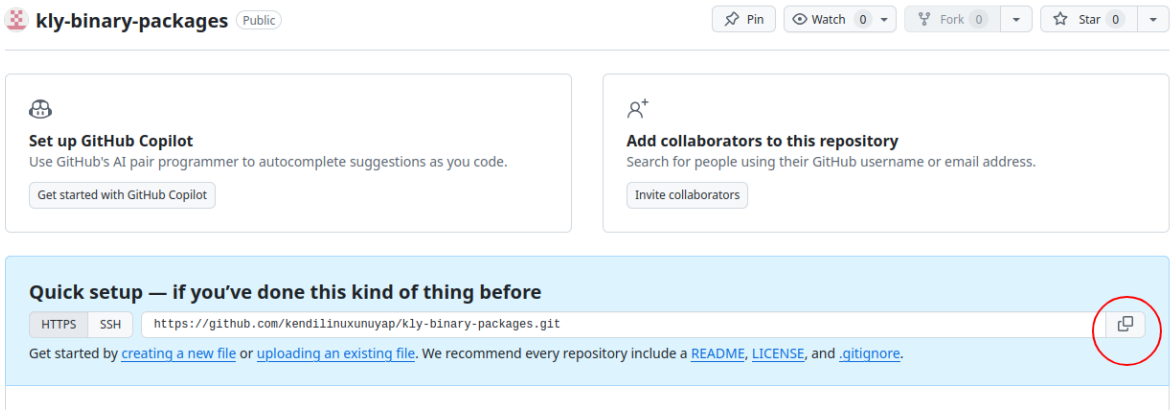
Depo Bilgilerini Girin

Açılan sayfada, depo adını (repository name) ve isteğe bağlı olarak bir açıklama (description) girin. Depo özel (private) veya herkese açık (public) olarak ayarlanabilir.



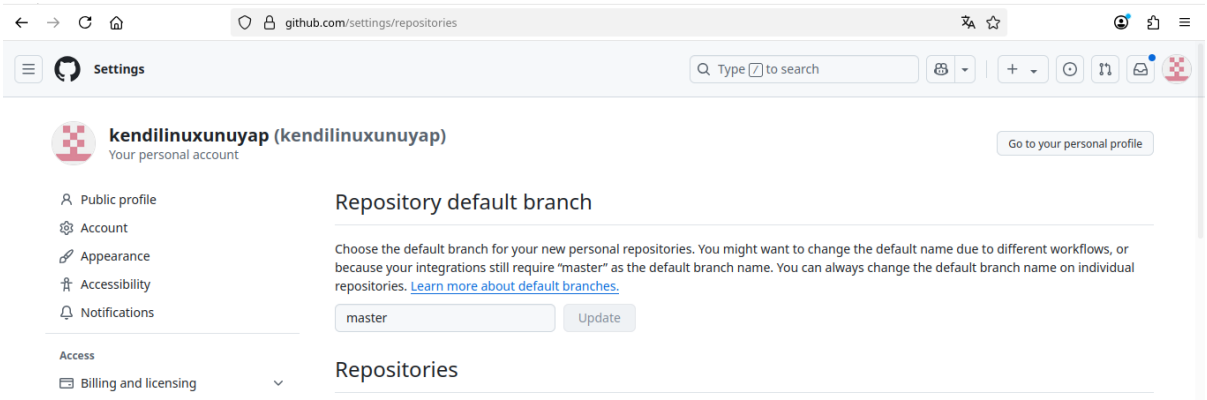
İlk Dosyayı Oluşturma

"Initialize this repository with a README" seçeneğini işaretleyerek, depo oluşturulduğunda otomatik olarak bir README dosyası oluşturabilirsiniz.

