5.Doküman

Yardım Alma Komutları

Konu Etiketleri

yardım alma , help , man , manuel , whatis , apropos

Yardım Almak

Bu kısımda bir nevi öğrenmeyi öğreneceğiz. Bunu da Linux sistemlerinin sahip olduğu çok geniş çaplı yardım mekanizmasını kullanarak başaracağız. Linux sistemlerinde yer alan bu geniş çaplı yardım mekanizmasının bulunmasının birçok nedeni var. Ancak genel olarak; çok fazla komutun çok fazla argüman alması veya her bir programın kendine has kurulum ve kullanım komutlarının olmasından kaynaklanıyor diyebiliriz. Ayrıca, Linux'un bünyesinde bulundurduğu yardım sayfalarının dışında da birçok yardım alma kaynakları mevcuttur. Zaten zamanla göreceksiniz ki; Linux'a kurmak için edindiğimiz hemen her araç veya program kurulum dosyalarının beraberinde, kurulum ve kullanımla ilgili açıklamayı içeren belgelerle birlikte geliyor olacak.

Bu bağlamda, Linux sistemlerinin temel doküman-bilgi kaynaklarını 3 türe ayırabiliriz.

Bunlar; **bilgi sayfaları(info)**, **kılavuz sayfaları(manuel)** ve **uygulamalar ile gelen `/usr/share/doc` konumunda bulunan dokümanlar**dır. Bizler de zaman zaman unuttuğumuz için veya bilmediğimizden dolayı, bu yardım sayfalarına ve dokümanlarına danışıyor olacağız. Bu girizgahtan sonra artık yavaş yavaş yardım alma komutlarımıza geçelim.

help Komutu

Hiç ingilizce bilmiyorum diyen birinin bile, "**help**" ifadesinin "**yardım**" anlamında olduğunu bildiğini düşünüyorum. Yani bu sebepten `help` komutu akılda kalması en kolay komutlardandır. Komutun kullanımına geçecek olursak; örneğin daha önce kullandığımız yetki verme işlevini gören `chmod` komutu ile ilgili yardım(bilgi) almak isteyelim. Bunun için komut satırına `chmod --help` şeklinde komutumuzu yazıyoruz. Ve aşağıda görüldüğü gibi gerekli bilgileri içeren yardım sayfası bizi karşılıyor.

```
└─| chmod --help
Usage: chmod [OPTION]... MODE[,MODE]... FILE...
  or:  chmod [OPTION]... OCTAL-MODE FILE...
  or:  chmod [OPTION]... --reference=RFILE FILE...
Change the mode of each FILE to MODE.
With --reference, change the mode of each FILE to that of RFILE.

  -c, --changes          like verbose but report only when a change is made
  -f, --silent, --quiet  suppress most error messages
  -v, --verbose          output a diagnostic for every file processed
      --no-preserve-root  do not treat '/' specially (the default)
      --preserve-root    fail to operate recursively on '/'
      --reference=RFILE  use RFILE's mode instead of MODE values
  -R, --recursive        change files and directories recursively
      --help     display this help and exit
      --version  output version information and exit

Each MODE is of the form '[ugoa]*([-+=]([rwxXst]*|[ugo]))+|[-+=][0-7]+'.

GNU coreutils online help:
Full documentation
or available locally via: info '(coreutils) chmod invocation'
┌──(root@taylan)-[~]
└─|
```

Bu kullanımın dışında **help** komutunun bir de **help komut** şeklinde kullanımı var, ancak bu kullanımda her zaman komut hakkında yardım bilgisi bulunmayabiliyor. O yüzden ilk öğrendiğiniz yol önceliğiniz olsun. Örneğin **help chmod** yazdığımızda komut satırı yardım bilgisi bulunmadığını belirtti.

```
┌──(root@taylan)-[~]
└─| help chmod
bash: help: no help topics match `chmod'.  Try `help help' or `man -k chmod' or `info chm
┌──(root@taylan)-[~]
└─|
```

"*Peki madem her zaman sonuç vermeyebiliyorsa neden ikinci kullanım yöntemi var ?*" ..diyecek olursanız: Bazı istisnai komutlarda **komut --help** şeklinde yazıldığında komutun kullanım şekli nedeniyle çıktı alınamıyor. Buna bir örnek verelim ancak örnekte kullanacağım komutu ileride ayrıntılı ele alacağım şimdilik sadece konumuzla ilgili olan kısımına odaklanın lütfen.

