

# Dance with Linux



Learn GNU/Linx the fun way

SaharShaker.com

تقدیم به همه کسانی که لینوکس و فلسفه آزادی را دوست دارند و برای یادگیری و از آن مهمتر برای یاد دادن وقت می‌گذارند.

## سلام :)

این pdf صرفا دستنوشته‌های دختری است که علاقه زیادی به دنیای لینوکس و پایتون دارد و تلاش می‌کند با انتشار این نوشته، افراد بیشتری را با این دنیای هیجان‌انگیز آشنا کند.

اگر نکات، کتاب، فیلم، مقاله و یا هرچیزی که فکر می‌کنید به روند پیشرفت این بحث کمک می‌کند را دیدید برایمان ارسال کنید.

تا زمانی که این علاقه و یادگیری تداوم داشته باشد، این pdf نیز آپدیت خواهد شد. برای دریافت آخرین آپدیت در ایمیل‌تان، [اینجا](#) کلیک کنید.

اگر برایتان مفید بوده، با دیگران به اشتراک بگذارید.



@saharshaker



shrshaker



saharshaker.com

## صرفاً جهت اطلاع!

این پی‌دی‌اف به صورت رایگان ارائه شده اگر جایی دیدین که دارن ازش سوء استفاده می‌کنن یا یجورایی می‌فروشنش خبر بدین بیل بیارم با هم بکشیم‌شون.

خیلی مرسی!

حمایتون به پیشرفت این کار خیلی کمک می‌کننه.

بزرگترین حمایت شما می‌تونه این باشه که کتاب رو به هر شکلی که فکر می‌کنین share کنین یا معرفیش کنین



[saharshaker.com](http://saharshaker.com)



من فقط پلدم از موس و کیپورد و خود کامپیووتر استفاده کنم.  
این کتاب بدردم میخوره؟

بله، این کتاب قدم به قدم، لینوکس را معرفی میکند و تماماً سعی شده مباحثت به سادگی بیان شود. با این حال اگر سوالی داشتید، در سطح اینترنت بسیاری از انجمن‌ها و سایت‌ها برای پاسخ به این سوالات شکل گرفته‌اند.



از همین الان سوالاش شروع شد. آخه اگه کسی قبل  
سیستم عاملی کار نکرده باشه که به فکر لینوکس نمی‌افته. فقط  
خواهشنا با این سوالات چاپی نگو فامیل منی، مرسی اه.



چرا نمی‌ذاری بیبیید من سوالمو پرسم؟؟؟ مسئولین  
چرا پیگیری نمی‌کنن؟؟

یه کلام از مادر عروس، خودش این شکلیه عروس چه شکلیه آخه.  
مسئولین اگه رسیدگی بخوان پکن اول دختر تو رو فیلتر  
می‌کنن. بعدشم کی چلو تو گرفته؟؟ داد می‌زنی همین. پکو یا  
چواب می‌ده یا نمی‌ده دیگه





چرا اصلاً لینوکس؟

لینوکس یکی از مهمترین پیشرفتهای تکنولوژی در قرن 21 است. علاوه بر اینکه به رشد اینترنت کمک بسیار شایانی کرده است، یک الگوی جا افتداده است برای اینکه شرکت‌ها و حتی افراد به تنها‌یی بتوانند ایده‌های خودشان را پیاده‌سازی کنند.

اندروید بر پایه لینوکس است و همان طور که شاهدیم روزانه هزاران گوشی اندروید به فروش می‌رسد. خود شرکت گوگل از هزاران هزار سرور لینوکسی استفاده می‌کند. با فراغیرشدن اینترنت اشیاء، استفاده از لینوکس بیشتر و بیشتر می‌شود به طوری که روزانه چیزی بیش از 700 هزار دستگاه تلویزیون ارائه می‌شود. بسیاری از شرکت‌های معروف و مطرح از لینوکس استفاده می‌کنند و برای پیشرفت و توسعه آن سرمایه‌گذاری می‌کنند، مثل IBM یا Oracle.

حتی لینوکس در بخش‌ها نظمی هم حرف برای گفتن دارد. مثلاً از آن در تفنگ‌های هوشمند استفاده می‌شود. تفنگ‌های هوشمند بر اساس یک فاصله مشخص، روی یک هدف تمرکز می‌کنند و آن را تعقیب کرده و در زمان مورد نظر به آن شلیک می‌کنند.

در صنایع حمل و نقل هم از لینوکس استفاده می‌شود؛ مثلاً در قطارهای فوق سریع یا جاها‌یی که بحث رانندگی خودکار مطرح است.

و یک دنیا استفاده دیگر دارد که به راحتی در اینترنت می‌توانید پیدا کنید، مثل صنایع غذایی، سیاست و...



راستی پرامون کامنٽ پذارید و مارو از حال خودتون پی خبر  
ندارین!

در ضمن نظراتتون رو پا گوش چان می شنویم کلی پرامون  
دلگرمیه.



Hey guys, it's john

یه ریاٽ هم گذاشتیم پرای آدمای خسته تر از خودم. به  
هر دلیل اگر حال نداشتین تا سایت پدین و احیانی  
دیدید چاپی سوتی دادیم، قبلاً اینکه آپدمون در سطح  
محلى، استانى، کشورى و پيئناللى بره په اين ریاٽ  
تلگرام پفرستید، تو سایت هم نوشته شد که چه پهتر

@saharshakerbot





پەزىزەتلىكىيى كۈرتىڭە نەزىقە ئېنۋەتكىيى

و بېرىنگەتە نۇرىيەتلىكىيى

## فهرست

11 .....	فصل اول
12 .....	حال ندارم بخونم الان
13 .....	شخصیت‌های اصلی
14 .....	داستان شکل‌گیری لینوکس
21 .....	وقتی از آزادی حرف می‌زنیم دقیقاً از چی حرف می‌زنیم؟!
29 .....	چیزی که باعث شد لینوکس فراتر از دنیای برنامه‌نویسا بره
31 .....	جایی که لینوکس ترکوند!!!!
33 .....	نقش نت اسکیپ در مهم کردن لینوکس
42 .....	نگاه کمی فنی‌تر به سیستم‌عامل
48 .....	از Debian تا Red Hat
49 .....	Fedora
53 .....	نصب لینوکس
55 .....	نصب لینوکس روی virtual box
66 .....	نصب لینوکس روی سیستم
68 .....	فصل دوم: قلب لینوکس
69 .....	شروع کار با دسکتاپ
75 .....	قدمزدن در فضای دسکتاپ
82 .....	کرنل (kernel)

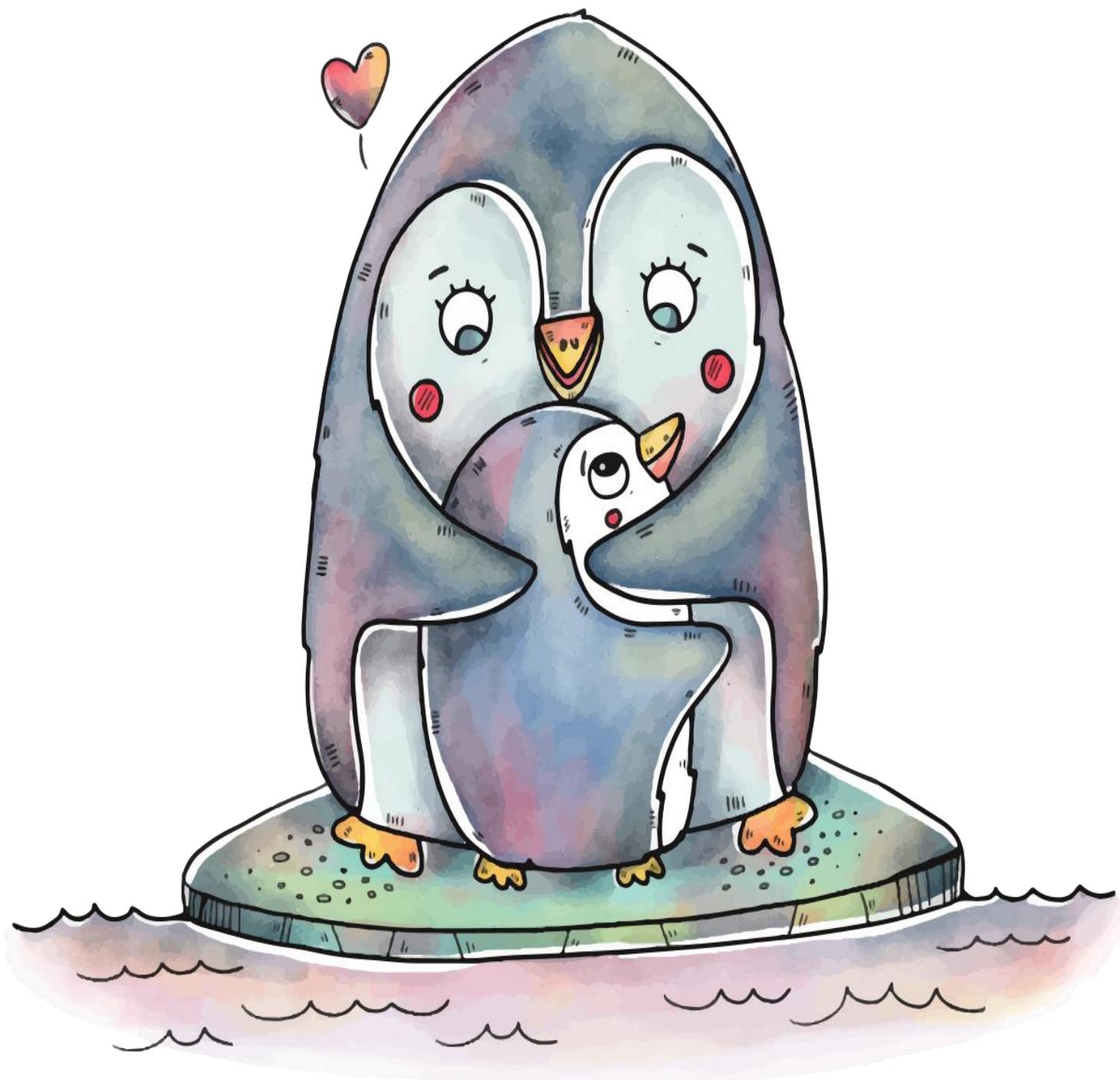
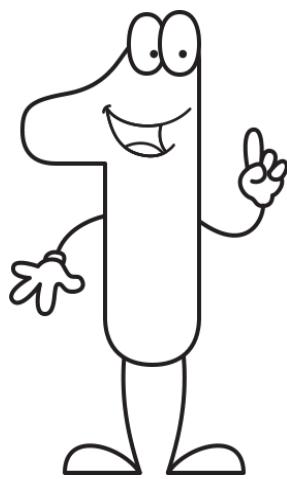
84 .....	چگونه به یک shell دسترسی داشته باشیم؟
87 .....	چگونه کار می‌کند؟ Bash
88 .....	بررسی لاین‌خود
88 .....	فهمیدن shell دیفالت
89 .....	بررسی دایرکتوری‌ها
90 .....	چجوری از یک shell خارج شویم؟
91 .....	نحوه نوشتمن کامندها یا command syntax
91 .....	آپشن‌ها (options)
97 .....	آپشن‌هایی که مقدار می‌پذیرند
99 .....	پیداکردن اطلاعات در مورد کامندها
100 .....	1-چک‌کردن PATH
101 .....	2-استفاده از کامند help
103 .....	3-می‌توانید help-را با دستوری که می‌خواهید به کار ببرید
106 .....	4-دستور man
108 .....	5-استفاده از info
108 .....	6-استفاده از دستور gnome-help
109 .....	آرگمان‌ها (arguments)
110 .....	جایی که آرگمان‌ها با آپشن‌ها قاطی می‌شوند
112 .....	متاکاراکترها
112 .....	متاکاراکتر نقطه‌ویرگول

113 .....	متاکاراکتر space
114 .....	متاکاراکتر امپرسند &
115 .....	متاکاراکتر دبل امپرسند &&
116 .....	متاکاراکتر پایپ، pipe با علامت
117 .....	متاکاراکتر دابل پایپلاین
118 .....	متاکاراکتر > یا خروجی استاندارد
122 .....	متاکاراکتر کوچکتر یا < یا ورودی استاندارد
131 .....	متغیرهای محیطی (enviroment variables)
133 .....	کامندهای پایه‌ای
134 .....	کامند mkdir
144 .....	پاک‌کردن فایل، کامند rmdir
147 .....	ساخت فایل
150 .....	کامند rm
153 .....	کامند cp

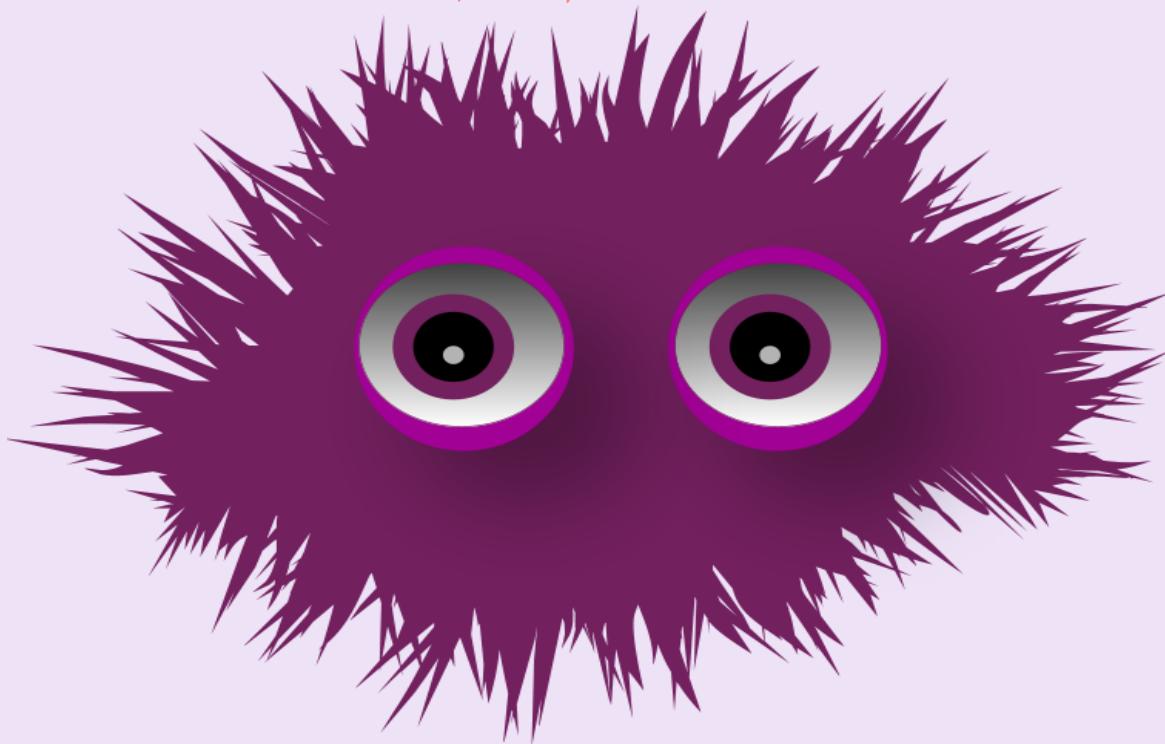
# فصل اول

تولد لینوکس از کجا آغاز شد؟

نصب لینوکس



حال ندارم بخونم الان



حال نداری؟!

بیا یه داستان بگم برات، خوندن این داستان واجب نیست اصلا، ولی می‌تونه  
کمکت کنه بیشتر عاشق لینوکس بشی و از فلسفه شکل‌گیریش سر دربیاری.  
تازه این pdf هم از آسون شروع می‌شه و کمکم فنی می‌شه.

لینوکس هیولا نیست که !

اگر کاربری باشید که تازه پا به دنیای لینوکس گذاشته است، شاید کلی سوال و ایده‌های  
مبهم در ذهن تان باشد که دنبال جواب می‌گردد یا از سایتها و افراد مختلف چیزهایی را  
شنیده است. این فصل جایی است که کمک می‌کند دنیای اپن‌سورس و لینوکس را از زبان  
خالقان آن بشنویم و با فلسفه شکل‌گیری اش بیشتر آشنا شویم.

## شخصیت‌های اصلی



### Richard Stallman

ریچارد استالمن

بنیان‌گذار پروژه گنو، طرفدار آزادی نرم‌افزار



### Eric Raymond

اریک ری蒙د

برنامه‌نویس، مهندس نرم‌افزار، نویسنده مقاله کلیسا و بازار، پیشناز دنیای اپن سورس



### Larry Augustin

لری آگوستین

مدیرعامل و بنیان‌گذار شرکت‌های

VA Research, SugarCRM, Geeknet



### Linus Torvalds

لینوس توروالدز

مهندس نرم‌افزار، دولوپر کرنل لینوکس



### Bruce Perens

بروس پرنز

برنامه‌نویس، مدافع جنبش اپن سورس، او تعریف نرم‌افزار اپن سورس را ایجاد کرد.