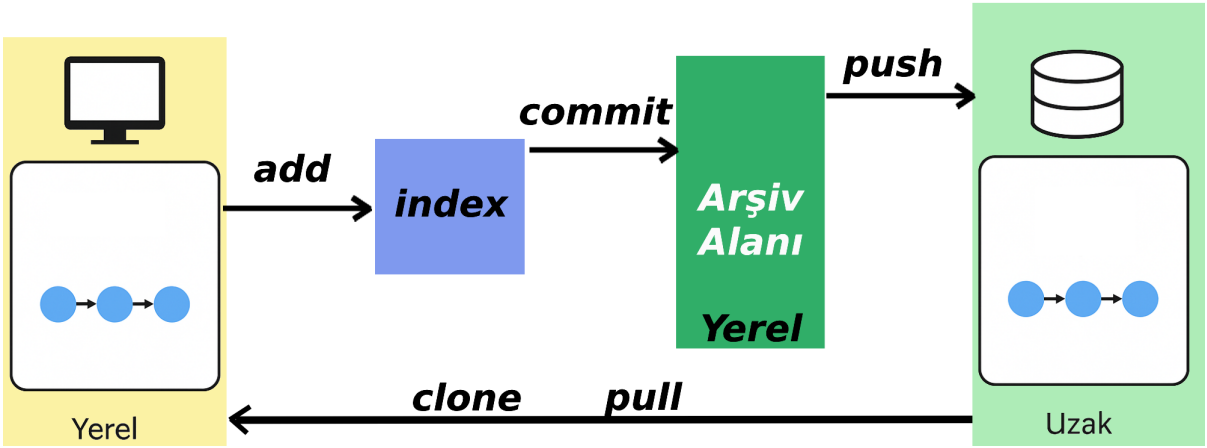


github Varsayılan Dal Ayarı:

githubda varsayılan olarak eski sürümlerde master, yeni sürümlerde main kullanılmaktadır. Projelerimizde ve burada kullanılan yapılarda **master** kullanıldığı için aşağıda görülduğü gibi varsalılan dalı **master** yapıyoruz.



github komut Kullanımı



github'ın çalışma mantığı yukarıda verilen resimde görüldüğü gibidir. Komutları kullanırken resimdeki gibi işlemleri yapmalıyız.

github'a göndermek için; **add --> commit --> push** kullanmalıyız.

github'dan indirmek için; **clone veya pull** kullanmalıyız.

Sık Kullanılan github Komutları

Depoyu Yerele indirme(Clone)

```
git clone https://github.com/kullaniciadi/depoadi.git
```

- **Yerel Depoda Dosya Ekleme:**

```
git add .
```

- **Yerel Depoda Değişiklik Etiketi Yapma:**

```
git commit -m "ilk adım"
```

- **Yerel Depodaki Bilgileri Guthuba Gönderme:**

```
git push origin master
```

- **Yerel Depodaki Bilgileri Guthuba Gönderme Reddedilirse:**

```
git push origin master --force
```

- **Yerelde github'daki Depoyu clone Yapmadan Oluşturma:**

```
cd proje
git init
git config --global user.name "name"
git config --global user.email "name@gmail.com"
git add README.md
git add .
git commit -m "first commit"
git remote -v      # push ve pull yapılacak adresleri görmek için kullanılır
git remote add origin https://github.com/userName/repoName.git
git push -u origin master
```

Dal(Branch):

Dal projenin birden fazla kişi ile yapılmasında veya yeni özellikler eklenmek istediğinde projenin bir kopyası ile çalışma gerektirir. Aşağıda dal işlemleri için komutlar verilmiştir.

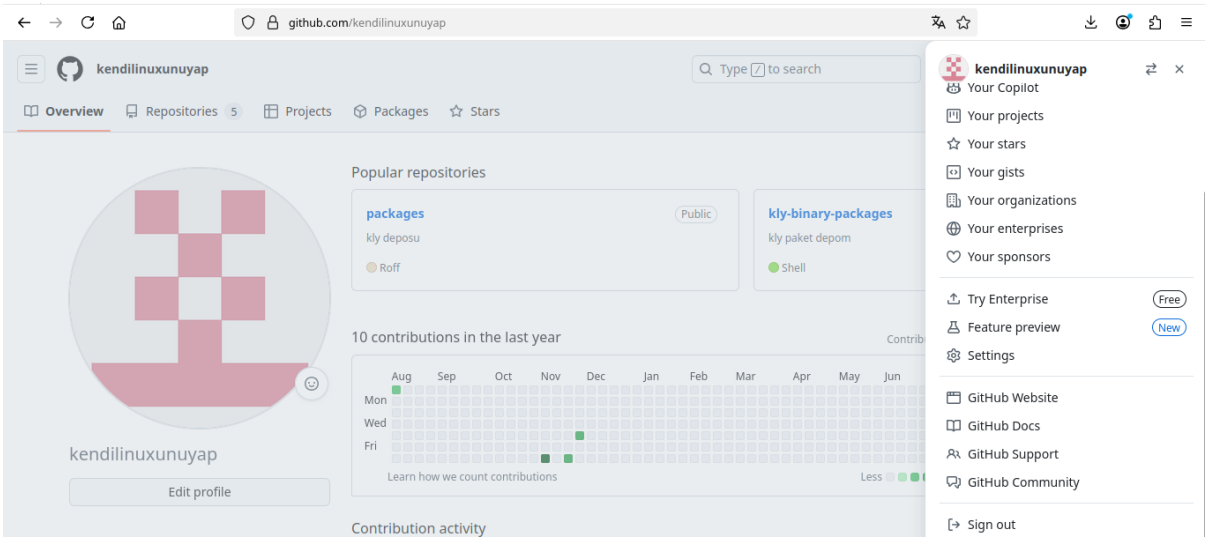
Yeni Dal Oluşturmak, Seçmek ve Yeni Dalı github'a Göndermek:

```
# Dalları Görmek kullanılır Seçili olan dalın rengi farklı olur ve önünde * olur
git branch
# Dal oluşturmak
git checkout yeni
# Yeni dalı uzak adrese göndermek
git push --force origin master komutu  yerine
# Uzak adresimizde master dalı dışında yeni dalımızda oluşacaktır...
git push --force origin yeni komutunu veriyoruz.
```