Örneğin; **echo** komutu kendisinden sonra yazılan ifadeyi ekrana basıyor. Yani ben **echo Merhaba** yazarsam, konsol ekranı da çıktı olarak "Merhaba" şeklinde bir ifade basıyor.

```
┌──(root@taylan)-[~]
└─| echo Merhaba
Merhaba
```

Eğer ben **help** komutunu kullanarak, **echo** komutu hakkında yardım bilgisine ulaşmak istersem; komutumu **echo --help** şeklinde kullandığımda, ekrana yardım bilgisi değil yalnızca "**--help**" ifadesi basılıyor.

```
┌──(root@taylan)-[~]
└─| echo Merhaba
Merhaba
```

İşte bu gibi durumlarda **help** komutunu, ikinci kullanım şekliyle kullanmaktan başka çare kalmıyor. Hemen bu durumu teyit edelim; eğer komutumu **help echo** şeklinde yazarsam çıktılara **echo** komutunun yardım bilgilerinin yer aldığını aşağıdaki resimde görebilirsiniz.

```
┌──(root@taylan)-[~]
└─| help echo
echo: echo [-neE] [arg ...]
    Write arguments to the standard output.

    Display the ARGs, separated by a single space character and followed by a
    newline, on the standard output.

    Options:
      -n        do not append a newline
      -e        enable interpretation of the following backslash escapes
      -E        explicitly suppress interpretation of backslash escapes

    `echo' interprets the following backslash-escaped characters:
      \a        alert (bell)
      \b        backspace
      \c        suppress further output
      \e        escape character
      \E        escape character
      \f        form feed
      \n        new line
      \r        carriage return
      \t        horizontal tab
      \v        vertical tab
      \\        backslash
      \0nnn     the character whose ASCII code is NNN (octal).  NNN can be
                0 to 3 octal digits
      \xHH      the eight-bit character whose value is HH (hexadecimal).  HH
                can be one or two hex digits
      \uHHHH    the Unicode character whose value is the hexadecimal value HHHH.
                HHHH can be one to four hex digits.
      \UHHHHHHHH the Unicode character whose value is the hexadecimal value
                HHHHHHHH. HHHHHHHH can be one to eight hex digits.

    Exit Status:
    Returns success unless a write error occurs.
```

Yani sonuç olarak istisnai komutların haricinde `help` komutu `komut --help` şeklinde kullanılıyorken, bazı özel durum oluşturan komutlar mecburen `help komut` yapısıyla kullanılmak durumunda kalıyor. Bu istisnaları kesinlikle bilmek zorunda değilsiniz. Sadece `help` komutu ile yardım bilgisine ulaşmak istediğinizde ilk olarak `komut --help` yapısını kullanın eğer yardım bilgisine bir şekilde ulaşamazsanız bu sefer komutunuzu `help komut` şeklinde kullanın.