## داستان شکل‌گیری لینوکس<sup>۱</sup>



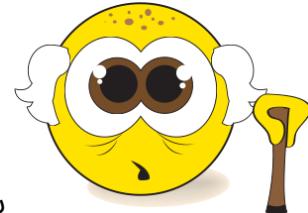
برای توضیح این که لینوکس چیه باید توضیح داد سیستم عامل چیه. و قضیه سیستم عامل اینه که آدم هیچ وقت قرار نیست اون رو ببینه چون هیچ‌کس در واقع از سیستم عامل استفاده نمی‌کنه. آدم‌ها از برنامه‌های کامپیوترشون استفاده می‌کنند. و تنها هدف زندگی سیستم عامل کمک به اجرای این برنامه‌هاست. پس سیستم عامل هیچ وقت خودش کاری انجام نمی‌ده. فقط منتظر می‌شه برنامه‌ها درخواست منبع خاصی رو بکنن. یا یه file از روی دیسک بخوان یا برنامه‌هایی بخوان که اونهارو به دنیای بیرون متصل کنه. اون وقته که سیستم عامل می‌آد وارد عمل می‌شه و سعی می‌کنه برنامه نوشتن رو برای مردم آسان‌تر کنه.



اپن‌سورس راهیه که مردم بتونن روی نرم‌افزارها باهم همکاری کنن بدون این که درگیر اون همه مشکلات حقوقی معنوی بشن. طوری که هر وقت کسی می‌خواهد یه نرم‌افزار بخره

<sup>۱</sup> این داستان برگرفته از دیالوگ‌های فیلم Revolution Os است.

مجبوه کلی قرارداد بالا و پایین کنه و با کلی وکیل چک و چونه بزنه . به طور کلی ما فقط می خواهیم نرم افزار کار کنه . و می خواهیم که بتوانیم از کمک های دیگران برای تصحیح و ... برخوردار باشیم . بنابراین بعضی از حقوق و اموال معنوی رو یک جورهایی قربانی می کنیم . و اجازه می دیم تمام دنیا از نرم افزار استفاده کنن .



بیاید یکم برگردیم عقب؛ قبل از اینکه حرف از لینوکس باشه، یه ریچارد استالمن بود که بابای جنبش نرم افزار آزاد لقب داشت .

ریچارد استالمن:



من سال 1971 وارد آزمایشگاه هوش مصنوعی MIT شدم . وارد اجتماع شکوفایی از هکرها شدم . کسایی که عاشق برنامه نویسی و عاشق کشف کارهایی بودن که می تونستن با کامپیوترها بکنن . اونها یه سیستم عامل درست کرده بودن که همچو همونجا نوشته شده بود ! عضو تیمی شدم که به ارتقای سیستم عامل ادامه می داد و قابلیت های جدیدی به اون اضافه می کرد . این شغلم بود و من عاشقش بودم . ما همه عاشقش بودیم . به همین خاطر این کار رو می کردیم . و اسم سیستممون رو گذاشته بودیم : «*the incompatible time sharing system*». فارسیش می شه : «سیستم اشتراک زمانی ناسازگار» که نشون دهنده روحیه بازیگوشیه . که مشخصه هکرهاست . هکرها کسانی هستن که از هوشمندی بازیگوشانه لذت می برند . خب اولین مشکلی که برآمدون پیش او مدد وقتی بود که دنیای خارج بهمون فشار آورد که کلمه عبور داشته باشیم . ما روی کامپیوترهایمان کلمه عبور نداشتیم . و دلیلش این بود که هکرهایی که اون اول سیستم رو طراحی کرده بودن متوجه شده بودن که کلمه عبور راهی برای رئیس هاست که بتوانن تمام کاربرها رو کنترل کنن . و نخواسته بودن قفل و کلیدی برای رئیس ها بسازن که بتوانن اونها رو کنترل کنن . بنابراین این کار رو نکردن و این مورد رو از قلم انداختن . فسلفه ما این بود که هر کس که پشت



کامپیوتر نشسته باید بتونه هر کاری که دلش می خواهد بکنه. و کس دیگه‌ای که دیروز اونجا بوده نباید بتونه کاری رو که آدم امروز می‌کنه رو کنترل کنه. وقتی اونها روی یکی از دستگاه‌های MIT کلمه عبور گذاشتند، من و یک مشت از هکرهای دیگه خوشمون نیومد. تصمیم گرفتم یه جور هک شورشی انجام بدم. کشف کردم چطور کلمه‌های عبور رو رمزگشایی کنم. اینجوری با نگاه کردن به پایگاه داده که کلمات عبور در اون قرار داشت، می‌تونستم بفهمم هر کسی برای ورود به سیستم چی تایپ می‌کنه. بعد به همون آدم پیغام می‌فرستادم و می‌گفتم:

سلام، من یعنیم که کلمه عبور خلاص رو انتخاب کرده‌اید! چطور حمون کاری رو  
بلنید که من کردم؟ یعنی به جای کلمه عبور فقط ایسرو بزنید؟ خیلی کوتاه‌تره و  
تازه تایپ کردندم خیلی آسوده‌تره

و البته با این پیغام من داشتم تلویحا بهشون می‌گفتم که امنیت سیستمشون عملا در حد شوخیه ! (جاست فور فانه) در هر صورت با این هک داشتم اونهارو دعوت هم می‌کردم. آخرشم یک‌پنجم کاربران اون کامپیوتر به من ملحق شدن. و به عنوان کلمه عبور فقط اینتر رو می‌زدند.



افکاری که الان به چیزی که دنیای اپن سورس می‌گیم، از کجا اومدن؟  
این کار چطور شروع شد؟ چه کسی شروعش کرد؟



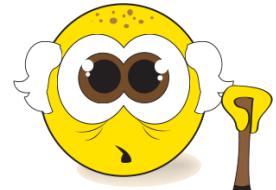
میعنی بین سالی  
59 تا 49

بروس پرنز:



این کار در واقع با پیدایش کامپیوتر شروع شد. چون در اون زمان نرم افزار بین مردم رد و بدل می شد. و فکر می کنم تازه در اواخر دهه 70 یا اوایل دهه 80 بود که مردم شروع کردن به بستن نرم افزارهاشون. و گفتن: «نه اصلا نمی شه سورس کد رو نگاه کنید. نمی شه نرم افزار رو تغییر بدید حتی اگه لازم باشه برای کامپیوتر خودتون درستش کنید. در واقع می شه مایکروسافت رو مقصراً قسمتی از این قضیه دونست. اون ها یکی از پیشگامان واقعی مدل نرم افزار خصوصی هستن.

همون fix کنید خودمون!



اواسط دهه 70 گروهی از هکرها و کامپیوترازان سیلیکون ولی که تفنتی کار می کردند، باشگاه کامپیوتراهای خانه ساز را تشکیل دادن. (انگلیسیش: Homebrew computer club). سال 76 بیل گیتس از شرکت تازه تاسیس مایکروسافت، یه نامه داد به این گروه. تا قبل از این کاربران کامپیوتر بدون اینکه خیلی تو فکر مالکیت نرم افزارها باشن آزادانه اون رو رد و بدل می کردن. بیل گیتس توی این نامه نوشته بود:



برای من حساس‌ترین بازار تفنن در حال حاضر نبود دوره‌های نرم‌افزاری، کتاب‌ها و نرم‌افزارهای خوب است. بدون نرم‌افزار خوب و مالکی که برنامه‌نویسی را بفهمد، کامپیوتر تفني شما هدر رفته است. آیا کسی برنامه‌هایی با کیفیت برای بازار تفنن خواهد نوشت؟... بازخوردی که ما از صدها نفر که می‌گویند از بیسیک استفاده می‌کنن گرفته‌ایم، بسیار مثبت بوده. هرچند دو نکته خارق العاده وجود دارد؛ یک: بیشتر این کاربران بیسیک را نخریده‌اند و دو: با مقدار حق التالیفی که ما از این تفني کاران دریافت کرده‌ایم، دستمزد کاری که صرف آلترا بیسیک شده است کمتر از ساعتی 2 دلار می‌شود. آخه چرا؟ همان‌طور که اکثریت تفني کاران مطلع هستند، بیشتر شما نرم‌افزارهای خود را می‌رزید پول سخت‌افزار را باید داد ولی نرم‌افزار را می‌توان شریک شد. چه اهمیتی دارد که کسانی که آن‌ها را تولید می‌کنند پولی دریافت کنند؟ آیا این منصفانه است؟ یک کاری که شما با دزدیدن نرم‌افزارها نمی‌کنید جواب دادن به میتس براي مشكلاتی است که داشته‌اید. میتس از فروش نرم‌افزار پولی در نمی‌آورد. کاری که شما دارید می‌کنید این است که جلوی نوشته‌شدن نرم‌افزارهای خوب را می‌گیرید. چه کسی می‌تواند به رایگان کار حرفه‌ای انجام دهد؟

کدام تفني کار می‌تواند 3 سال وقت سر برنامه‌نویسی بگذارد، تمام مشکلات را پیدا کند، مستندات آن را تولید کند و بعد آن را به رایگان توزیع کند؟ حقیقت این است که هیچ‌کس غیر از ما سرمایه گذاری زیادی در نرم‌افزارهای تفني نکرده است. آن‌هایی که آلتربیسیک را دوباره می‌فروشنند چه؟ آیا آن‌ها از نرم‌افزارهای تفني پول در نمی‌آورند؟ بله ولی کسانی که به ما گزارش شده‌اند بالاخره خواهند باخت. آن‌ها کسانی هستند که تفني کارها را بدنام می‌کنند و باید از هر جلسه باشگاه که در آن پیدایشان شود بیرون‌شان انداخت.

من از نامه‌های هرکسی که بخواهد دینش را بدهد یا نظر و پیشنهادی داشته باشد استقبال می‌کنم.

بیل گیتس. سهام دار اصلی مایکروسافت.



ریچارد استالمن:

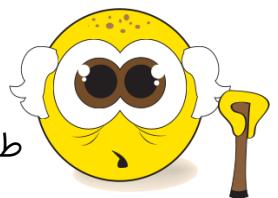


من یه سیستم‌عامل نویس هستم. اگر یک سیستم‌عامل دیگه درست می‌کردم و بعد به عنوان مولف همه رو تشویق می‌کردم که اون رو به دیگران بدن (کن) نه تنها می‌تونستم برای خودم راهی فراهم کنم که بدون خیانت به بقیه مردم از کامپیوتر استفاده کنم بلکه می‌تونستم این راه رو در اختیار بقیه هم بذارم. همه راهی برای فرار از اون تنافق اخلاقی پیدا می‌کردند. پس این پروژه رو توی ژانویه 84 شروع کردم. این وقتیه که از شغلم توی MIT استعفا دادم و کار ساخت سیستم‌عامل گنو رو شروع کردم. اینجا باید توضیح بدم که کلمه گنو، خودش یه هکه! چون مخفف «گنو یونیکس نیست» است.

62



من داشتم سیستم‌عاملی رو درست می‌کردم که شبیه سیستم‌عامل یونیکس بود ولی خود سیستم‌عامل یونیکس نبود. این سیستم فرق داشت مجبور بودیم تمامش رو از صفر بنویسیم. چون یونیکس خصوصی بود. ما اجازه نداشتیم یونیکس رو به اشتراک بذاریم. پس مجبور بودیم جایگزینی براش بنویسیم.



طی دهه 80 که ریچارد استالمن داشت پروژه گنو رو می‌ساخت. دانشمندان کامپیوتر توی دانشگاه کالیفرنیا در برکلی در حال ساخت سیستم‌عامل آزاد خودشون بودن. این سیستم‌عامل که به عنوان یونیکس برکلی یا BSD شناخته می‌شد برپایه هسته



یونیکس و با اجازه AT&T ساخته می شد. به هر جهت، به دلیل مشکلات قانونی با AT&T و به دلیل چند پارچگی سورس کد، هکرها و دیگر کاربران غیرسازمانی به کندي BSD را می پذیرفتند.

ریچارد استالمن:



خب یونیکس از تعداد زیادی برنامه مجزا تشکیل شده بود که با همدیگه ارتباط برقرار می کردند. بنابراین فقط لازم بود این برنامه هارو یکی یکی جایگزین کنیم. پس شروع کردم به نوشتن. بعد کم کم افراد دیگه ای بهم ملحق شدن چون من اطلاعیه ای منتشر کرده بودم که از مردم دعوت می کرد به من ملحق بشن و توی نوشتن این برنامه ها کمک کنن. تا حدود سال ۹۱ عملا همه برنامه هارو جایگزین کرده بودیم.

این برنامه هایی که جایگزین کرده بودند چه برنامه هایی بودند؟



خب ما باید یه سیستم عامل کامل می داشتیم. یک کرنل لازمه، که برنامه ای که منابع رو به همه برنامه های دیگه تخصیص می ده، یه کامپایلر لازمه تا برنامه هارو از سورس کدی که برنامه نویس ها می فهمن به یک سری عدد ترجمه کنه . برای این کار برنامه های دیگری هم در کنار مترجم لازم اند. یه دیباگر، یه تکست ادیتور، تکست فورم، ایمیل و ... چیزهای خیلی خیلی زیادی لازمه. سیستم عامل های شبیه یونیکس صدها برنامه مثل این دارند.

مایکل تیمن:



اطلاعیه ریچارد استالمن رو دیدم و فوریه ۸۷ خودشو ملاقات کردم. اون برای یه دوره آموزشی پنج روزه برای برنامه Emacs به شرکت ما اوmd. در طول روز اون درباره روش های جدید نگاه به Emacs توضیح می داد. و روش های گسترش دادن و بهتر کردن سورس کدهای Emacs حرف می زد. ولی شبها سرش حسابی گرم یه کامپایلر بود و



اون موقع هنوز اون رو برای عموم منتشر نکرده بود. برای همین یکم مراقب بود که چه کسی داره سورس کدش رو می بینه. ولی من خیلی مشتاق بودم و تا ماه ژوئن اون سال که اعلامش کرد و بلافاصله از اینترنت دانلودش کردم با برنامه ور رفتم و چندتا سوال ازش پرسیدم وقتی سورس کد رو برash پس فرستادم، اون با ناباوری دید که چقدر سریع تو نسته بودم تکنولوژی اون رو ارتقا بدم.

## وقتی از آزادی حرف میزنیم دقیقا از چی حرف میزنیم؟!



ریچارد استالمان:



مهم ترین خاصیت گنو اینه که نرم افزار آزاده. و آزاد بودن نرم افزار نه به قیمتش که به آزادی ربط داره. پس به آزادی بیان فکر کنید. آزادی ای که دارم صحبتیش رو می کنم اینه که اگر خواستین تغییراتی در نرم افزار بدین یا کسی رو استخدام کنید که برآتون تغییرات رو انجام بده. اگر برای کسب و کارتون از یه نرم افزار استفاده می کنید که بتونیں مجدداً نسخه های اون رو کپی کنید و با مردم share کنید. تا برنامه رو بهتر کنید و نسخه بهتر شده رو منتشر کنید که بقیه مردم هم از مزایای اون بهره مند بشن. و این آزادی هاست که نرم افزار آزاد رو از نرم افزارهای غیر آزاد متمایز می کنه. این آزادی هاست که مردم رو قادر می کنه یه اجتماع (یه کامیونیتی) تشکیل بدن. شما اگر تمام این آزادی ها رو نداشته باشید بینتون تفرقه می افته و یکی بہتون مسلط می شه .

نرم افزار آزاد که می گین همه چیش ولنگه و بازه؟! هیچ لاینسیس یا مجوزی نداره؟



اگه نرم افزارهارو در مالکیت عمومی بگذاریم یکی می تونه بیاد و یک خرده تغییر توی اون بده و تبدیلش کنه به یک بسته نرم افزاری خصوصی و معنیش اینه که کاربرانی که از نرم افزار ما استفاده می کنند آزادی همکاری یا به دیگران دادن اون رو نخواهند داشت. برای جلوگیری



از این وضعیت ما از تکنیکی به اسم "کپی لفت" استفاده می‌کنیم. مفهوم کپی لفت یه "کپی رایت" سر و تهه. کاری که ما می‌کنیم اینه که می‌گیم این نرمافزار کپی رایت داره و ما نویسنده‌گان اون به شما اجازه می‌دیم که دوباره اون رو توزیع کنید، تغییر بدید یا چیزی بهش اضافه کنید. ولی وقتی که دوباره اون رو توزیع می‌کنید باید دوباره تحت همین شرایط باشه. نه بیشتر نه کمتر.

تا هرکسی که اون رو از شما می‌گیره آزادی همکاری با بقیه مردم رو، اگه بخواهد، داشته باشه. و به این روش، نرمافزار هر جایی که بره آزادی همراهش می‌ره. و همکاری با بقیه مردم. و تشکیل اجتماعات به یک حق لاینفک تبدیل می‌شه.