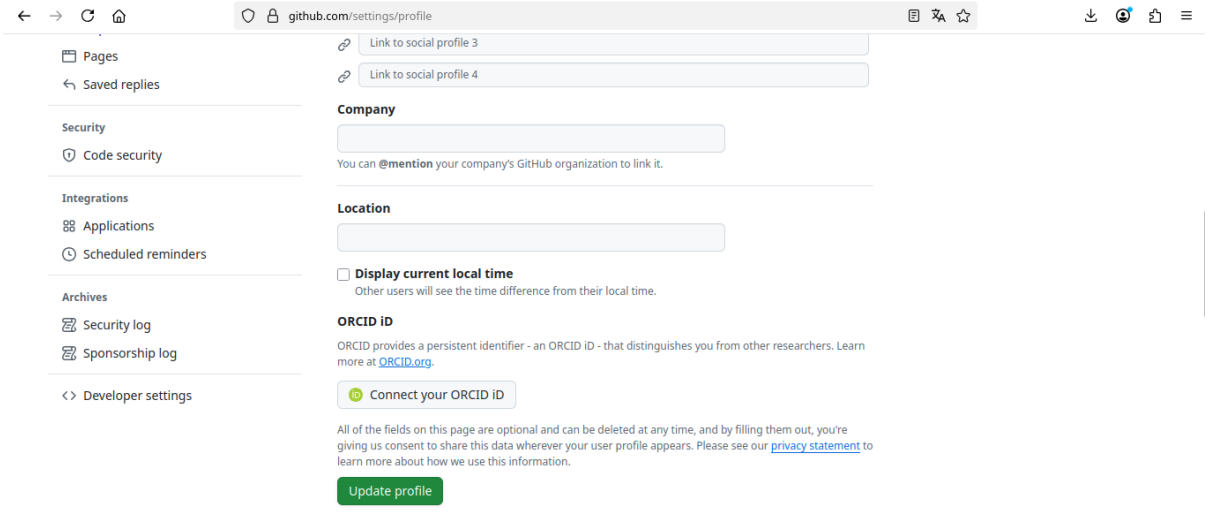
github token Oluşturma Kullanma

github kullanırken dosya gondermek(**commit**) için kullanıcı adı ve parola ister. Burada hesap parolası yerine **token** kullanılır. Yeni bir **token** için aşağıdaki işlem adımlarını yapmalıyız.