man(Manuel Sayfası) Komutu

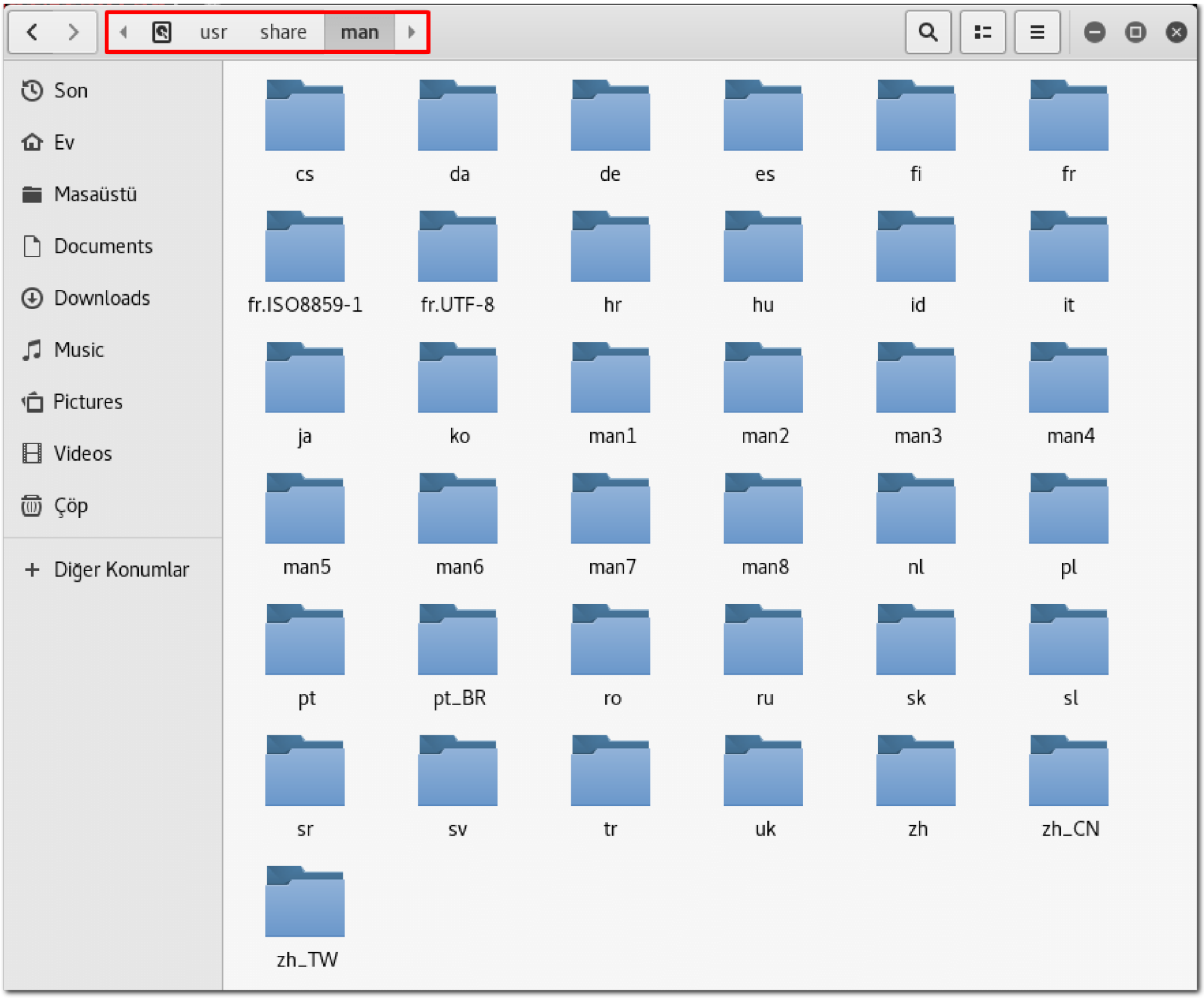
man(**manuel**) sayfaları temel yardım alma dosyalarıdır. Ve **kılavuz sayfaları** olarak da bilinir.

`man` komutunu kullanmak için komut satırına, hakkında bilgi edinip yardım almak istediğimiz komutu `man komut` şeklinde giriyoruz. Örneğin ben `chmod` hakkındaki bilgilere ulaşmak istiyorsam konsola `man chmod` şeklinde yazarak gerekli bilgilere ulaşabilirim. Komutun çıktısında göreceğiniz gibi uzunca bir açıklama sizleri bekliyor. Açılan bu kılavuz sayfasında yön tuşlarını ve `space` tuşunu kullanarak rahatlıkla gezinebilirsiniz. Ayrıca `man` sayfasının kısayollarını ve kullanımını görmek isterseniz `man` sayfası açıkken `h` tuşuna basarsanız sizi `man` kılavuzunun yardım sayfası karşılar, orada **man** komutunun kullanımı ile ilgili detaylı bilgi yardımı mevcuttur. Kılavuz sayfasını kapatmak isterseniz de, sadece `q` tuşuna basmanız yeterli olacaktır.