این چی بود؟ این چه لایسنسی بود؟



خب کپی لفت ایده کلی بود. برای استفاده از اون باید یک نمونه مشخص داشت. نمونه مشخصی که ما برای اکثر بسته‌های نرمافزاری گنو به کار می‌بریم این است:

### GNU GENERAL PUBLIC LICENSE (GPL)

(مجوز عمومی همگانی گنو)

یک سند به زبان حقوقی که این کار رو انجام می‌ده. خیلی ها همین مجوز رو به کار برداشتند، مثلا : لینوس توروالدز هم از این مجوز برای لینوکس استفاده کرد.

بروس پرنز:



خب مجوزی که من بکار بردم، GPL بود همونی که ریچارد استالمان نوشت و به نظر من این مجوز کار بسیار حیرت انگیزیه. این یکی از محدود مجوزهای نرمافزاریه که از جایگاه اجتماع نوشته شده. نه از جایگاه محافظت از یک شرکت یا در مواردی مثل مجوزهای MIT



و BSD. از جایگاه اجرای یک برنامه تحت حمایت دولتی . GPL از این نظر واقعاً بی‌نظیره. اون فقط مجوز نیست یک فلسفه کامله که فکر می‌کنم انگیزه تعریف اپن‌سورس شد. من کتمان نمی‌کنم که خیلی از کارهای من از کارهای استالمن ناشی شده.



خیلی از شرکت‌های تجاری شروع به استفاده از این نرم‌افزارها و سیستم‌عامل گنو و لینوکس کردند که نقطه شروع آن‌ها، آزمایشگاه تحقیقات الکترونیک در دانشگاه استنفورد بود. و منبع الهام اولین شرکت‌های گنویی و لینوکسی شد.



چجوری از گنو پول درمیارین؟



لری آگوستین:

مایکل تیمن شرکتی تاسیس کرده بود به اسم نرم‌افزاری سیگنوس. با این ایده که حول نرم‌افزارهای آزاد گنو مشاوره و خدمات بفروشه. و خب کار مایکل با سیگنوس خیلی گرفت.



مایکل تیمن:

کلی وقت صرف کردم که بفهمم چجوری قراره پول دربیاریم. در بیانیه اصلی گنو که بخش آخر کتاب راهنمای Emacs گنو بود. استالمن چند راه مختلف برای پول درآوردن پیشنهاد کرده بود.



ریچارد استالمن:



از آغاز جنبش نرم افزارهای آزاد تصور من این بود که جا برای تجارت هم در آن وجود دارد. یکی از مزایای نرم افزارهای آزاد اینه که بازار آزادی برای هر نوع خدمات و پشتیبانی وجود دارد. این جوری وقتی آدم در کارش از نرم افزار استفاده می کنه و پشتیبانی (سایپورت) خوب می خواهد، می تونه آدمهایی رو پیدا کنه که این کار رو براش انجام بدند. می تونه شرکت هایی رو پیدا کنه که کار تامین پشتیبانی هستند. اون ها عموماً مجبورند پشتیبانی خوبی به شما ارائه بدن و گرنه شما کس دیگه ای رو پیدا می کنید. در مورد نرم افزارهای خصوصی، پشتیبانی انحصاریه، عموماً یک شرکت هست که سورس کد رو داره و فقط اون ها هستند که می تونن پشتیبانی ارائه بدن. یعنی عموماً آدم به لطف این طرف وابسته است! مثلاً در مورد مایکروسافت وضع به همین منواله. به این ترتیب تعجبی نداره که پشتیبانی شون اینقدر بده (البته این صحبت ریچارد استالمن مال قدیم بود الانو نمی دونم! عموماً ما ایرانیا که اصل نرم افزار رو استفاده نمی کنیم که پشتیبانی بخوایم یا از طرفی هم مدام تحریمیم!!)

مایکل تیمن:



مزایای نرم افزارهای آزاد فوق العاده بودند، مگر هزینه پشتیبانی داخلی و این مدیران رو خیلی خیلی نگران می کرد. بنابراین ایده اساسی که من داشتم این بود که بتونیم مدلی درست کنیم که بتونه 2 تا 4 برابر قابلیت پشتیبانی و کمک رسانی رو که یک مهندس داخل سازمان داره در اختیار بذاره. و بتونیم این کار رو با نصف تا یک چهارم همون هزینه انجام بدیم. خیال مون می تونه جمع باشه که مردم واقعاً اون رو می خرند. و تا پاییز اون سال ما همه چیز رو درباره این که توی گروه فنی چه کسانی باید باشند، شرایط فروش چی باید باشه و شاخص های کلیدی قیمت چی هستند رو درآورده بودیم. در واقع در نوامبر 89 بود که شرکت رو ثبت کردیم. اونم به اسم سیگنوس.





بروس پرنز:

خیلی دقیق می‌تونم بگم که سیگنوس اولین شرکتی بود که مختص نرم‌افزارهای آزاد بود. پشتیبانی سیگنوس از نرم‌افزارهای آزاد شکاف بسیار مهمی رو پر کرد چون نرم‌افزارهای بسیار خوبی داشتیم که می‌شد مفتی به دستشون آورد ولی نمی‌شد براشون پشتیبانی پیدا کرد. اون ها پولشون رو با ارائه پشتیبانی در می‌آوردن.




ریچارد استالمن:

کرنل آخرین چیزی بود که نوشتنش رو شروع کردیم و هنوز خیلی از شروعش نگذشته بود که سر و کله لینوس توروالدز پیدا شد.

اون یک کرنل ساخت و قبل از اینکه ما مال خودمون رو راه بندازیم راهش انداخت و خیلی هم خوب و حسابی راهش انداخت. اسم کرنلش Linux گذاشت.




لینوس توروالدز:

هدف اولیه من هدف خیلی شخصی‌ای بود. این که بتونم روی کامپیوتر خودم محیطی شبیه به محیطی که روی کامپیوترهای دانشگاه بهش عادت کرده بودم ایجاد کنم و نمی‌تونستم چیزی پیدا کنم که از این جهت به دردم بخوره. این طور شد که من اصولاً تمام عمرم رو پشت کامپیوتر نشسته بودم. به این نتیجه رسیدم که باید خودم دست به کار بشم. بیشتر ایده‌ها رو اون اوایل از سیستم عامل سان گرفتم (SunOS) که چیزی بود که اون موقع توی دانشگاه ازش استفاده می‌کردیم. حالا چی شد که من انقدر سریع تونستم راهش بندازم، این بود که از روش برنامه‌نویسی قدیمی که جوابش رو پس داده بود استفاده کردم و مفهوم یکپارچه بودن رو شکستم.

یکپارچه بودن اصولاً به این معنیه که سیستم‌عامل یک نهاده و تقسیم ناپذیر. در حالی که در microkernel سیستم‌عامل در حقیقت چیزی نیست جز مجموعه‌ای از



سرویس دهنده هایی که کارهای مختلفی انجام می دن و پروتکل مشترکی برای ارتباط با همدیگه دارن.



خب چطور شد که پروژه گنو که این همه جلوتر بود و از قبل داشت این کار رو می کرد چطور شد که اون اینجوری آخر کار وارد شد؟

ریچارد استالمن:



خب در واقع ما کار روی Hurd گنو رو خیلی قبل از اینکه اون لینوکس رو شروع کنه شروع نکردیم و اتفاقاً ما طراحی ای رو انتخاب کردیم که طراحی بسیار پیشرفته ایه. از نظر قدرتی که به آدم می ده ولی از اون طرف معلوم شد که دیباگ کردنش هم خیلی سخته. ما تصمیم گرفتیم کرنل رو که یک برنامه بود چند تکه کنیم و به کلی برنامه کوچکتر تقسیمش کنیم که برای ارتباط برقرار کردن پیام های ناهمzman به هم می فرستادن. مسئله اینه که این روش برنامه نویسی زمینه زیادی برای باگ ها داشتن. باگ هایی که پیدا کردن شون اغلب خیلی سخته چون به این بستگی داره که آیا فلان برنامه، فلان پیغام رو قبل از این می فرسته که اون یکی برنامه بهمون پیغام رو می فرسته یا بعدش؟!! و نتیجه این شد که سال ها طول کشید که تونستیم برنامه رو به کار بندازیم.

رابطه لینوکس با پروژه گنو چیه؟



لینوس توروالدز:



خب می شه گفت در چندین سطح با گنو رابطه داره. یکیش سطح فلسفیه که آدم فکر کنه باز کردن سورس ایده خوبیه.



ریچارد استالمان:



وقتی لینوس کرنل اش رو نوشت این کار رو برای پروژه گنو نمی‌کرد. این کار رو مستقلاً انجام می‌داد و مستقلاً هم منتشرش کرد. ما هم خبر نداشتیم. ولی بعضیا که خبر داشتن تصمیم گرفتن بگردن ببینن دیگه چی می‌توونن پیدا کنن تا کنار کرنل‌شون بذارن و یه سیستم عامل کامل داشته باشن. او ناگشتن و دیدن که عجب! تمام چیزایی که لازم دارن از قبل وجود داشته. فکر کردن چه شانسی آور دیم! ولی در واقع شانسی در کار نبود. اون ها تمام تکه‌های سیستم گنو رو پیدا کرده بودن که فقط کرنل رو کم داشت. به این ترتیب وقتی همه چیز رو سرهم کردن در واقع داشتن لینوکس رو توی شکاف سیستم گنو جا می‌دادن. ولی اونا اینو نمی‌دونستن!

لینوس توروالدز:



خیلی از این برنامه‌ها کار بنیاد نرم‌افزارهای آزاد و کار آدم‌هایی که لینوکس رو دوست داشتن بود. و یک همزیستی بین لینوکس و این برنامه‌ها وجود داره. که این برنامه‌ها روی لینوکس اجرا می‌شن و در عین حال از لینوکس به عنوان پلتفرم استفاده می‌کنن. در حالی که لینوکس به خاطر امکان استفاده از این برنامه‌ها از اون‌ها فایده می‌بره!

چه برنامه‌هایی؟



در واقع مهمترینشون کامپایلر سی گنو است. بدون وجود کامپایلر سی، تولد لینوکس و اکثر این پیشرفتها شدنی نبود. لینوکس از GPL استفاده می‌کنه و من با فلسفه‌ای که پشت GPL هست موافقم. این رو هم بگم که GPL خودش اصلاً متن قشنگی نیست. دلیلش هم احتمالاً اینه که هیچ متن حقوقی‌ای اون قدرها خوشایند نیست و نمی‌تونه هم باشه.





لarry آگوستین:

کم کم متوجه شدم که تو خونه به یه دستگاه یونیکس احتیاج دارم. رفتم دنبالش و دیدم می‌تونم از لینوکس روی یه کامپیوتر شخصی استفاده کنم. با حدود 2000 دلار یه سیستم سرهم کردم که 1/5 تا 2 برابر سریع‌تر از سیستم‌های SUN بود. فوق العاده بود. من 1/5 تا 2 برابر سرعت رو با 1/3 تا 1/4 قیمت داشتم! یه چیزی تو مغزم جرقه زد! فهمیدم اینجا فرصتی نهفته است. این شانس ما بود که کاری رو واقعاً بهتر از sun انجام بدیم اون هم با استفاده از اپن‌سورس و لینوکس.



ایده اسم لینوکس چی بود؟



لینوس توروالدز:



لینوکس اسمی بود که پیش خودم روش گذاشتم فقط به خاطر اسمم لینوس و X هم که حتماً باید باشه بخاطر یونیکس. اولش فکر کردم نمی‌تونم اسمش رو رسماً بذارم لینوکس چون خیلی خودخواهانه می‌شه.

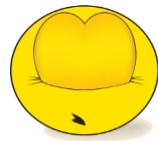


ریچارد استالمان:

اون‌ها فکر کردن یه عالمه برنامه پیداکردن و دارن اون رو دور لینوکس می‌ذارن اینجوری شد که اسم قضیه رو گذاشتمن سیستم عامل لینوکس. و این اصطلاح جا افتاد و نتیجه این شد که میلیون‌ها نفر حالا دارن از این گونه سیستم گنو به اسم سیستم‌عامل گنو/لینوکس استفاده می‌کنن و اکثرشون این رو نمی‌دونن.



بعضی‌ها پیشنهاد می‌کنن این سیستم گنو/لینوکس نامیده بشه. نظر شما چیه؟

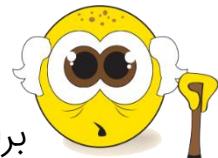


لینوس توروالدز:



خب من فکر می‌کنم موجهه، ولی در صورتی موجهه که یه توزیع گنویی از لینوکس منتشر کنند. همونطور که لینوکس رد هت SuSe مشکلی نداره یا لینوکس سوزه Debian Linux یا لینوکس دبین Linux. چون در واقع اگر آدم بخواهد توزیع خودشو از لینوکس درست کنه اسمش رو هم خودش می‌ذاره. ولی اینکه کلا اسمش رو بذاریم گنو لینوکس به نظرم مسخرست.

## چیزی که باعث شد لینوکس فراتر از دنیای برنامه‌نویسا بره



برای رشد لینوکس به ماورای دنیای برنامه‌نویسان کامپیوتر ، استفاده و کاربردی نیاز بود که اون رو به یه فناوری غیرقابل اغماض بدل کنه چیزی که باعث عبور از این آستانه شد به وجود آمدن برنامه‌ای بود که ساختن وبسایتها پیچیده رو امکان پذیر می‌کرد. این برنامه آپاچی وب‌서ور بود. Apache web server

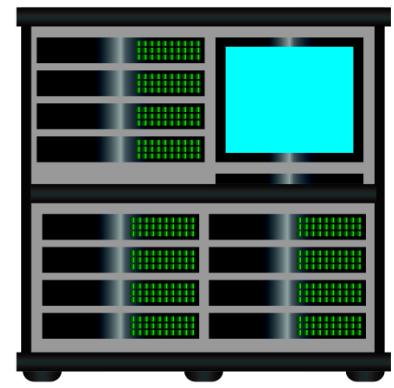
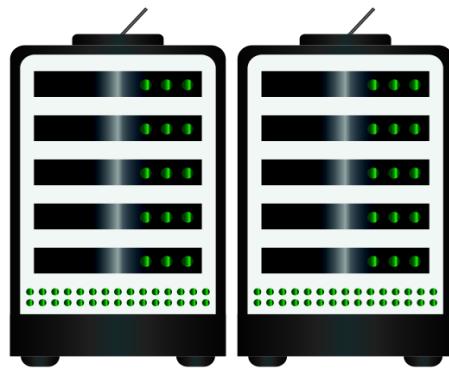
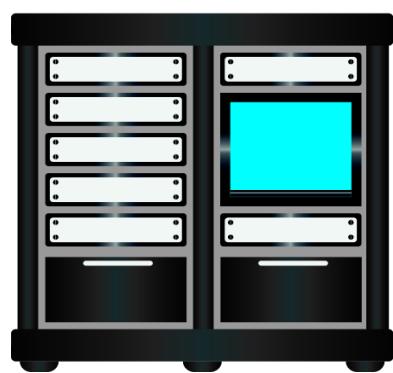
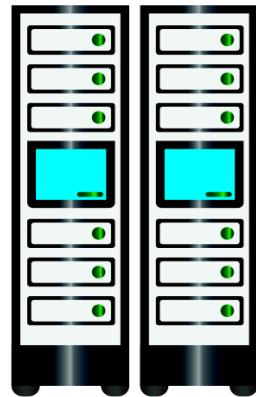
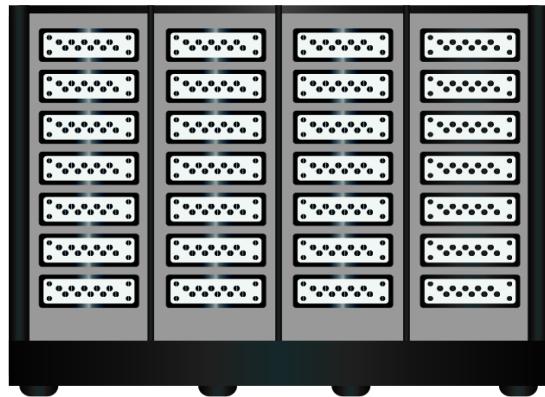
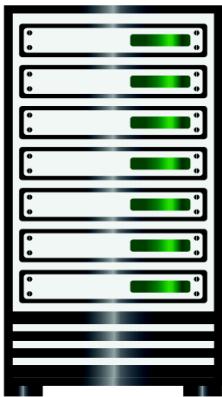
اریک ریموند:



گل سرسبد برنامه‌های لینوکس بدون شک آپاچی وب سرور بود. اگر به تاریخچه لینوکس دقت کنید می‌بینید که منحنی استفاده از لینوکس و منحنی استفاده از اینترنت دقیقا هم‌دیگه رو دنبال می‌کنن. سال ۹۳ که پروژه آپاچی وب سرور شروع شد همون سالی بود



که انفجار پر طرفدار ISP هم شروع شد. وقتی که اینترنت یک کالا فروشی عمدۀ شد. و ایده تجارت الکترونیکی مبتنی بر وب و ارتباطات عمومی به حقیقت پیوست.