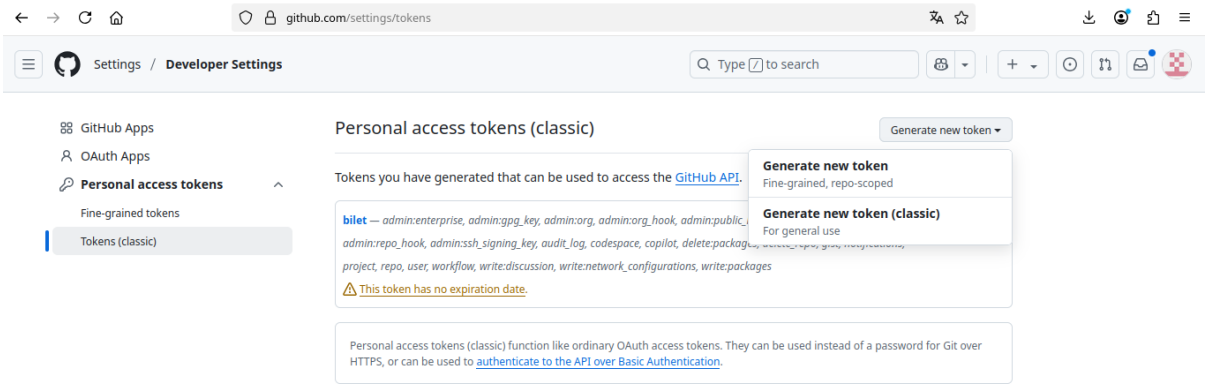
Settings Seçilir;



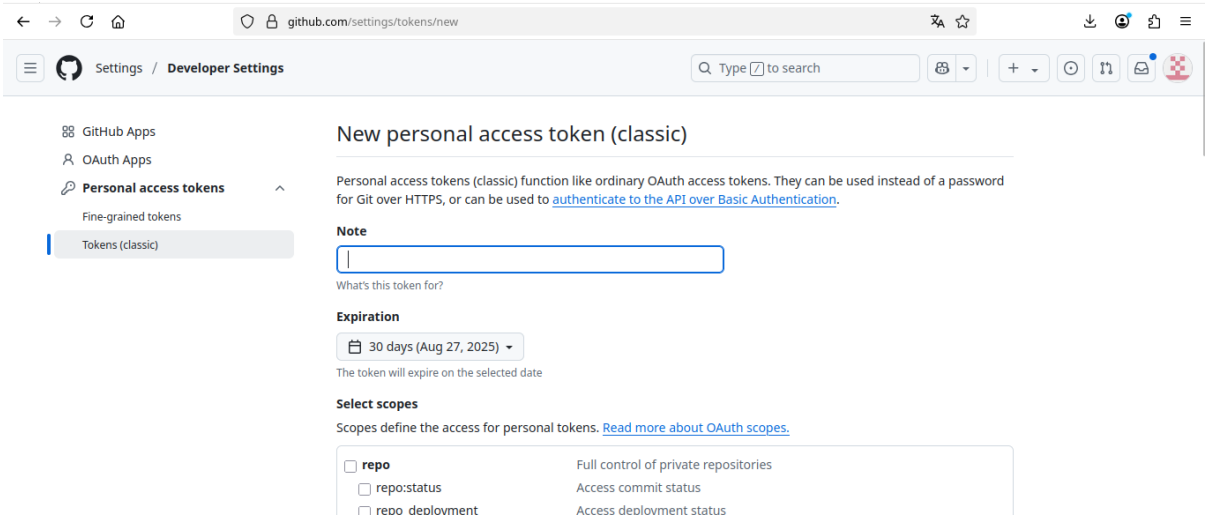
Developper Settings Seçilir;



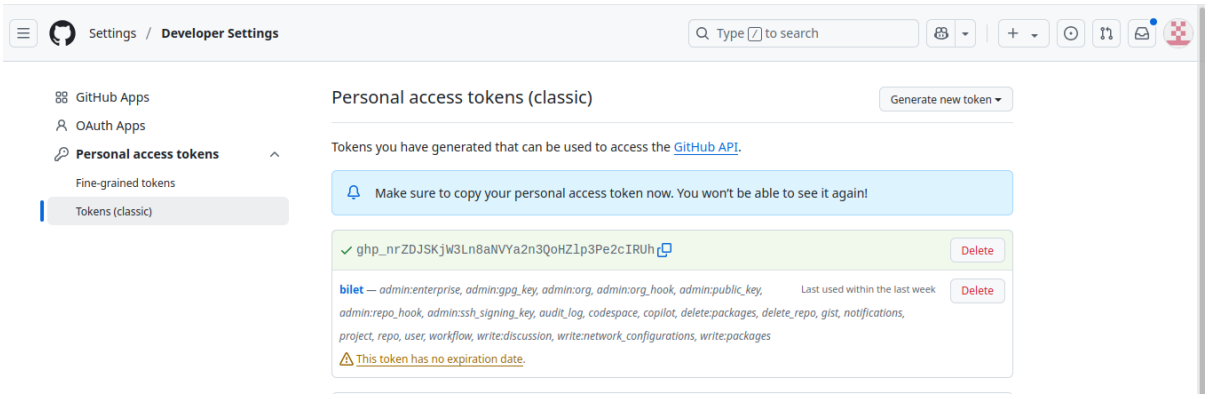
Generate New Token Seçilir;