Şimdi biraz da **man** sayfasının iç yapısına değinecek olursak:

- **NAME:** Komutun ismi ve açıklama kısmı.
- **SYNOPSIS:** Komutun kullanım açıklaması(nasıl kullanılacağı).
- **DESCRIPTION:** Komutun yaptığı iş(fonksiyonu) hakkında açıklama.
- **EXAMPLES:** Komutun kullanımı ile ilgili örnekler ve açıklamalar.
- **SEE ALSO:** Diğer ilgili başlıklar.

`man` kılavuzunun komutlarla ilgili tuttuğu bilgi sayfaları `/usr/share/man` konumu altında tutulur. Yani konsol ekranına basılan yardım(kılavuz) bilgileri bu konum içerisinde yer alan dosyalardan alınarak basılır. `man` kılavuz sayfalarının kaynağı burasıdır ve bu dosyalar belirli bir düzene göre sıralanmıştır.



İlgili dizin içerisindeyken bakalım; örneğin, `man` sayfasının yapılanması nasıl oluyor yani **/usr/share/man** konumunda yer alan dil dosyaları dışındaki diğer dosyalar olan **man1, man2, man3, man4, man5, man6, man7, man8** dosyaları ne ifade ediyor kısaca ona değinelim.

- **man1:** genel kullanıcı programlarını ifade eder.
- **man2:** sistem programlarını ifade eder.
- **man3:** kütüphane fonksiyonlarını(C programlama ile ilgili) ifade eder.
- **man4:** özel dosyaları ifade eder.
- **man5:** dosya biçimlerini ifade eder.
- **man6:** ekran koruyucuları ve oyunları ifade eder.
- **man7:** diğer kategorilere girmeyen çeşitli komutları ifade eder.
- **man8:** sistem yönetimini ve bakımını ifade eder.

whatis Komutu

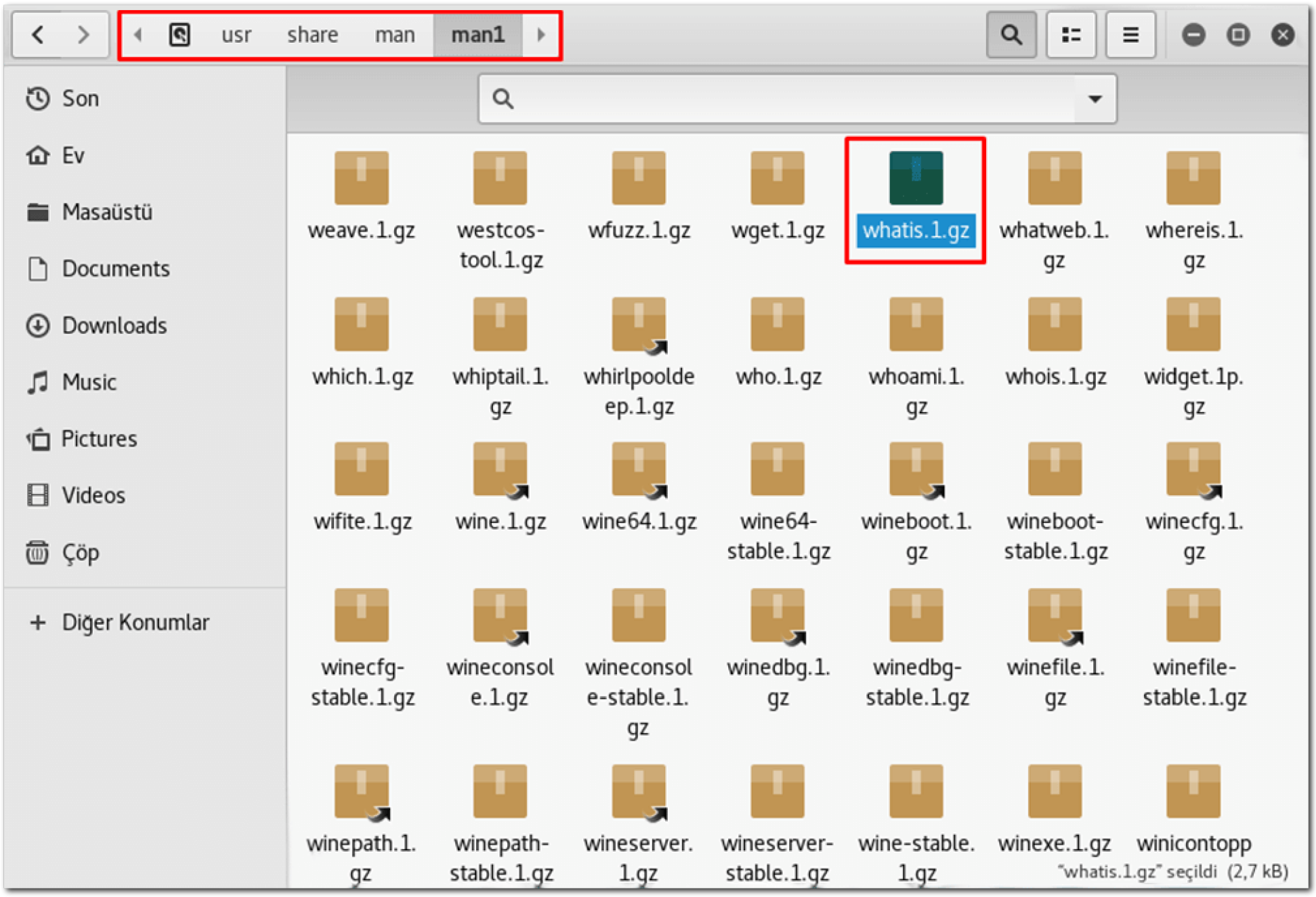
Hazır yeri gelmişken yukarıda gördüğümüz `man` sayfası yapılanmasıyla ilgili olarak `whatis` komutundan da söz edelim. Bu komut sayesinde hangi komutun hangi `man` sayfasında olduğunu öğrenebiliyoruz. Daha iyi anlamak için örnekler yapalım. Komutun kullanımı `whatis komut` şeklindedir.

```
└─| whatis chmod
chmod (1)            - change file mode bits
chmod (2)            - change permissions of a file
```