BRIAN BEHLENDORF



فکر می‌کنم آپاچی یکی از اولین برنامه‌هایی بود که باعث شد مردم بگن : خب اگر لینوکس نصب کنم یه فایده مشخصی برام داره. البته اون موقع کلی برنامه جالب روی لینوکس وجود داشت. ولی می‌دونید واقعاً انگیزه‌ای وجود نداشت. می‌شه گفت فایده تجاری

نداشت که کسی بجای NT از لینوکس استفاده کنه. تا وقتی که آپاچی اوmd و کلی چیزهای دیگه که به آپاچی وصل می‌شدن. می خواست بگم وقتی آدم می خواست بره و یسری سرور راه بندازه خیلی مقرون به صرفه‌تر بود. مقرون به صرفه‌تر از نظر پول واقعی که اون رو با لینوکس و آپاچی بسازه تا با IIS و NT . حتی اگر به این معنی باشه که آدم یه خرده پول خرج کنه. که به افرادش آموزش بده که چطور ازش استفاده کنن یا آدمهایی رو پیدا کنه که بهش واردن ولی خوبیش این بود که این دانش خیلی گرون نبود. چون کلی دانشجو وجود داشتن که مدت‌ها از لینوکس استفاده می‌کردن و خیلی باهاش آشنا بودن.



اریک ریموند:

اگه به منحنی استفاده وب سرورها نگاه کنید، آپاچی همچ داشته سهم بیشتری از بازار رو بدست می‌آورده. یکسره داشته به رقیب‌های close source اش ضرر می‌زده و علت‌ش اینه که مطمئن‌تره و انعطاف‌پذیرتره، گستردگی‌پذیرتره. کارهایی رو که وب‌مسترها واقعاً نیاز دارن رو می‌کنه. و ترکیب آپاچی و لینوکس راهش رو به تعداد بسیار زیادی از شرکت‌های تجاری باز کرده.



جایی که لینوکس ترکوند!!!



BRIAN BEHLENDORF

آپاچی احتمالاً روی لینوکس و FreeBSD بهتر اجرا می‌شه. و به همین دلیله که کامپیونیتی‌های مربوط به این سیستم‌عامل‌ها هنوز بیشترین مشارکت رو در تولید آپاچی داشتن. و این که این‌ها سیستم‌عامل‌هایی بودن که تامین کنندگان خدمات اینترنت شدیداً شروع به استفاده از اون کردن. و تامین‌کنندگان خدمات اینترنت خیلی از آپاچی خوششون اوMD چون بهشون اجازه می‌داد کلی کارهای متفاوت بکنن که بعضی از سرورهای تجاری



وب نمی‌کردن. مثل این قابلیت که هاست بیش از یک وبسایت روی یک دستگاه باشن که به وضوح اگر آدم یه ISP باشه و 40هزار کاربر داشته باشه و همشون بخوان برای خودشون وبسایت داشته باشن یه مقدار برای آدم مهم می‌شه.



یکی از عوامل کلیدی در رشد لینوکس، به وجود آمدن شرکت‌هایی بود که تخصصشان توزیع و پشتیبانی خود سیستم عامل بود. از میان این شرکت‌های نرم‌افزاری از همه معروف‌تر است.

یکی از توزیع‌های لینوکس بشمار می‌رده، داستان اون چی بوده؟ Red Hat



:DONNIE BRNES

Red Hat به عنوان یکی از محصول‌های مارک یوونیگ شروع شد. Marc Ewing وقتی که داشت توی IBM کار می‌کرد و توزیع لینوکسی می‌خواست که کمی بهتر باشه پس شروع کرد با لینوکس و رفتن. بعد دید داره وقت بیشتری صرف نگهداری توزیع لینوکسش می‌کنه تا صرف پروژه جدیدش توی IBM. اینجوری شد که یجورایی خودش یه توزیع رو شروع کرد. بعد با باب یانگ Bob Young آشنا شد که اون موقع شرکتی به اسم کتاب‌فروشی ACC رو اداره می‌کرد. باب می‌خواست چیزی رو بفروشه که بیشتر مال خودش باشه تا محصولای آدمای دیگه رو. کار بازاریابیش خوب بود. مارک هم می‌دونست که توی بازاریابی به کمک احتیاج داره چون توی بحثای فنی خیلی وارد بود این شد که باهم متحد شدن.



ما در واقع توی آپارتمانی شروع به کار کردیم که مارک یووینگ توش زندگی می‌کرد. بعد از روز دستشویی خونه ترکید و خونه همسایه پایینی رو آب برداشت. اونجا بود که ساکنان آپارتمان فهمیدن ما اونجا یه شرکت راه انداختیم و تصمیم گرفتن بندازنمون بیرون. یه هفته وقت داشتیم اولین دفترمون رو پیدا کنیم و کردیم.



## نقش نت اسکیپ در مهم کردن لینوکس

اریک ریموند:



یه مقاله نوشتیم به اسم کلیسا و بازار The Cathedral & the Bazaar و به این پرداخته بودم که اپن‌سورس چجوری کار می‌کنه. حتی اونموقع هنوز اسم "اپن‌سورس" رو نداشتیم و می‌گفتیم نرم‌افزارهای آزاد. و گفته بودم که چرا ما تونسته بودیم نرم‌افزارهایی با کیفیت بی‌نهایت بالا رو تولید کنیم با اینکه مدام داشتیم قواعد مهندسی نرم‌افزار رو زیر پا می‌ذاشتیم. فرق بین دو روش متفاوت برنامه‌سازی یا developement کردن برنامه رو توضیح می‌دادم. یکی روش سنتی بود یا همون closed development style که من اسمش رو گذاشته بودم روش کلیساوی. توی این روش آدم باید تک‌تک اهداف رو توضیح بده و گروههای پروژه کوچکی داشته باشه که با داشتن اختیارات سلسله مراتبی باهم کار می‌کنن و باید دوره انتشار محصولش هم طولانی باشه. از طرف دیگه اتفاقی که دیدم داره برای دنیای لینوکس می‌افته، روش خیلی فرد به فرد و نامت مرکز و شبیه یه بازار واقعی بود، که فواصل انتشار خیلی کوتاه بود و دائماً از نظرات آدمایی که خارج از پروژه بودن بهره می‌برد.

بروس پرنز:



دلیل اهیمت نت اسکیپ Netscape's اینه که اون‌ها اولین شرکت بزرگی بودن که به دنیای اپن‌سورس اومند. البته ما سیگنوس رو برای ارائه پشتیبانی داشتیم ولی واقعاً



چندان تجاری در کار نبود. نت اسکیپ اصلا برای این اپن سورس شد که یجوری با مایکروسافت بجنگه. که داشت اینترنت اکسپلورر رو بیرون میداد. ولی اجازه نمیداد دیگران متن اون رو داشته باشن. اجازه نمیداد شرکت‌های دیگه همکاری کنن.



FRANK HECKER



(مهندس سابق نت اسکیپ)

من در بخش فروش کار می‌کدم. تصور خوبی داشتم که چرا مردم نرم‌افزارهای ما را می‌خرن و چه چیزهایی لازمه که نرم‌افزارهای ما در برابر محصولات رقیب در بازار موفق باشن. در هر حال مسئله این بود ما می‌دیدیم که با گذشت زمان نرم‌افزارهای ما با نرم‌افزارهای دیگه به خصوص مایکروسافت در رقابت بود. با گذشت زمان قیمت محصولات ما باید کمتر می‌شد چون بقیه داشتن محصولاتشون رو رایگان یا با بهای کم بیرون می‌دادن.

اریک ریموند:



اونا میترسیدن مایکروسافت بازار مرورگر رو به دست بگیره و از اون برای منحرف کردن استانداردهای HTTP و HTML که وب وابسته به اون‌هاست استفاده کنه و به محض



اینکه اون استانداردها رو به ابزار نگهداشتن کاربر تبدیل کردن بعدش میتونستن از اون تسلط استفاده کنن تا نت اسکیپ رو از بازار سورورها بیرون کنن. بازار سورورها هم جایی بود که نت اسکیپ ازش پول اصلیش رو در میاورد.



FRANK HECKER

آدم های کافی برای اینکه بتونیم نرم افزارهای ما را توی بازار زنده نگه داریم رو نداشتیم. مقاله کلیسا و بازار بریک ریموند نقش بزرگی برای انتشار سورس کد برنامه نت اسکیپ داشت. البته از قبلش داشتم به منتشر کردن سورس کدها فکر می کردم. یه مقاله نوشتیم و توش ارجاع دادم به مقاله کلیسا و بازار. قصدیم این بود که یه بحث جنجالی بندازم توی نت اسکیپ که چرا باید سورس کد رو منتشر کنیم. اسم مقاله رو هم گذاشتیم: «سورس کد برنامه نت اسکیپ به عنوان محصولی از نت اسکیپ» عنوان می خواست اینو جا بندازه که سورس کد نه تنها ازش می تونیم در تهیه محصول استفاده کنیم بلکه خودش هم یه محصول حساب می شه ! چیزی که مشتری ها و مردم دیگه بتونن ازش استفاده کنن. بعد فکر کردم خب لاینسنس رو چی بذاریم؟ چطور باید محصولاتم را توی این فضا بفروشیم؟ به رقبا نگاه کردم، مخصوصا مایکروسافت، اگر سورس کد رو منتشر کنیم او ناچه کارایی ممکن بود بکنن؟ آیا راهی بود که از همین سورس کدها علیه خودمون استفاده کنن؟



اریک ریموند:

از اینجا به بعد دیگه جریان اصلی توجه و اعتماد سرمایه‌گذاران رو به ما داد



FRANK HECKER

یه نسخه از مقاله رو دادم به مارک اندریسن Mark Andreessen یکی از بنیان‌گذاران نت اسکیپ اون زمان بود. او نم پخشش کرد و رسید به دست جیم بارکس دیل Jim Barksdale . بعد از مدتی هم او نا اعلام کردن که قصد دارن سورس کد نت اسکیپ رو



منتشر کن! و اینجوری نت اسکیپ توجه عمومی رو به نرم افزارهای آزاد که بعداً گفتن اپن سورس رو جلب کرد. و سیستم عامل لینوکس در کانون توجه قرار گرفت.



لری آگوستین:

وقتی می‌گفتیم free مردم فکر می‌کردن مجانیه. فکر می‌کردن نمی‌تونن ازش پول دربیان یا اونو بفروشن که تصور اشتباهیه. می‌خواستیم این مفهوم رو برسونیم که **نرم افزار آزاد** یعنی **اینکه کدش در دسترسه و بازه**. یه جلسه گذاشتیم و به مفهوم اپن سورس رسیدیم. زنگ زدیم به لینوس گفتیم خوشت میاد ازش یا نه؟ برash جالب بود و خوشش اومد.



از طرفی هم ریچارد استالمن به طور رسمی اعلام کرد که با خونده شدن اپن سورس به جای نرم افزارهای آزاد کاملاً موافقه. چون هدفش از آزادی این بود که مردم برای همکاری باهم و تشکیل کامپیونیتی‌ها آزاد باشن و این برای کیفیت زندگی همه مهمه. و اومدن گفتن که اپن سورس 9تا حق مشخص رو به شما می‌ده:



بروس پرنز:



#### 1-توزيع مجدد و آزاد

آزاد یا free ربطی به قیمت نداره. به معنی آزادیه. آدم باید آزاد باشه که بتونه نرم افزارش رو دوباره بین آدمهای دیگه توزیع کنه. در واقع رایگان بودن یه تاثیر جانبیه. می‌شه بابت توزیع مجدد پول گرفت یا نگرفت.



## 2- در دسترس بودن سورس کد

باید سورس کد همراهش باشه که بشه برنامه رو نگهداری کرد. مثلا اگر بخوان از PC بجای استفاده کنن باید بتونن نرم افزار رو تبدیل کنن.



## 3- کار اشتقاقي هم باید مجاز باشه

Derived Works have to be possible

اگر قرار باشه نرم افزار شما رو کسی گسترش بده باید قادر باشه که نتیجه کار رو توزیع کنه. یه شرطی هم درباره یکپارچگی سورس کد مولف هست



## 4- یکپارچگی سورس کد مولف

می گه مولف می تونه یه جورایی نام خودش رو حفظ کنه. و اگر شما تغییری ایجاد کردید ممکنه مجبور باشید اسم برنامه رو تغییر بدید یا تغییراتتون رو خیلی واضح مشخص کنید که تغییراتتون به حساب مولف گذاشته نشه

## 5- نباید تبعیضی بین آدمها و گروهها باشه

## 6- نباید تبعیضی بین زمینه های کاری باشه

معنی اش اینه که از نرم افزار باید بشه چه در تجارت چه در مدارس استفاده کرد.

## 7- لاینس باید قابل توزیع باشه

من باید قادر باشم که این لاینس رو به شخص دیگه ای بدم و در این صورت لاینس باید باز هم، اگه اون شخص اون رو به شخص سومی بده، کار کنه

## 8- نمی شه لاینس مخصوص به یک محصول مشخص باشه



اگر من نرم افزارم رو روی سیستم ردهت توزیع کرده باشم لایسنس نمی تونه بگه که نمی شه این رو روی سیستم های SuSE یا Debian توزیع کرد.

#### 9- لایسنس نمی تونه نرم افزارهای دیگه رو محدود کنه

اگر من این رو روی یک CD با نرم افزار دیگه ای منتشر کنم، لایسنس نمی تونه بگه اون برنامه دیگه هم باید آزاد باشه و گرنه نمی شه برنامه من رو توزیع کنیں

اریک ریموند:



یه اتفاق دیگه این بود که پایگاه های داده به این جمع اضافه شدن! اواخر ژوئیه و اوایل اوت بود که اوراکل اعلام کرد برنامه هاشو به لینوکس حمل می کنه . و همینطور Sybase و دیگر سازندگان مهم پایگاه داده این رو اعلام کردند.



لری آگوستین:



رفته رفته لیست مخاطبها و مشتریان اپن سورس زیاد شدو کسایی مثل سیسکو ، شرکت دات کام داشتن وارد این لیست می شدن. بالاخره توجه سرمایه گذارها جلب شد. عکس لینوس روی مجله فورچون چاپ شد و این یعنی توی اپن سورس داشت اتفاقایی می افتاد. دیگه سرمایه گذارا نمی تونستن نادیده بگیرن و از دست من که مدام بهشون سرزده بودم خسته شده بودن. آخرش به این نتیجه رسیدن که سرمایه گذاری کنن.



اریک ریموند:



مايكروسافت داره از لینوکس برای دفاع از خودش استفاده می کنه. اون ها از لینوکس برای اثبات این ادعا استفاده کردند که اون ها انحصار رو در دست ندارن چون اصولا لینوکس می تونه هر لحظه اون ها رو از موضع برتری که دارن پایین بکشه. بحث خیلی جالبی بود. یه ادعای کاملا غلط انداز، چون به هیچ وجه پاسخی به این اتهامات نداد که اون ها قبلا



مشغول قلدر بازی و دیگر کارهای غیررقابتی بودن. ولی اونها زرنگی کردن و البته در این مورد، قاضی زیر بار نرفت.



درباره میلیاردها دلار ثروتی که از ساخته تو دراومده و چیزیش مستقیما به تو نرسیده چه احساسی داری؟



لینوس توروالدز:



اگه من لینوکس رو به همه نمی‌دادم اونجوری هم هیچ پولی بهم نمی‌رسید. یعنی می‌خواستم بگم این یه موقعیت بدون باخته. این مسئله که کلی شرکت‌های تجاری وجود دارن به این معنیه که کلی آدم مرتبط با لینوکس هست که قبلًا وقت اضافی‌شون رو روی لینوکس کار می‌کردن و حالا برای کاری که دوست داشتن بکنن پول می‌گیرن. این مسئله از این جهت به من کمک می‌کنه که من به هرحال می‌خواستم اونها روی لینوکس کار کنن.

ریچارد استالمان:



کل پروژه گنو در واقع یک هک بزرگ. کاری بزرگ از سر هوشمندی شورشی و بازی‌گوشانه برای بهترکردن جامعه. من فقط به دنبال بهتر کردن هستم. ولی به روش هوشمندانه





شما لینوکسی؟



حالا هر چی، امیرتونن؟؟؟



چی شد که شما پنگوئن‌ها شدین نماد لینوکس؟ مثلاً این مینیون‌ها پیشون بوده‌یا فیل‌شاه که کارتونشم دارین. اصلاً یه سوال کلی تدر، چرا هر چی جک و چونوره شده نماد رُپان برنامه‌نویسی، چرا یه هیوالارو نمی‌دارن؟ ما په این گوگولی مگولی!!!!



(سیسیسیش). نعیدم. یه روز لینوس مارو توی پاغ وحش دید و عاشقون شد و یه نوع پنگوئن به اسم tux رو که چاق و خوشحاله رو گذاشت روی لینوکسش



Oh my GOD

چه مخفن، خیلی تعجب کردم تو قیاقم معلومه؟

چقد پنگوئن خوشگلی شما!!! لینوس حق داشته ها!

پسر امروز چند شنبه اس؟



هشت شنبه است دیگه. چطور؟

عارضم خدمتتون که ما توی هشت شنبه ها مخ  
نمی زیم



او نوچت پر چه (اساس؟)

پر اساس اصل خودم. اصل خرها.