Erişim yapabileceği alanlar Seçilir;



token Kullanmak üzere kullanmak üzere saklanır. Bu ekrandan sonra sadece silebiliriz. Göremeyiz kopyalayamayız.



elogind Yapılandırması

Bu belge, **systemd kullanmayan** sistemlerde (OpenRC tabanlı) elogind login manager'ın kurulumu, yapılandırılması ve gerekli kullanıcı/PAM/agetty ayarlarını anlatır.

1. Gerekli Kullanıcı ve Grup Tanımları

elogind servisi için sistemde gerekli kullanıcı ve gruplar oluşturulur:

```
groupadd -g 190 systemd-journal
groupadd -g 191 elogind
useradd -r -s /sbin/nologin -g elogind -u 191 elogind
```

Açıklama:

- *systemd-journal* grubu, journal log erişimi için kullanılır.
- *elogind* kullanıcı ve grubu, servis için ayrılmıştır.
- */sbin/nologin* ile bu kullanıcıyla doğrudan login engellenir.
- *-r* parametresi sistemi kullanıcı oluşturur, yani normal login için kullanılmaz.

2. OpenRC Hizmet Dosyası (/etc/init.d/elogind)

OpenRC ile *elogind* servisini yönetmek için basit bir init script örneği:

```
#!/sbin/openrc-run
description="elogind login manager"
command=/usr/lib/elogind/elogind
pidfile=/run/elogind.pid

depend() {
    need dbus
    after dbus
}
```

Açıklama:

- *command*: elogind daemonunu başlatır.
- *pidfile*: servis PID bilgisini saklar.
- *depend()*: servis başlatılmadan önce D-Bus'ın aktif olmasını garanti eder.

3. Başlatma ve OpenRC'ye Ekleme

```
rc-update add elogind default
rc-service elogind start
```

Açıklama:

- *rc-update add*: Servisi boot sırasında otomatik başlatmaya ekler.
- *rc-service start*: Servisi hemen başlatır.
- **Önemli Kontrol:** elogind'in çalışıp çalışmadığını kontrol etmek için:

```
loginctl list-sessions
```

- Eğer aktif oturumlar listeleniyorsa, elogind başarıyla çalışıyor demektir.

4. PAM, Passwd ve Group Ayarları

/etc/nsswitch.conf Örneği:

```
passwd:      files
group:       files
shadow:      files
```

Açıklama:

- Kullanıcı ve grup bilgileri *files* (yerel /etc/passwd ve /etc/group) üzerinden çözülür.

/etc/pam.d/system-auth Örneği:

```
auth        required      pam_unix.so
account     required      pam_unix.so
password    required      pam_unix.so
session     required      pam_unix.so
session     include       elogind-user
```

> Note
>
> **`session include elogind-user` satırı mutlaka ekli olmalıdır.** Bu satır, kullanıcı oturumlarının elogind ile doğru şekilde yönetilmesini sağlar. Eğer yoksa manuel olarak eklenmelidir:
>
> ```
> sed -i "/elogind-user/d" /etc/pam.d/system-auth
> echo -e "\nsession    include    elogind-user" >> /etc/pam.d/system-auth
> ```

5. OpenRC Agetty Ayarı

Bazı durumlarda *agetty*, varsayılan olarak */bin/login* kullanmaz ve bu nedenle elogind kullanıcı oturumunu başlatamaz.

Ayar:

```
sed -i "/agetty_options/d" /etc/conf.d/agetty
echo -e "\nagetty_options=\"-l /usr/bin/login\"" >> /etc/conf.d/agetty
```

Açıklama:

- *-l /usr/bin/login* ile agetty doğru login programını çağırır.
- Bu ayar, sanal konsollardan kullanıcı girişlerinin doğru şekilde yapılmasını garanti eder.

6. Diğer Öneriler ve Notlar

- *elogind*, sistemde */run/systemd* gibi dizinler oluşturur; böylece bazı uygulamalar systemd'ye ihtiyaç duymadan çalışabilir.
- Polkit yetkilendirme sorunları genellikle kullanıcıların *wheel* grubuna dahil olmamasıyla ilişkilidir.
- OpenRC + elogind ile systemd'ye gerek kalmadan oturum yönetimi sağlanabilir.