`whatis` komutuna, `chmod` komutunun **man** sayfasındaki açıklamalarının, yukarıdaki açıkladığımız (man1,man2..3..4..5..6..7..8) dosyalarından hangi dosyada olduğunu sorduk. Konsol yanıt olarak hem **1** hemde **2** de bulunduğunu bizlere bildirdi. Siz bunu istediğiniz komut için sorgulayabilirsiniz hatta `whatis` komutu için bile sorgulayabiliriz.

```
└─| whatis whatis
whatis (1)           - display one-line manual page descriptions
```

Bu çıktıların doğruluğu `man` sayfalarının tutulduğu **/usr/share/man** dizinine gidilerek kontrol edilebilir. Örneğin ben **/usr/share/man/man1** konumuna gittiğimde `whatis` komutunun yardım sayfalarının burada olduğunu görebiliyorum. Sizler de çıktıları bu şekilde teyit edebilirsiniz.



`whatis` komutunun kullanımı bu kadar ile sınırlı değil ancak ben geri kalanını burada vermiyorum. Dileyen arkadaşlar `man` komutu yardımı ile gerekli bilgilere ulaşabilirler.

apropos Komutu

Yardım alma komutlarını noktalamadan önce; Sizlere `man` sayfasındayken `h` tuşuna basarsak `man` sayfasının kullanımı hakkında detaylı bilgiye ulaşabileceğimizi belirtmiştim. O sayfaya bakarsanız `man -k` şeklinde parametre alan komutun işlevinin, sorguladığımız komutun geçtiği uygulamaları listelemek olduğunu görebilirsiniz. Yani örnek vermek gerekirse komut satırına `man -k chmod` yazdığımızda çıktısı aşağıdaki şekilde olacaktır.

```
└─$ man -k chmod
chmod (1)            - change file mode bits
```

Gördüğünüz gibi belirtmiş olduğumuz `chmod` komutunun geçtiği uygulamalar listelenmiş oldu. İşte `apropos` komutu da tam olarak bu işin aynını yapıyor. Örneğin Terminale `apropos chmod` komutunu girersek çıktısı aşağıdaki şekilde olacaktır.

```
└─$ apropos chmod
chmod (1)            - change file mode bits
```

Çıktılar karşılaştırıldığında görülüyor ki; `man -k` komutu ile `apropos` komutları arasında işlevsel yandan hiç bir fark bulunmuyor.