دلیل از این پالاند؟



## نگاه کمی فنی تر به سیستم عامل

همان طور که لینوکس تورووالدز در ابتدای داستان اشاره کرد، لینوکس یک سیستم عامل کامپیوتری است. یک سیستم عامل شامل نرم افزارهایی است که کامپیوتر شما را مدیریت می کند و اجازه می دهد برنامه های خود را روی آن اجرا کنید. و چندین ویژگی دارد که به آن اشاره می شود:

شناسایی و تنظیم سخت افزارها: وقتی که لینوکس بوت می شود (یعنی وقتی کامپیوترون رو روشن می کنیں)، به اجزاء سیستم شما نگاهی می اندازد (مثل CPU، کارت شبکه و...) و بعد نرم افزارها (درایورها و ماژول ها) و مواردی که نیاز به دسترسی به این سخت افزارهای خاص را دارند لود می کند.

سخت افزار در واقع هر چیز فیزیکی است که بتوانیم به آن دست بزنیم، مثل رم، هارد دیسک ...

مدیریت پروسه ها یا فرآیندها: سیستم عامل باید همزمان چندین کار را اجرا کند و از طرفی تصمیم بگیرد که کدام باید به CPU دسترسی پیدا کند و چه زمانی باید این دسترسی صورت بگیرد. همچنین سیستم عامل باید راه های برای شروع، متوقف کردن، و تغییر وضعیت فرآیندها و پروسه ها را ارائه دهد.

مدیریت حافظه: رم و بخش swap (جلوتر می گیم چیه) باید به برنامه هایی که نیاز به حافظه دارند، داده شود. سیستم عامل این تصمیم را می گیرد که چگونه به حافظه یا همان مموری درخواست بدهد و آن را مدیریت کند.

ارائه دادن یک رابط کاربری: یک سیستم عامل باید راهی برای دسترسی به سیستم را برای یوزر ایجاد کند. سیستم هایی که از لینوکس استفاده می کنند، چیزی دارند به اسم shell . در واقع برای کار با سیستم لینوکس، ما درون shell برنامه یا دستورات را می نویسیم و اجرا می کنیم. اشتباه نکنید! برای اینکه از لینوکس استفاده کنید لزوماً به این معنی نیست که باید از shell استفاده کنید. این روزها با آمدن توزیع های مختلفی، مثل ubuntu، رابط های گرافیکی دارند که می توان بدون داشتن دانش استفاده از shell و دستورات یا به



اصطلاح کامن‌لاین‌ها، از لینوکس استفاده کرد. درست مثل چیزی که در ویندوز شاهد آئیم! یعنی ویندوز را نصب می‌کنید و وارد یک محیط گرافیکی می‌شوید و کارهایتان را انجام می‌دهید. ولی قدرت استفاده از لینوکس، همین داشتن دانش استفاده از shell و کامن‌لاین‌هاست. البته بیشتر مبتدی‌ها از این بخش فرار می‌کنند و به لینوکس پشت می‌کنند. اما مطمئن باشید اگر مفهومی آن را متوجه شوید، استفاده از آن قدرت شیرینی به شما می‌دهد.

کنترل فایل سیستم‌ها: در واقع سیستم‌عامل، مالکیت و دسترسی به فایل‌ها و دایرکتورها (فولدرها) می‌جذب موجود را کنترل می‌کند.

ایجاد دسترسی برای کاربران و احراز هویت آن‌ها (شناسایی آن‌ها): ایجاد یوزر اکانت‌ها و تعیین محدوده برای کاربران. کاربران و گروه‌های کاربری را از هم جدا می‌کند تا بتوانند هرکس به فایل‌های مربوط به خودش دسترسی داشته باشد و یا فرآیندها و پروسه خودشان را انجام دهند و کنترل کنند.

ارائه امکانات مدیریتی: در لینوکس شاید صدها یا هزاران دستور و پنجره‌های گرافیکی وجود دارد تا کارهایی همچون اضافه کردن یوزرها، مدیریت دیسک‌ها، مدیریت شبکه، نصب نرم‌افزار، و به طور کلی ایجاد امنیت و مدیریت کامپیوتورتان را فراهم کنند.

راه‌اندازی سرویس‌ها: برای استفاده از چاپگرهای رسمی به پیام‌های ورودی، و ارائه انواع خدمات مربوط به شبکه و... باید در سیستم‌عامل انجام شود. سرویس‌های زیادی در لینوکس اجرا می‌شود.

لینوکس راه‌های زیادی برای اجرا و متوقف کردن این سرویس‌ها را در اختیار می‌گذارد.

ابزارهای برنامه‌نویسی: طیف وسیعی از ابزارهای برنامه‌نویسی و لایبرری‌های مختلف برای ساختن نرم‌افزارها و اپلیکیشن‌ها موجود است.

به عنوان کسی که می‌خواهد سیستم‌های لینوکسی را مدیریت کند، باید بدانید که هر بخش چگونه کار می‌کند. با اینکه بسیاری از کارها را می‌شود با استفاده از رابط گرافیکی انجام داد



ولی فهم و دانستن دستورات یا همان کامن‌لاین‌ها در shell برای کسی که مدیریت سیستم‌های لینوکس را دارد یک "باید" به شمار می‌رود.

ویژگی‌های پیشرفته لینوکس که اغلب در شرکت‌های بزرگ استفاده می‌شود، به شکل زیر است:

کارهای پیشرفته لینوکس، همچون clustering (خوشبندی)، Real-time computing (مجازی‌سازی)، cloud computing (پردازش یا رایانش ابری)، (یا رایانش بی‌رنگ! برای همینه که توی صفحه اصلی، گفتم از ترجمه‌کردن اصطلاحاتی که مال ما نیستند خوشم نمی‌آید!)، specialized storage (ذخیره‌سازی تخصصی) بیشتر در شرکت‌ها استفاده می‌شود.



چه فرقی بین لینوکس و سیستم‌عامل‌های دیگه هست عایا؟

اگر تازه پا به دنیای لینوکس گذاشتید، حتما گذرتون به ویندوز مایکروسافت یا مک افتاده است. اما فرق اساسی که با لینوکس دارند در زیر به آن‌ها اشاره شده است:

نمی‌توانید کدهایی که برای ایجاد سیستم‌عامل استفاده شده است را ببینید، بنابراین نمی‌توانید سیستم را مطابق نیازهای خودتان و انتظاراتی که از آن دارید تغییر دهید.

نمی‌توانید کدها را برای پیداکردن باگ‌ها چک کنید. راحت بفهمید که ایراد کار از کجاست یا اگر مشکل امنیتی دارد از کجا آب می‌خورد. و از همه مهم‌تر نمی‌توانید یاد بگیرید که یک کد چگونه کار می‌کند.

ممکن است به راحتی نتوانید نرم‌افزارهای خودتان را به سیستم‌عامل وصل کنید تنها به این دلیل که سازندگان سیستم‌عامل شما می‌خواهد این انحصار را حفظ کند!



شاید پیش خودتان بگویید چه اهمیتی داره؟ من که نمی‌خوام دولوپر نرم‌افزار بشم. نمی‌خواهم ببینم که چطوری می‌شه سیستم‌عامل را تغییر بدم.

شاید حق با شما باشد. اما حقیقت این است که بسیاری از افراد و شرکت‌های دیگر که پیشگام خلاقیت و نوآوری هستند از این امکانات و نرم‌افزارهای اپن‌سورس و بعض‌رایگان استفاده می‌کنند، مثلًا گوگل، فیسبوک و...

از طرفی لاینس ویندوز رایگان نیست و اکثر ما به طور غیرقانونی در حال استفاده از آن هستیم. مثلًا هنگام نصب ویندوز کافی است قبل از زدن تیک گزینه License Agreement حوصله به خرج دهید و متن آن را بخوانید! به طور وحشتناکی اشاره می‌کند به کلی از کارهایی که نباید بکنیم و می‌کنیم.

در بخشی از بندهای آن اشاره شده: لاینس این نرم‌افزار به شما فروخته شده است نه خود نرم‌افزار. یا مثلًا در جایی دیگر اشاره شده است که فقط می‌توانید خودتان از این نرم‌افزار استفاده کنید آن‌هم تنها بر روی یک کامپیوتر، نباید به کسی آن را بدهدید و...

اگر جزء اقیلیتی هستید که لاینس را پرداخت کردید، مطمئناً هم در این دنیا و هم در آن دنیا رستگارید! ( فقط دست مارم سر پل بگیرید! )



خوب که چی؟ می‌خواهی پکی تو الان لاینس شو داری؟ و مثلًا کیلی کیلی فارسی پلد نیستی؟

در کل نمی‌خواهم بگویم استفاده از مایکروسافت بد است و اخ! هنوز هم که هنوز است بسیاری از نرم‌افزارها همچون فتوشاپ، یا سیویل‌تری‌دی و اتوکد و... در محیط لینوکس وجود ندارد. البته مشابه آن وجود دارد. مثلًا بجای وردآفیس، چیزی هست به اسم LibreOffice؛ یا بجای فتوشاپ، Gimp وجود دارد ولی به هیچ عنوان به کاملی فتوشاپ و یا نرم‌افزارهای اصلی در ویندوز نیست و تنها پاسخ‌گوی بخشی از نیازهای کاربران است.



شاید ورود به دنیای لینوکس کمک کند تا بیشتر از اینی که هست مصرف‌گرا نباشیم. و چیزهای جدید را تجربه کنیم.

از طرفی چون شرکت‌ها و کمپانی‌های بزرگی از لینوکس استفاده می‌کنند، پس به افراد بیشتری نیاز دارند که این مهارت را داشته باشند که با سیستم‌های لینوکسی کار کنند و آن‌ها را مدیریت کنند.



توزيع لینوکس چیه؟

اگر داستان را خوانده باشید، فهمیدید که لینوکس مستقیماً اشاره به کرنلی دارد که لینوس توروالدز ساخت و هر سیستمی که از این کرنل استفاده کند، اصطلاحاً به آن گنو/لینوکس می‌گویند.

حالا کرنل را لینوس توروالدز ساخت و نرم‌افزارها و یا بهتر بگوییم پکیج‌ها را هم گنو تولید کرده بود. پس به زبان ساده اگر این‌ها را کنار هم بگذاریم و کامپایل کنیم، یک سیستم‌عامل خواهیم داشت. یعنی اینجوری هر کسی توانایی این را دارد که یک لینوکس برای خودش بسازد. به این لینوکس‌هایی که هر کسی برای خودش مطابق نیازش می‌سازد، "توزیع" یا "distribution" می‌نامند. (ولی عامیانه بهش می‌گن)

بعضی از این distro‌ها را شرکت‌ها ساختند. یعنی پشت یک distro، یک شرکت سفت و سخت نشسته و مدام در حال آپدیت‌کردن پکیج‌ها و سایر موارد آن distro است. وقتی بحث شرکتی پشت یک distro می‌آید، یک کلمه Enterprise به آن می‌چسبد. (مثلاً جلوتر در مورد Red Hat distro‌ای که شرکت رد هت یا Red Hat ساخت حرف می‌زنیم، اونجا هم همین کلمه به لینوکس‌ش چسبیده!)

شرکت‌های مطرحی همچون Debian و Red Hat و ... به این ترتیب لینوکس خودشان را از پایه ساختند. در حال حاضر کلی distro داریم.

در واقعیت ساختن یک سیستم‌عامل کار چندان آسانی نیست، چون جدا از اینکه باید برای کامپایل کردن و کنار هم قراردادن پکیج‌ها تلاش کنیم، بحث پشتیبانی آن هم هست. یعنی اگر برنامه‌ای آپدیت شد باید تست کنیم و بعد ارائه کنیم و کلی کارهای دیگر.



حالا یعنی چی؟ راهی نداره؛ ما می‌میریم  
می‌میریم

چرا، بسیاری از شرکت‌ها همین distro‌هایی که وجود داشت، مثل Debian (دبیان)، کمی تغییر دادند و distro خودشان را ارائه کردند. (اینم کی از مزیت‌های دنیای اپن سورس‌ه که می‌تونی هر چیزی رو مطابق نیازت تغییرش بدی) در واقع با اینکار بعضی از distro‌های محبوب، پایه شکل‌گیری و تولد distro‌های دیگر شدند. مثلاً ubuntu محبوب بر پایه Debian ساخته شده است.

در این بین برخی distro‌ها به کل تولدشان برای مصارف خاص و مشخصی بود. مثل: Mandrake، Gentoo، KNOPPIX

نمی‌خواهم پیاز داغ ماجرا را زیاد کنم، اما شکل‌گیری لینوکس تنها یک سیستم‌عامل نبود و نیست. اگر داستان را دنبال کرده باشید، متوجه شدید که پشت آن فلسفه‌ای وجود دارد. آن هم فلسفه آزادی.



لینوکس اگر رایگانه پس ملّت پول و درآمدشون رو از کجا در  
میارن؟!



این جاست که باید تفاوت بین دو واژه freedom با free مطرح شود. همان طور که در بندهای تعریف اپن‌سورس گفته شده، درست است که لینوکس اپن‌سورس می‌باشد اما لزوماً به معنای رایگان بودن نیست. و این یعنی سورس‌کدها در دسترس هست اما ممکن است خود آن "پولی" باشد. یعنی در این بین ممکن است خود فروش سورس‌کد هم یک محصول به حساب بیاید. مثل کاری که شرکت Red Hat کرد. علاوه بر این شرکت‌ها بابت خدماتی که ارائه می‌دهند هم هزینه می‌گیرند.

## از Debian تا Red Hat



خوب این اسمای کج و کوله چیه؟

من می‌دونم! اسم دوچی شد کت یا پلاخدره چاییه که از اپن‌سورس و دنیای لینوکس شروع کردن پول درآوردن و خدمات دادن



در اواخر سال 1990 به چند دلیل محبوب شد:  
یکی استفاده از package management RPM.

دلیل دیگر محبوب شدن این distro، آسانی نصب اش بود. یکسری سوالات ساده را پاسخ می‌دهید که در بیشتر اوقات هم گزینه‌های دیفالت دارد.



دلیل دیگر ش استفاده از محیط گرافیکی بود. با این کار یوزرها دیگر نیازی نداشتند که کامند یا دستوری را یاد بگیرند تا بتوانند از لینوکس استفاده کنند. در محیط گرافیکی ابزارهایی گذاشته شد که یوزر می‌توانست تاریخ و زمان، و کارهای پایه و اولیه خودش را انجام دهد.

سال‌های زیادی بود که Red Hat distro محبوبی برای هر دو قشر شد. هم برای کسانی که حرفه‌ای کار می‌کردند و هم برای کسانی که علاقمند به دنیای لینوکس بودند. به تدریج که دنیای لینوکس گسترده می‌شد و افراد بیشتری وارد آن می‌شدند باعث شد مرز این جنبش و فضا به جایی فراتر از Red Hat بروند. هم کار قبلی را رها کرد و بجای آن شروع کرد و دو سیستم‌عامل را توسعه داد: Red Hat Enterprise و Fedora.

سال 2012 این شرکت اولین شرکت اپن سورسی بود که به درآمد سالانه بیش از ۱ میلیارد دلار رسید. این درآمد را از طریق فروش محصولاتی در حوزه Red Hat Enterprise Linux یا به اختصار RHEL بدست آورد.

با این حال، Red Hat distro های دیگر تمرکز خودشان را روی دسکتاپ‌ها (کامپیوتراهای شخصی) یا مشاغل کوچک گذاشته بودند. اما Red Hat بیشتر روی ویژگی‌ها و برنامه‌هایی کار می‌کرد که برای کسب‌وکارها و دولت نیاز بود.

## Fedora

با اینکه Red Hat RHEL را عرضه کرده بود، اما Red Hat distro ای به نام Fedora را ارائه کرد برای کسانی که می‌خواستند از لینوکس برای مصارف شخصی و رایگان استفاده کنند و از طرفی بتوانند این شرکت را در کامیونیتی دوستداران لینوکس نگه دارد.

Fedora بیش از ۱۶ هزار پکیج نرم‌افزاری را دارد. بسیاری از این‌ها با آخرین تکنولوژی اپن سورس در دسترس هستند. به عنوان یک کاربر یا یوزر، شما قادرید از آخرین نسخه آن برای server (سرور) یا دسکتاپ و کارهای دیگر به رایگان استفاده کنید.



چون Fedora تمرکزش بیشتر بر این است با آخرین تکنولوژی پیش برود، پس خیلی با ثبات نیست! بنابراین برای اجرای بعضی از نرم افزارها یا انجام یکسری کارها، باید کارهای اضافی تری هم برای به راه آفتدان کارتون انجام بدید.

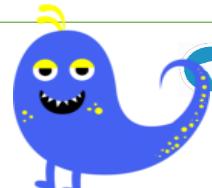
با این حال نباید به آسانی از این توزیع گذشت، Red Hat بسیاری از اپلیکیشن های جدید را در Fedora تست می کند. با کار کردن با این disrto می توانید مهارت هایی که برای کار با بعضی ویژگی های Red Hat Enterprise Linux هست را بدست آورید.