Kaynaklar:

```
- elogind resmi dokümanı: https://www.freedesktop.org/wiki/Software/systemd/elogind/
- OpenRC belgeleri: https://wiki.gentoo.org/wiki/OpenRC
- PAM belgeleri: https://www.linux-pam.org/
- Polkit belgeleri: https://www.freedesktop.org/wiki/Software/polkit/
```

D-Bus Yapılandırması

Bu belge, **systemd kullanmayan** sistemlerde (OpenRC tabanlı) D-Bus servisinin kurulumu, yapılandırılması ve başlatılmasını anlatır. D-Bus, Linux ortamında uygulamalar arasında mesajlaşma sağlayan bir IPC (Inter-Process Communication) sistemidir.

Adım 1: Kullanıcı ve Grup Oluşturma

```
echo "messagebus:x:109:" >> /etc/group
echo "messagebus:x:103:109::/nonexistent:/usr/sbin/nologin" >> /etc/passwd
```

Adım 2: Sistem UUID ve machine-id Ayarı

```
if [ ! -f /etc/machine-id ] ; then
    dbus-uuidgen --ensure=/etc/machine-id
fi
```

Açıklama:

```
- `/etc/machine-id` dosyası sistemin benzersiz kimliğini tutar.
- `dbus-uuidgen --ensure` komutu, dosya yoksa oluşturur, varsa korur.
- Bu kimlik, D-Bus servislerinin ve bazı uygulamaların doğru çalışması için zorunludur.
```

Adım 3: dbus-daemon-launch-helper İzin Kontrolü

```
if busybox ls /bin/su -la | grep "^...s" >/dev/null ; then
    chmod 4755 /usr/libexec/dbus-daemon-launch-helper
fi
```

Açıklama:

```
- `dbus-daemon-launch-helper`, kullanıcı bazlı D-Bus daemonlarını başlatır.
- SUID bitinin (4) set edilmesi, root yetkisi ile çalışmasını sağlar.
```

Adım 4: D-Bus Yapılandırma Yenileme

```
dbus-send --system --type=method_call --dest=org.freedesktop.DBus / \
    org.freedesktop.DBus.ReloadConfig >/dev/null 2>&1 || :
```

Açıklama:

```
- Bu komut, D-Bus sistem servisine yeni yapılandırmaları yüklemesini söyler.
- `--system` parametresi sistem bus'ını hedef alır.
- Hata oluşursa komut sessizce geçer (`|| :`).
```

Adım 5: D-Bus Servisini Başlatma (OpenRC)

D-Bus için OpenRC init script'i */etc/init.d/dbus* şu mantıkla çalışır:

```
#!/sbin/openrc-run
#-----------------------------------------------------------------------------------------------
name="System Message Bus"
description="An IPC message bus daemon"

extra_started_commands="reload"

supervisor=supervise-daemon
command="/usr/bin/dbus-daemon"
command_args="--system --nofork --nopidfile --syslog-only ${command_args:-}"
command_background="yes"
pidfile="/run/$RC_SVCNAME.pid"

depend() {
    need localmount
    after bootmisc
}

start_pre() {
    mkdir -p /run/dbus
    /usr/bin/dbus-uuidgen --ensure=/etc/machine-id
}

stop_post() {
    [ ! -S /run/dbus/system_bus_socket ] || rm -f /run/dbus/system_bus_socket
}

reload() {
    ebegin "Reloading $name configuration"
    /usr/bin/dbus-send --print-reply --system --type=method_call \
            --dest=org.freedesktop.DBus \
            / org.freedesktop.DBus.ReloadConfig > /dev/null
    eend $?
}
```

Açıklama:

```
- `supervise-daemon`: OpenRC'nin daemon'u denetlemesini sağlar.
- `command` ve `command_args`: D-Bus daemonunu sistem bus modunda başlatır.
- `start_pre()`: Servis başlamadan önce gerekli dizini oluşturur ve machine-id'yi doğrular.
- `stop_post()`: Servis durduğunda socket dosyasını temizler.
- `reload()`: Yapılandırma değişikliklerini uygulamak için D-Bus'a mesaj yollar.
- `depend()`: Servisin başlatılması için gerekli bağımlılıklar (localmount, bootmisc) belirtilir.
- `pidfile`: Servisin PID bilgisini tutar.
```