Komutun kullanım amacını daha iyi anlamak için; örneğin, bir komutun ismini tam olarak hatırlayamıyorsunuz ancak işlevini hatırlıyorsunuz diyelim. İşte bu noktada `apropos` komutu sayesinde **man klavuz sayfaları içinde anahtar kelime araştırması yapılmasıyla** yazdığınız kelime ile ilgili tüm komutlara ulaşabiliyorsunuz. Yani aslında biz `apropos` komutuyla sadece **man kılavuz sayfalarında detaylı bir araştırma** yapmış oluyoruz.

Örneğin

Bir komut vardı.. silme işlevindeydi.. neydi.. neydi diye düşünüyorken; konsola `apropos delete` şeklinde yazarsanız, karşınıza man klavuz sayfalarında yer alan, aradığınız "**delete**"anahtar kelimesiyle uyuşan ifadeler listelenir. Yani siz hatırlamadığınız komutun işlevinin silmek olduğunu bildiğimizden "**delete**" ifadesini `apropos` komutu ile sorgulayarak **man sayfalarında "delete" ifadesinin geçtiği kısımları** listelemiş oluyorsunuz. Yani man sayfalarında yer alan açıklamalar sayesinde, sistemle ilgili hatırlayamadığınız kavramları kolaylıkla sorgulayabilirsiniz.

```
└─$ apropos delete
git-branch (1)       - List, create, or delete branches
git-replace (1)      - Create, list, delete refs to replace objects
git-symbolic-ref (1) - Read, modify and delete symbolic refs
git-tag (1)          - Create, list, delete or verify a tag object signed with GPG
groupdel (8)         - delete a group
shred (1)            - overwrite a file to hide its contents, and optionally delete it
systemd-tmpfiles (8) - Creates, deletes and cleans up volatile and temporary files and di
systemd-tmpfiles-clean.service (8) - Creates, deletes and cleans up volatile and temporar
systemd-tmpfiles-clean.timer (8) - Creates, deletes and cleans up volatile and temporary
systemd-tmpfiles-setup-dev.service (8) - Creates, deletes and cleans up volatile and temp
systemd-tmpfiles-setup.service (8) - Creates, deletes and cleans up volatile and temporar
tr (1)               - translate or delete characters
userdel (8)          - delete a user account and related files
```

Bir örnek daha yapalım. Mesela ben komutun yalnızca bir kısmını hatırlıyorum, yani komutun yazılışının tamamını hatırlayamıyorum diyelim. İşte bu gibi bir durumda da yine `apropos` komutumu ya da `man -k` komutumu kullanarak, tamamını hatırlayamadığım ilgili komutu bulabilirim.

```
└─$ man -k kill
choom (1)            - display and adjust OOM-killer score.
kill (1)             - send a signal to a process
killall5 (8)         - send a signal to all processes.
pkill (1)            - look up or signal processes based on name and other attributes
skill (1)            - send a signal or report process status
systemd-rfkill (8)   - Load and save the RF kill switch state at boot and change
systemd-rfkill.service (8) - Load and save the RF kill switch state at boot and change
systemd-rfkill.socket (8) - Load and save the RF kill switch state at boot and change
systemd.kill (5)     - Process killing procedure configuration
yes (1)              - output a string repeatedly until killed
└─$ apropos kill
choom (1)            - display and adjust OOM-killer score.
kill (1)             - send a signal to a process
killall5 (8)         - send a signal to all processes.
pkill (1)            - look up or signal processes based on name and other attributes
skill (1)            - send a signal or report process status
systemd-rfkill (8)   - Load and save the RF kill switch state at boot and change
systemd-rfkill.service (8) - Load and save the RF kill switch state at boot and change
systemd-rfkill.socket (8) - Load and save the RF kill switch state at boot and change
systemd.kill (5)     - Process killing procedure configuration
yes (1)              - output a string repeatedly until killed
```

Yani `apropos` ve `man -k` komutları man klavuz sayfaları içerisinde bizim sorguladığımız her türlü ifadeyi arayarak bulup getiriyor. Ayrıca kılavuz(man/manuel) sayfa içeriklerinin büyük çoğunluğunun İngilizce olduğunu da ele alarak, belirtmek isterim ki; sorgulamalarınızı mümkün oldukça İngilizce üzerinden yaparsanız sonuca ulaşmanız sizin için çok daha kolay olacaktır.