با اینکه Fedora محبوب است اما در سال های اخیر distro ای به نام Ubuntu توانسته بیش از Fedora محبوب شود.

Ubuntu بر پایه Debian محبوب distro ساخته شده است. و نسبت به Fedora از ثبات بیشتری برخوردار است. بسیاری از distro های دیگر بر پایه Debian ساخته شده، که می توانند در سایت distrowatch.com پیدا شون کنند. بیش از 130 توزیع فعال وجود دارند که برپایه Debian هستند. بعضی از distro های محبوب برپایه Debian اینها هستند: elementary OS، Linux Mint (من برای اولین بار که وارد دنیای لینوکس شدم از این توزیع استفاده کردم!), Kali Linux، LXLE، Zorin Os (بیشتر برای امنیت بکار می ره) و بسیاری از توزیع های دیگر. ولی محبوبیت بیشتر Debian بخارط پیدا شن (ubuntu) Ubuntu بود.

نصب این distro بسیار ساده است. از محیط گرافیکی خوبی استفاده می کند و برای نصب، برخی نرم افزارها را پیشنهاد می دهد. (یه چیزی مثل بازار برای اندورید یا play store داره که نرم افزارهایی رو برای دانلود گذاشته)

حتی می توانید Ubuntu را روی CD یا فلاش به صورت لایو load کنید و از آن استفاده کنید. لینوکس لایو به لینوکس هایی گفته می شود که کلا از روی سی دی می تواند بوت شود و همه کارهایتان را می توانید انجام دهید. و اگر هم بخواهید چیزی را save کنید به یک storage دیگه نیاز دارید. بیشتر به این درد می خورد که اگر یک سخت افزاری دارید که دچار



مشکل شده است اینطوری تست کنید، یا می خواهید به هر دلیلی سیستم عامل خودتان را جابه جا کنید.

یعنی چی؟ مخانواده (ینچا نشسته‌ها)! الان قهر می کنه می ره دیگه  
ادامه شو نمی گه. شما به دل نگید. منتظر اینه از چه تو زیعی  
استفاده کنیم؟ او نم پرای شروع؟



وقتی پا به دنیای لینوکس می‌گذارید، اولین سوالی که پیش می‌آید این است که با چه distro یا توزیعی کار را شروع کنم؟

وقتی می‌خواهید از ویندوز استفاده کنید، یکی از ورژن‌ها (مثلا 10 یا 7 و...) را انتخاب می‌کنید و نصب می‌کنید. ویندوز مشخصاً توسط یک کمپانی (مايكروسافت) تولید می‌شود. اما لینوکس اينگونه نیست. و مردم روی بخش‌های مختلف این سیستم عامل کار می‌کنند.

ما دنیایی از distro‌ها را داریم، اما چیزی که مطرح است این است که باید بدانیم که گاهی می‌خواهیم از لینوکس روی دسکتاپ کامپیوتر (کامپیوتری شخصی) استفاده کنیم و گاهی روی سرور. مثلاً ممکن است آن‌هایی که برای سرور هستند رابط گرافیکی نداشته باشند.

در واقع 70 درصد توزیع‌ها مشابه یکدیگرند و در بعضی بخش‌ها مثل مدیریت بسته یا پیکیج منیجمت فرق دارند. همان‌طور که ریچارد استالمن اشاره کرده، سیستم عامل‌های یونیکسی (مثل گنو/لینوکس) چیز جز بسته‌ها نیستند. حالا فرق بعضی از distro‌ها در مدیریت این بسته‌های است. مثلاً RPM از Red Hat استفاده می‌کند ولی distro‌هایی که مبتنی بر debian (دبیان) مثلاً Ubuntu، از مدیریت بسته از dpkg استفاده می‌کنند. برای



همین خیلی درگیر انتخاب distro نشید، برای شروع یکی را انتخاب کنید و بعدها می‌توانید عوض کنید.

برای همین ما در این کتاب از Ubuntu شروع می‌کنیم. پیش از این هم به دلایل اش اشاره شده. با این حال اگر از distro‌های دیگر استفاده می‌کنید مشکلی با بحث‌ها نخواهد داشت.

پا لینوکس چی کاره می‌شم؟ مصلاً زن و پچم می‌تونن چوچه‌ای  
که شما آدم‌می‌خورین رو پغورن؟



پیا فک و فامیلتو جمع کن بید آپرومون رو پردن ای پاپا. پلد  
نیست سوالشم درست پنویسه. مصلاً آخه یا مثلاً؟

شما به گندگی خودت پیخش، هیولان، ندیدن دیگه.



بسیاری از مشاغل مهندسی، شاید لینوکس را در کنار کارهای دیگرشان و برای اجرای ایده‌هایشان استفاده می‌کنند. با این حال، استفاده از لینوکس در دنیای امروزی با شدت زیادی رو به افزونی است. پس زمینه‌های کاری زیاد هم به طبع به وجود می‌آید. با این حال، مشاغلی که به طور مستقیم به لینوکس اشاره دارند:

مهندس نرم‌افزار، پشتیبانی سیستم، برنامه‌نویسی روی لینوکس، آموزش، مهندس شبکه، متخصص سرور، کار با LAMP و...



**نکته:** اگر مشخصا و یک راست! می خواهید آزمون های Lpic را شرکت کنید، بسیاری از افراد در سطح اینترنت به رایگان در حال آموزش این آزمون هستند. مثلًا جادی عزیز.

روند پیشرفت این کتاب به سمت دست و پنجه نرم کردن با لینوکس! و همینطور استفاده از آن در دنیا و ... خواهد رفت. با این حال برخی مطالب مربوط به آزمون ها گفته خواهد شد.

(اینو گفتم برای کسایی که صاف میخوان برن فقط آزمون Lpic رو بدن که این کتاب یه cook book نیست که بتونید اون آزمون رو بدید، منابع زیادی توی نت هست که بیشتر به کمکتون میاد.)

محوصلم سر رفت. می گی اصل کاری رو یا نه؟



## نصب لینوکس

چون distro های مختلف توسط شرکت های مختلف ساخته شده اند، پس مراحل نصب هم ممکن است تفاوت داشته باشد. (مسئله شاید خیلی جدی نباشه! هر distro ای که دستتون بیاد و دوست داشته باشید نصب کنید، کافیه یه سرچ کنید که چجوری نصبش کنم).

ولی چیزهایی که بین نصب distro ها مشترک است و البته مهم، یکی نحوه پارتیشن بندی و دیگری انتخاب فایل سیستمی است که می خواهید انتخاب کنید. والبته بحث نرم افزارهایی که می خواهید داشته باشید. (یا همون نصب پکیج ها)

(گفتیم لینوکس در واقع اشاره مستقیم داره به کرنلی که لینوس ساخت و هر کی این کرنل رو استفاده کنه اصطلاحا بهش میگن لینوکس داره! پس تا الان فهمیدیم که لینوکس با



فرق داره! پس وقتی می‌گه لینوکس، یعنی هر سیستمی که از این کرنل استفاده می‌کنه نه لزوماً یک distro مشخص)

اگر نمی‌خواهید به طور کامل از ویندوز به لینوکس مهاجرت کنید و قصد دارید دو موتوره از سیستم‌تون لذت ببرید، سه راه وجود دارد:

1- درون ویندوز برنامه‌ای برای مجازی‌سازی مثل Virtual box نصب کنید و لینوکس را درون آن بسازید

2- می‌توانید از دو سیستم عامل روی یک کامپیوتر بهره ببرید: کلاً بعد از بوت‌کردن سیستم شروع به نصب اش کنید(یعنی روی خود سیستم‌تون نصب کنید) در راه حل دوم هر بار وقتی که سیستم را روشن کنید از شما می‌پرسد که آماده‌اید با ویندوز پرواز کنید یا لینوکس! در واقع می‌توان لینوکس را به صورت dual boot در کنار ویندوز نصب کرد. فقط اینکه شما می‌توانید تنها یک boot loader داشته باشید. (روی سکتور صفر فقط یکی میره) چون بوت لودر ویندوز نمی‌تواند بوت لودر لینوکس را بالا بیاورد پس بهتر است اول ویندوز را نصب کرده و بعد اقدام به نصب لینوکس کنید.

3- اگر از ویندوز 10 استفاده می‌کنید، آپهای لینوکس را از store ویندوز دریافت کنید و محیط کامندلاین آن را بررسی کنید.



لطف می‌کنی پیش عذارو بگی؟ موارد لازم چهت طرز تهیه‌اش. انگار کلاس آموزش لینوکس پا مانع هستیم

اگر می‌خواهید روی Virtual box یا هر ماشین مجازی دیگری نصب کنید، اول آن را دانلود کنید.



نسخه آخر ubuntu را هم [اینجا](#) می‌توانید پیدا کنید. (اگر نشد دستی سرچ کن: اون بالا بگرد یه گزینه download پیدا می‌کنی)



(ها من پیدا کردم یه فایله ISO اومد پرای دانلود



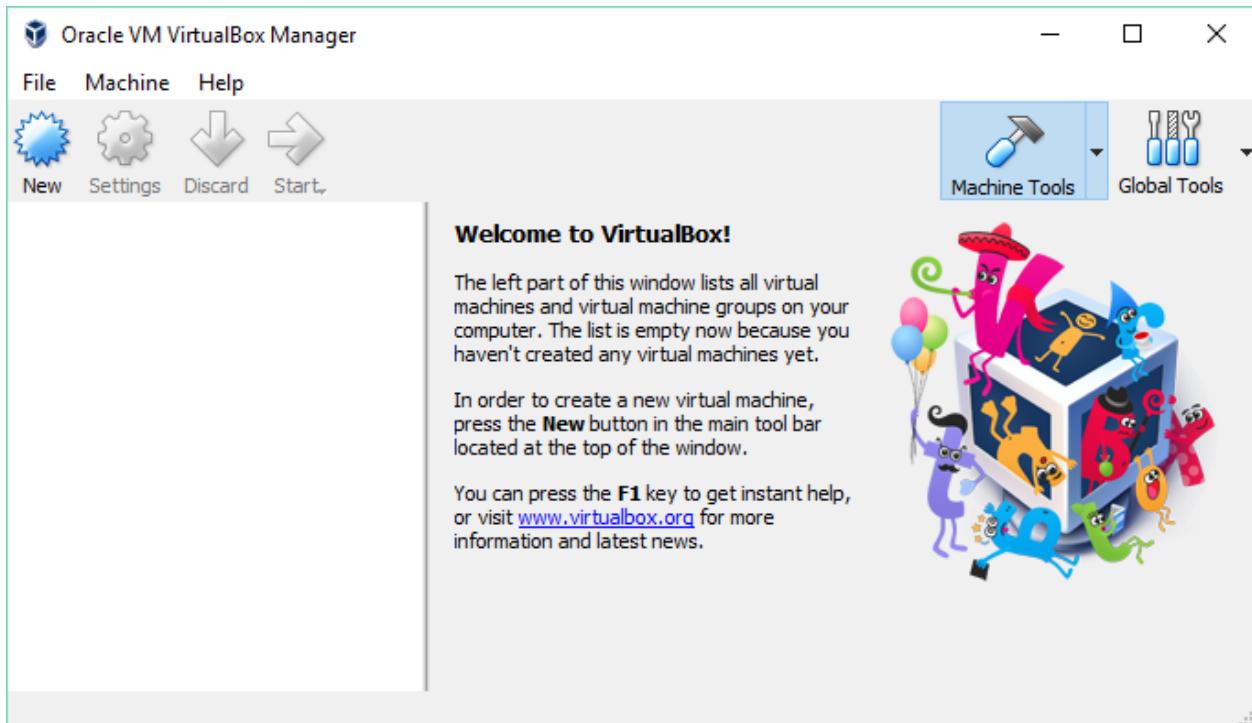
پاشه تو مخوبی عنیز

حدود هفت الی هشت گیگابایت هم دست کم نیاز دارید.

## نصب لینوکس روی virtual box

Vitrual box را باز کنید، روی new کلیک کنید





در قسمت Name اسم انتخاب کنید.

من دوست دارم اسمشو پذارم دلپر دلپر، تاج سدم،

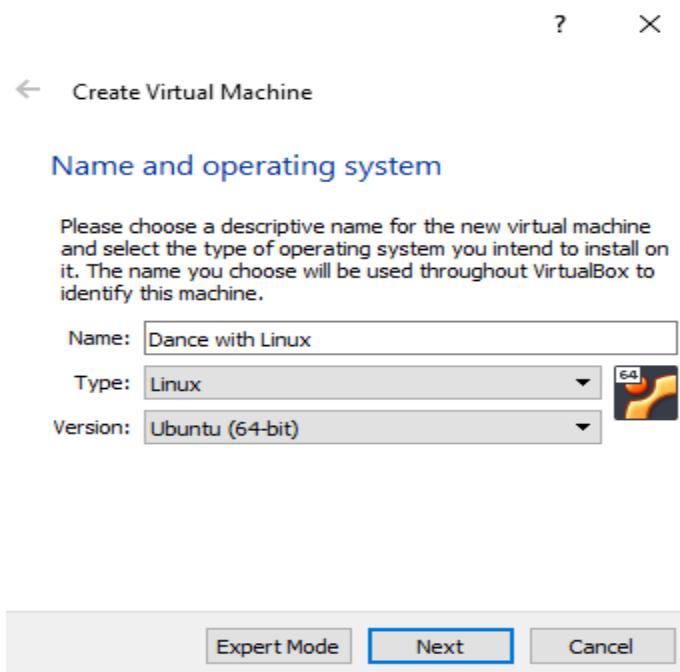
فکرت نمی ره از سدم....



پسه گرمون کردی. نخون

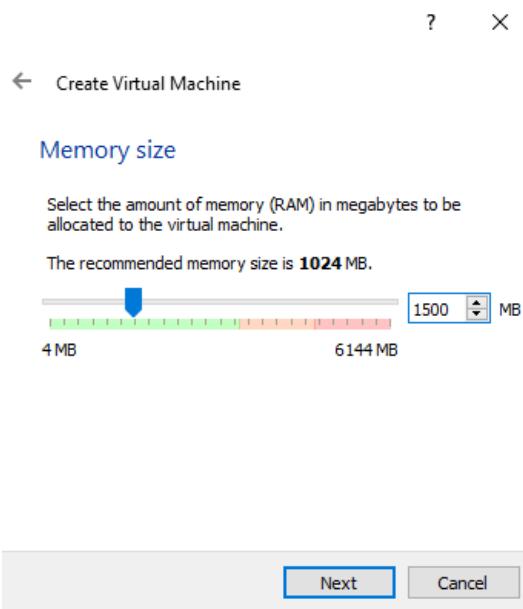


در قسمت `linux` را انتخاب کنید. در قسمت `version` مطابق چیزی که دارید

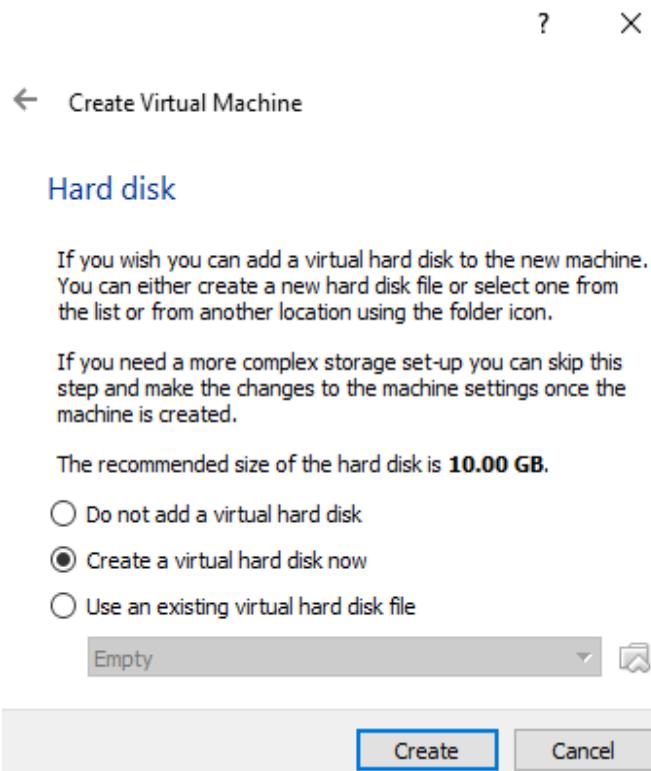


اگر هم distro شما در گزینه‌ها نبود، در گزینه‌های ابتدایی لیست، آخرین کرنل را انتخاب کنید.

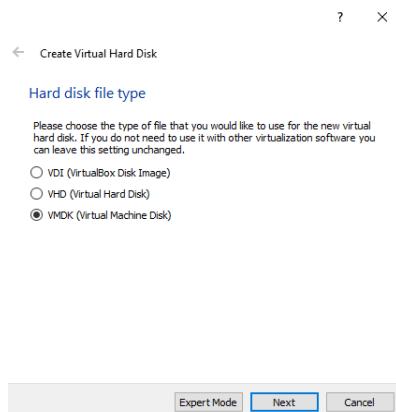
در قسمت Memory size RAM میزان Memory size را انتخاب کنید.