Servis Dosyasının Yaptıklar:

```
- D-Bus için sistem kullanıcı ve grup oluşturulur.
- machine-id ve gerekli izinler ayarlanır.
- OpenRC init script'i `/etc/init.d/dbus`, servisin başlatılmasını, durdurulmasını ve yeniden yüklenmesini yönetir.
- `rc-service dbus start` ile sistemde mesajlaşma servisi aktif olur.
```

Kaynaklar:

```
- D-Bus resmi sitesi:https://www.freedesktop.org/wiki/Software/dbus/
- D-Bus belgeleri:https://dbus.freedesktop.org/doc/dbus/
- OpenRC belgeleri:https://wiki.gentoo.org/wiki/OpenRC
- dbus-daemon ve dbus-send man sayfaları:https://manpages.debian.org/dbus-user-session
```

Kaynaklar

- https://tr.wikipedia.org/wiki/Linux - 02/07/2025
- https://www.subrat.info/build-kernel-and-userspace/ - 08/07/2025
- https://medium.com/@chienhaotan/compiling-and-running-a-minimal-kernel-with-busybox-bfc45a991017 - 08/07/2025
- https://stackoverflow.com/questions/64838052/how-to-delete-n-characters-appended-to-ldd-list - 20/06/2025
- https://gist.github.com/bluedragon1221/a58b0e1ed4492b44aa530f4db0ffef85 - 09/07/2025
- https://app.diagrams.net/
- https://www.ubuntubuzz.com/2021/04/how-to-boot-uefi-on-qemu.html - 20/06/2024
- https://wiki.gentoo.org/wiki/OpenRC - 30/06/2025
- https://busybox.net/ - 01/07/2025
- https://busybox.net/about.html - 01/07/2025
- https://sulincix.gitlab.io/distro-kitabi/ - 05/07/2025
- https://gitlab.com/turkman/devel/doc/wiki/ - 05/07/2025
- https://turkman.gitlab.io/devel/doc/wiki/ - 05/07/2025
- https://grok.com/share/c2hhcmQtMw%3D%3D_5c049e44-be57-4652-872f-55def669d917 - 09/07/2025
- https://www.linuxfromscratch.org/lfs/
- https://wiki.archlinux.org/title/GRUB_(T%C3%BCrk%C3%A7e)#UEFI_sistemler - 08/07/2025
- https://tldp.org/HOWTO/html_single/SquashFS-HOWTO/ - 09/07/2025
- https://askubuntu.com/questions/437880/extract-a-squashfs-to-an-existing-directory - 11/07/2025
- https://grok.com/share/c2hhcmQtMw%3D%3D_2ad2ac6b-d067-40b0-9b55-46db0c2c98dc - 10/07/2025
- https://chatgpt.com/

Not: Metin düzenlemelerinde chatgpt kullanılmıştır.

Geliştiricilere Mesajımız

Gnu/Linux, açık kaynaklı bir işletim sistemidir. Kullanıcı ve geliştiriclere bir çok avantajlar sunmaktadır. Bunlar;

- Kodlarına erişilebilir(Açık Kaynak)
- Dağıtılabir
- Değiştirilebilir
- Virüsten etilenme azdır
- Özelleştirilebilir
- Bağımlılığı azaltır
- Güvenlik en önemli şarttır
- Düşük donanımlarda iyi performan verir

Bu özellikleri açısından Gnu/Linux tercih etmek çok avantajlıdır. Geliştirme aşamalarına destek vermek, öğrenmek bize, toplumumuza ve ülkemize sayısız faydalar saylayacaktır.

Lisans Bilgileri

1. **Kaynak Kodlar (Source Code)**:

   Bu dokümandaki kaynak kodların tamamı Free Software Foundation tarafından yayınlanan GNU Genel Kamu Lisansı'nın (GPL) 3. versiyonu ile lisanslıdır.

   Lisansın bir kopyasını şu adresten edinebilirsiniz: https://www.gnu.org/licenses/gpl-3.0.html

İletişim

- https://github.com/kendilinuxunuyap
- https://kendilinuxunuyap.github.io/x11
- kendilinuxunuyap@gmail.com

ISBN: 978-625-00-5873-2

Yayın Yılı: Nisan 2026