Bu konudaki noktayı da **man kılavuz sayfaları güncelleme** işlemi ile yapalım. "*Nedir bu işlem ve neden gerekli ?*"..diyecek olursanız; Elbette man sayfalarının sürekli kullanılabilir şekilde verimli olması için, zamanla yeni bilgilerin girişi ve eski bilgilerin düzenlenmesi yani kılavuz sayfalarının güncellenmesi şarttır. Eğer aradığımız yardımı man sayfasında bulamadığınız bir durum olursa man sayfalarını güncelleyerek tekrar sorgulama işlemini yapabilirsiniz. Ara ara güncellemek yeni bilgilere de ulaşmamıza olanak sağlar. Güncelleme için konsola `mandb` komutunu girmemiz yeterli olacaktır. Komut satırı, güncelleme işleminden sonra yapılan değişiklikleri de son satırda bizlere bildirir.

```
└─$ mandb
Purging old database entries in /usr/share/man...
Processing manual pages under /usr/share/man...
Purging old database entries in /usr/share/man/ru...
Processing manual pages under /usr/share/man/ru...
Purging old database entries in /usr/share/man/sl...
Processing manual pages under /usr/share/man/sl...
Purging old database entries in /usr/share/man/sr...
Processing manual pages under /usr/share/man/sr...
Purging old database entries in /usr/share/man/sv...
Processing manual pages under /usr/share/man/sv...
Purging old database entries in /usr/share/man/tr...
Processing manual pages under /usr/share/man/tr...
Purging old database entries in /usr/share/man/zh_CN...
Processing manual pages under /usr/share/man/zh_CN...
Purging old database entries in /usr/share/man/zh_TW...
Processing manual pages under /usr/share/man/zh_TW...
Processing manual pages under /usr/local/man...
0 man subdirectories contained newer manual pages.
0 manual pages were added.
0 stray cats were added.
0 old database entries were purged.
```

Yardım alma komutlarını bilmek bir zorunluluk değil ihtiyaç meselesidir. Zaten zamanla bu komutlara ve kullanımlarına alışacaksınız. Yardım sayfalarında yer alan çoğu açıklamanın İngilizce olmasını da dert etmeyin, ne yaparsak yapalım eninde sonunda bu işlerin yolu İngilizce'den geçiyor. Yani artık bu duruma alışmamız gerek. Bu noktada kendimizi biraz zorlamalı ve kesinlikle pes etmemeliyiz. Kendimizi biraz zorlayarak, çaba harcayarak öğrenirsek, öğrenilenlerin kesinlikle daha da kalıcı olacağına emin olabilirsiniz. Sakın pes etmeyin çok iyi gidiyoruz...

Alıştırmalar Hakkında

Yalnızca okumak yetmez, öğrendiğiniz bilgilerin kalıcı olabilmesi için bolca alıştırma yapmalısınız. Doküman içerisindeki bilgileri pekiştirmek için aşağıdaki alıştırmalar ile başlayabilirsiniz. Elbette burada yer alan alıştırma faaliyetleri dışında, konuyu öğrendiğinizi hissede kadar kendiniz de bolca pratik yapmayı da ihmal etmeyin lütfen. Aksi halde öğrendiğiniz bilgiler kısa sürede unutulup gidecektir.

`ls` komutu hakkında bilgi almak için `help` komutunu kullanın.

`echo` komutu hakkında bilgi almak için `help` komutunu kullanın.

`man` komutu ile `ls` komutu hakkında bilgi alın.

`ls` komutunun **man**(**manuel**) **kılavuz sayfaları**nda yer alan bilgilerinin hangi dosyada tutulduğunu öğrenin.

"***kullanıcı***" anahtar kelimesini kullanarak **man klavuz sayfaları**nda yer alan ilgili komutları listeleyin.

Geri Bildirimde Bulunun

Sizlere daha verimli bir kaynak sunabilmemiz için, uygulamada veya dokümantasyonlarda yer alan tüm hata ve eksiklerimizi bize bildirebilirsiniz.

Geri Bildirimde Bulunun