در قسمت create a virtual hard disk now روی Hard disk را بزنید.



اگر روی گزینه VMDK بگذارید، می‌توانید این از آن در دیگر سیستم‌هایی که ماشین مجازی دارند نیز استفاده کنید.



در قسمت dynamically storage on physical hard disk روی گزینه گزینه بگذارید.

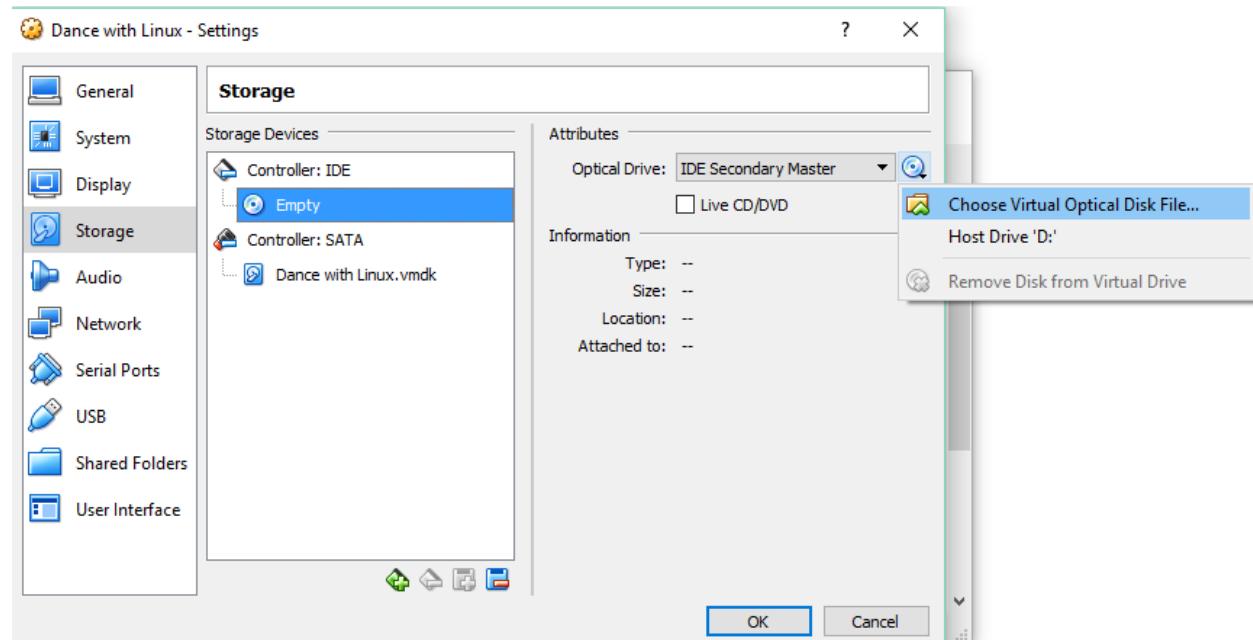
در قسمت بعدی یک اسم برای hard disk بگذارید. (یا همان اسم قبلی را بگذارید).



تا الان پیش زمینه آماده شد. یعنی ماشین مجازی شما آماده است. حالا باید ubuntu را نصب کنید.

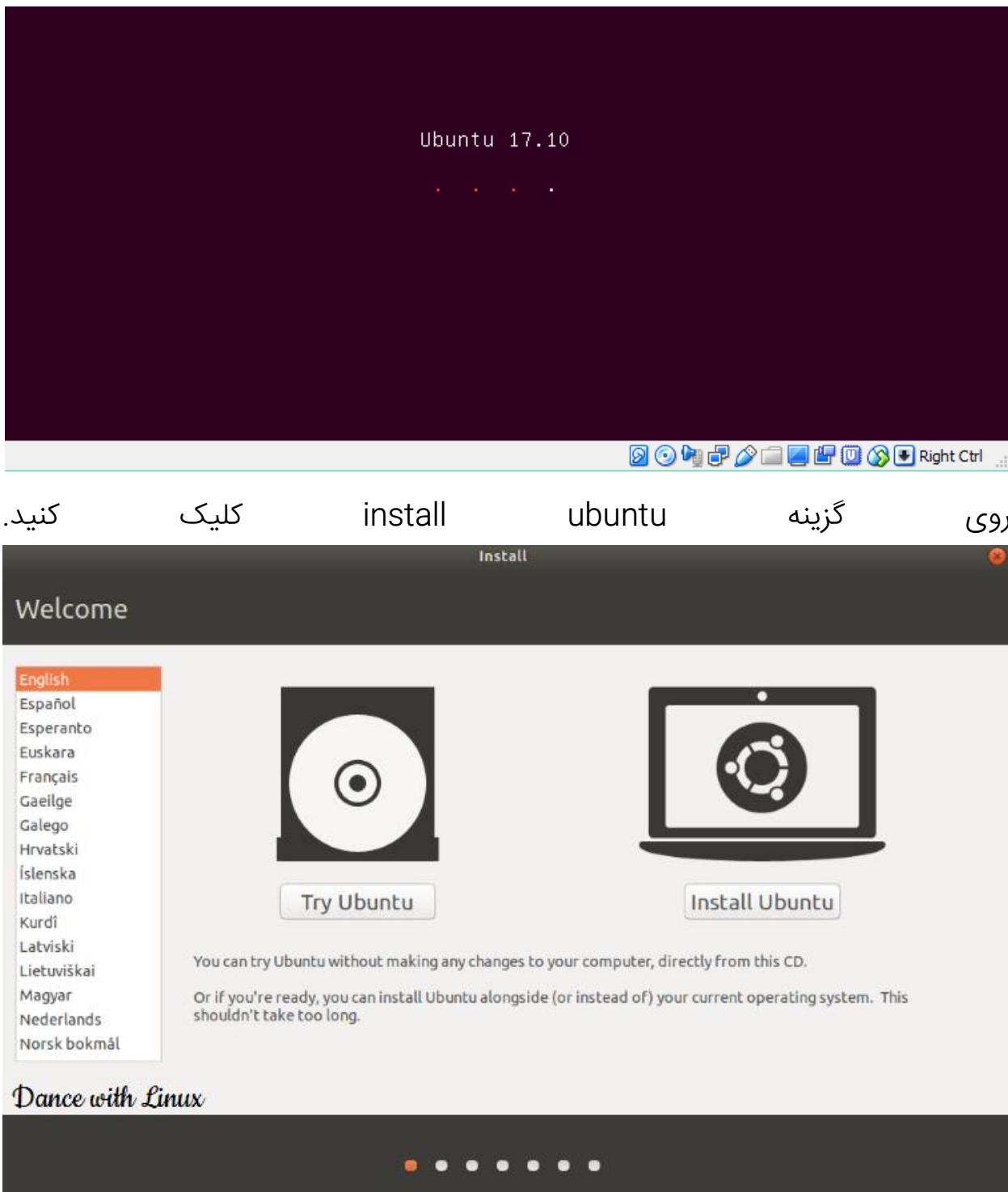
روی setting کلیک کنید. (بالا دومین آیکن)

در بخش storage روی Empty بزنید و در سمت راست یک آیکن CD می‌بینید، کلیک کنید و فایل iso که مربوط به فایل ubuntu دانلود شده است انتخاب کنید.



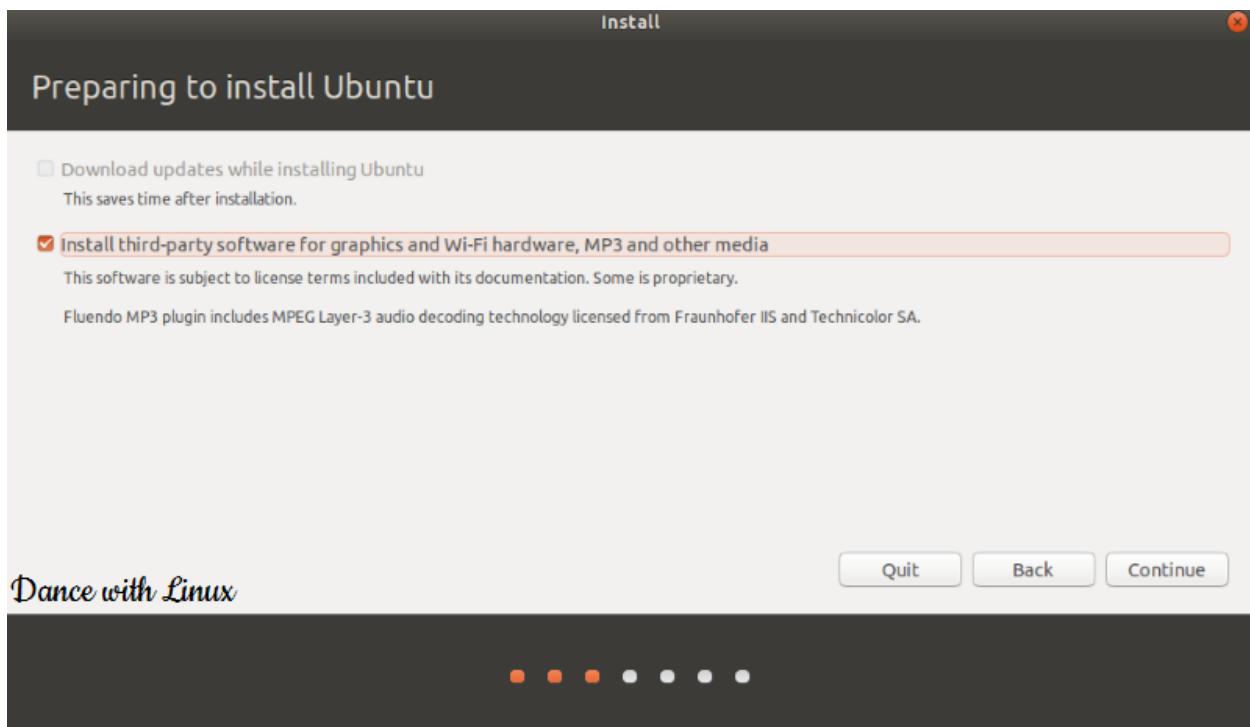
بعد روی continue و در آخر روی start بزنید.





در قدم بعدی اگر تمایل دارید آپدیت‌ها را موقع نصب دانلود کنید، سیستم‌تون را به اینترنت وصل کنید و تیک گزینه download updates while installing را بزنید.



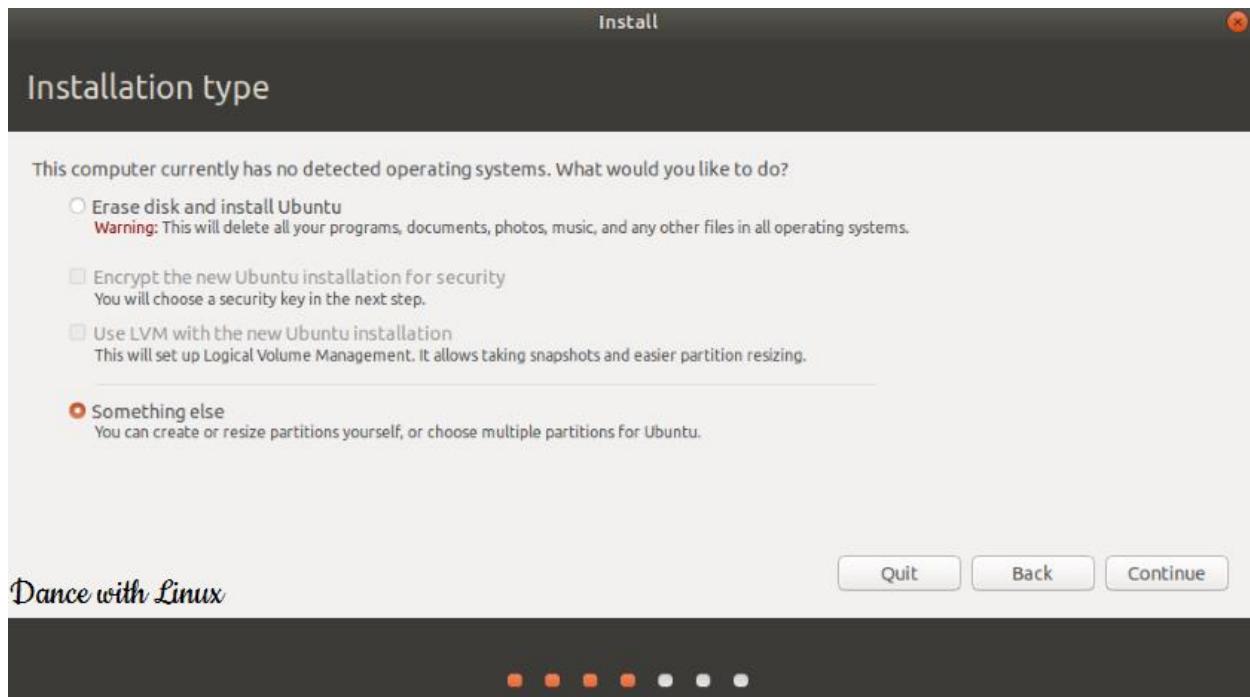


اینجا بخشی است که انتخاب می‌کنیم از چه پارتیشن‌بندی‌ای می‌خواهیم استفاده کنیم و چه فایل سیستمی مد نظر ماست.

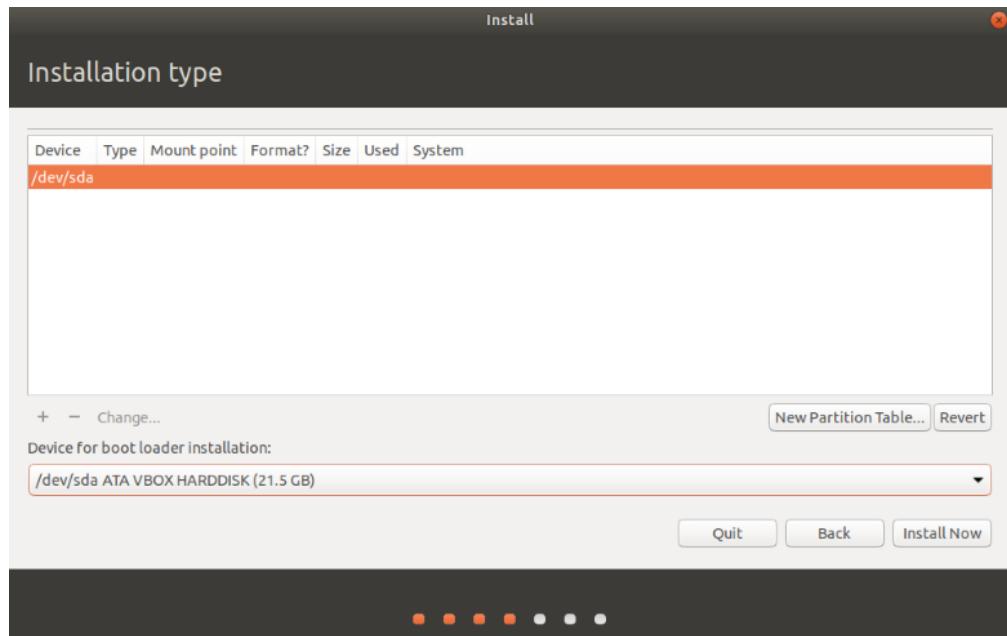
در گزینه اول، که دیفالت نصب هم است، به طور کامل هارد دیسک شما را پاک می‌کند و پارتیشن‌های مورد نیاز را نصب می‌کند. این گزینه بیشتر به درد کسانی می‌خورد که تازه وارد فضای لینوکس شده‌اند. با زدن این گزینه همه کار را خود نصب‌کننده انجام می‌دهد. منتهی برای توضیح برخی نکات ما گزینه sth else را زدیم.

**هشدار: اگر بصورت dual boot لینوکس را نصب کنید، با زدن گزینه اول، همه هارد دیسک شما را پاک می‌کند.**



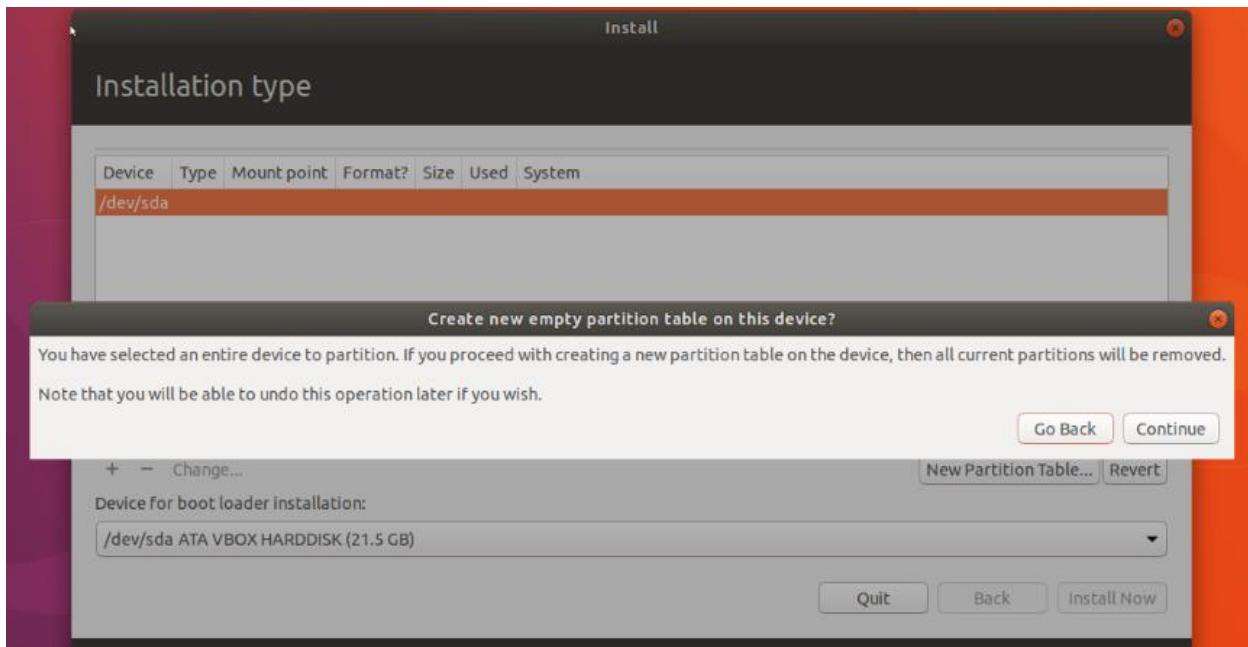


قسمت بعدی مربوط به لیست هارد دیسک‌های شماست. همه چیز در لینوکس به صورت فایل به شما ارائه می‌شود. dev در حقیقت فولدری است که سخت‌افزارها را به صورت فایل نشان می‌دهد و /sda هم مربوط به اسم هارد دیسک شماست.



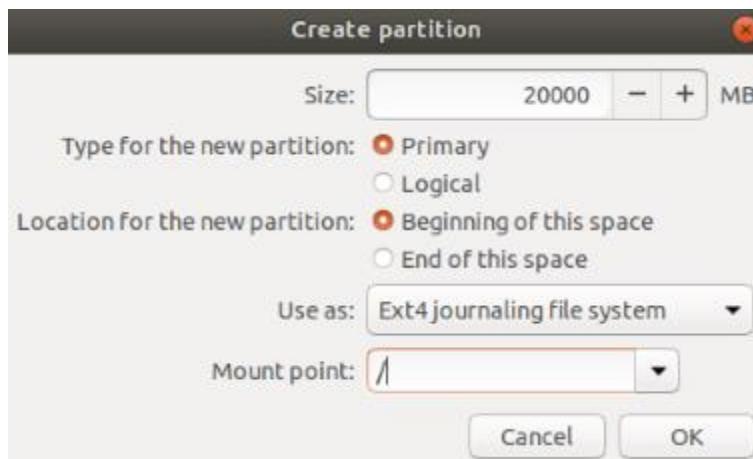
این هارد دیسک الان پارتیشن ندارد. روی new partition بزنید و بعد روی کانتینیو.





برای ساخت پارتیشن روی + پایین سمت چپ بزنید. اندازه پارتیشن را انتخاب کنید. نوع دیسک را روی primary و جای آن را روی اول دیسک یعنی beginning of this space بزنید. گزینه بعدی اشاره دارد که وقتی پارتیشن شما ساخته شد با چه فایل سیستمی فرمت شود. (بگذارید همان دیفالت یعنی Ext4 journaling file system بماند).

در قسمت Mount point یک اسلش یعنی / بذارید با این کار می‌فهمانید که می‌خواهید لینوکس شما در این پارتیشن نصب شود. اگر نگذارید، صرفاً یک پارتیشن ایجاد شده است که لینوکس روی آن نصب نمی‌شود و نمی‌توانید به مرحله بعد بروید.



اگر فضای خالی داشته باشید می‌توانید یک پارتیشن دیگر نیز بسازید. بهتر است یک پارتیشن برای swap بسازید.



Swap پچیہ؟ این دو میں پارہ دارہ می کیا ॥۱۰॥ اکھ فحشے پگو صدا پر نم ہیوا لاهائی (پچھے) پلا پیانی ॥۱۱॥

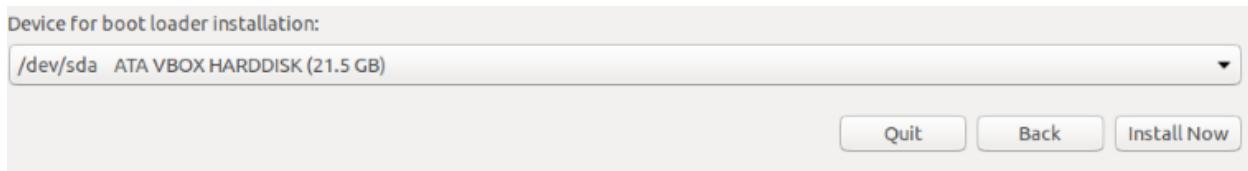
در یک سیستم عامل وقتی رم پر می‌شود، سیستم دیگر قادر به اجرای پردازش‌های جدید نیست. بنابراین نیاز به یک مموری خالی داریم. برای راهکار، اگر ما یک پارتیشن را به swap اختصاص بدهیم، لینوکس قادر خواهد بود که بخش اشغال شده مموری را به swap بفرستد تا جا برای اجرای پردازش جدید باز شود. توجه داشته باشید که از swap نمی‌توان به عنوان مموری اصلی استفاده کرد فقط بخشی را می‌توان به آن فرستاد و بعداً باید آن را برگردانیم به سرجایش. اگر سیستمی کارش به swap بیافتد بسیار کند خواهد شد و این هشداری است برای اینکه بدانید باید فکری به حال حافظه یا افزایشش بکنید.

برای ساختن این پارتیشن، این بار روی logical گزینه swap use as را گزینید.



در قسمت بعدی باید انتخاب کنیم که boot loader روی کدام پارتیشن نصب شود. در واقع boot loader لینوکس را load می‌کند. گزینه را روی هارد دیسک بگذارید. . (دلیل فنی: نیاز دارید boot loader روی سکتور صفر قرار بگیرد).





در آخر روی گزینه install و بعد continue بزنید.

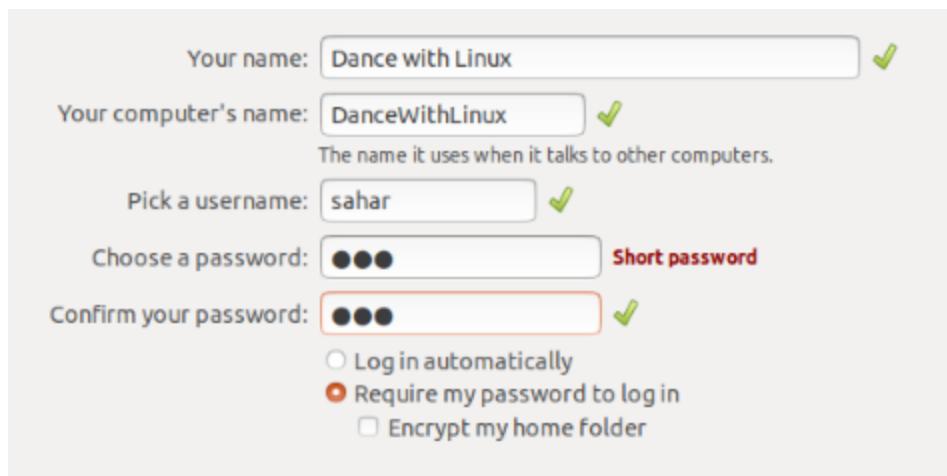
کشور را انتخاب کنید.

توصیه: کیبور را روی انگلیسی بگذارید. بعده می‌توانید کیبور فارسی را اضافه کنید.

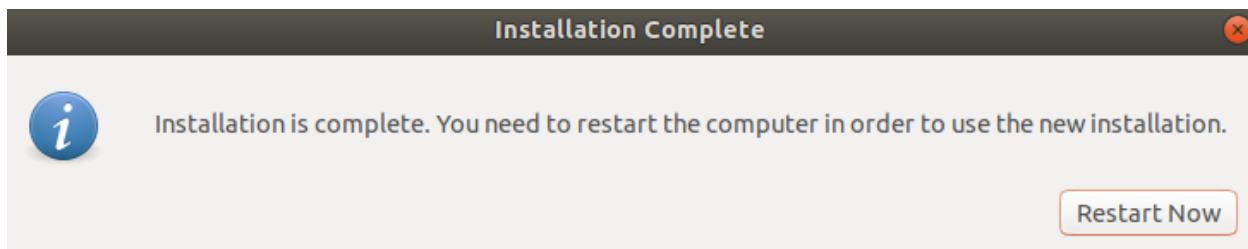
حالا باید یک یوزر بسازید.

یک پسورد انتخاب کنید.

توصیه: پسورد ساده‌ای انتخاب کنید چون بارها اتفاق می‌افتد که موقع اجرای کامندی، باید پسورد خود را وارد کنید. با انتخاب گزینه log in automatically موقع ورود به سیستم از شما پسورد نمی‌خواهد اما با انتخاب گزینه بعدی، از شما واردکردن پسورد را می‌خواهد.

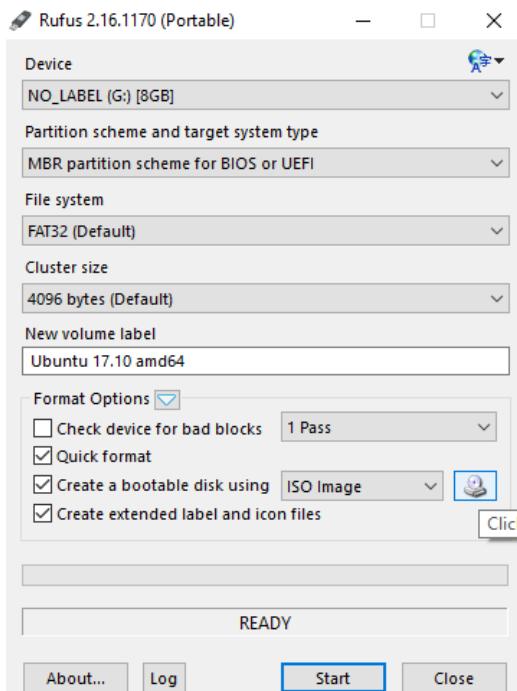


در انتهای هم پس از نصب پیغام restart سیستم را می‌بینید.



## نصب لینوکس روی سیستم

برای نصب لینوکس روی سیستم، بیش از هر چیز، نیاز دارید یک فلش یا DVD ای را بوتیبل کنید. برای این کار بسیاری از نرم افزارها وجود دارند. مثلاً ما از Rufus استفاده کردہ‌ایم. بعد از دانلود نرم افزار آن را باز کنید.



در قسمت Device فلشی که برای بوتیبل کردن انتخاب کردید را انتخاب کنید!



عینْ حیاپانی حرف می‌زنی! انتخاب رو انتخاب می‌کنن دیگه.

برای نحوه پارتیشن‌بندی هم، MBR را انتخاب کنید و برای فایل سیستم هم FAT32. در انتهای فایل ISO دانلود Ubuntu را از آیکن جلویی ISO Image انتخاب کنید و در انتهای start در ادامه هشدار می‌دهد که تمام فایل‌ها فلش شما پاک خواهد شد. دو تا اوکی بدین و تموم.



بعد سیستم را ریسٹارت کنید و فلش را بوت کنید. (یعنی موقعی که صفحه مشکی اومد جلوتون یکی از کلیدارو بزنید می‌ره و فلشتون رو بوت می‌کنه. اگر نشد، یعنی در قسمت تنظیمات بوت شما، گزینه این که اول فلش را بوت کنه فعال نیست، اونو فعالش کنین.)

بعد از بوت کردن فلش، مراحل نصب به ترتیبی است که در تصاویر بالا، از قسمت `install` آمده است.



## فصل دوم: قلب لینوکس



معرفی کرنل

کار با shell



## شروع کار با دسکتاپ

دسکتاپ یکی از شخصی‌ترین بخش‌های زندگی یک آدم به شمار می‌رود! نحوه چینش آیکون‌ها و اینکه چه فایل‌ها و برنامه‌هایی روی صفحه باشند یا حتی اینکه از چه دسکتاپی استفاده کنیم، همگی تابع سلیقه آدم‌هاست بیشتر.



شُنیدی که چی گفت؟ گفت دسکتاپ شخصی‌ترین  
بخشن کامپیوٽر.

مثل مسواج



دلپنده مسواک



پس من عکس‌ای شخصی آینده خودم و لینوکس‌چان و  
پچه‌های من رو می‌ذارم روی دسکتاپ. رست‌های این شکلی  
هم می‌گیرم



خدری دیگه خدر

پس چی. من اقتدار می‌کنم به خدر پوندم. اصلا هر کی  
بعز من پگه خدم، پاهاش قانونی مطابق قانون خودم  
پدمنورد می‌شه.



چون لینوکس یک سیستم چند کاربره یا multiuser به حساب می‌آید برای ورود به سیستم اولین کاری که می‌کنید لاگین کردن است.



وقتی لاگین می‌کنم دقیقاً چی می‌شه؟



سوالو نگاه، وقتی په مخونتون می رسی دقیقاً په چې دسترسی پیدا می کنی؟



هر چې با من بحث نکن، فرق داره.



چه فرقی؟

وقتی از پیدون میام خونه یا همین لاگین می کنم او لین  
چیزی که پهش دسترسی دارم سرویس پهداشتیه،

ولی وقتی خونه پاشم و لاگین کنم په خونه، او لین چین  
میتونه خیلی چیندا پاشه. بحث فلسفیه، از درک تو خارجه.  
باید توی موقعیتش پاشی تا پغومی، اصلا پیا پریم  
خونمنون پهت پکم.



الان که فکر می کنم می پینم فلسخت خیلی سنگینه، همینجا  
می موئم ولی همین که جای دستشویی گفتی سرویس  
پهداشتی چای شکرش باقیه، این فلسقتم نګه دار په لینوکس  
پکو و پیدتاش خونت نه من، شعور می خواهد که از تو کلا خارجه.





چوای پیچه رو بدہ

چشم تو فقط عصبی نباش. وقتی لاگین می کنی تقدیبا همین وضعیت که خرده گفت. یعنی وقتی لاگین می کنی می تونی به shell و دسکتاپ مختص به خودت و کلی اچاره های دیگه دسترسی داشته باشی.



چون لینوکس یک سیستم چند کاربره یا multiuser به حساب می آید برای ورود به سیستم اولین کاری که می کنید لاگین کردن است. شاید به طور خلاصه بتوان گفت که عمل login کردن سه کار اصلی را انجام می دهد:

- 1- شما را به عنوان یک کاربر خاص شناسایی کرده
- 2- Shell یا پوسته و دسکتاپ مختص به خودتان را ارائه می دهد
- 3- مجوزهایی را برای دسترسی به فایلها و یا تغییر آنها را ارائه می دهد. چون هر کاربری متناسب با اینکه چه نوع کاربری است، سطح دسترسی متفاوتی دارد.

می‌خوام دستشویی رو از سطح  
دسترسیت مخارج کنم ۳ا فلسفة پنهانی





عچب خریه‌ها، کی اصلا همچین دسترسی رو په  
تو میده؟



هدپی! اختیار خونه مخدمو که دارم



خونت نمیام

پس عروسیم نیا، چون اونجا دیگه خیلی چیز ارو از سطح  
دسترسیت خارج می کنم، مثلًا اون دختره... آره همون، با شد ک  
آشناش می کنم... یعنوری از سطح دسترسی خارچت می کنم که  
همش پوچ اشغال پذنی



چرا کلا بحث لاغین را مطرح کردیم؟ چون در لینوکس می‌توان چندین سطح دسترسی تعریف کرد اما قصد اصلی این بوده که بتوانیم تفاوت بین یوزر عادی یا regular user را با کاربر روت یا root user را بیان کنیم. (به کاربر روت، کاربر ریشه هم گفته می‌شود.)

کاربر عادی: شخصی است که از سیستم استفاده می‌کند. یک نام کاربری منحصر به فرد و کلمه عبور مختص به خودش را دارد. یک سیستم می‌تواند چندین کاربر عادی داشته باشد. به طور کلی، کاربر عادی تنها اجازه استفاده از سیستم را دارد.

کاربر root: هر سیستم لینوکس، فقط یک کاربر root دارد. این کاربر اجازه دسترسی‌های خاص را دارد و اگر به عنوان کاربر root لاغین کنید باید خیلی مواظب باشید چون اشتباهات در این ناحیه صدمات جدی دارد. به کاربر root اصطلاحا super user نیز گفته می‌شود.



این کاربر کنترل کامل یک سیستم‌عامل در اختیارش است. توانایی اینکه هر برنامه‌ای را باز کند یا اجرا کند یا پاک‌کردن یوزرهای دیگر را نیز دارد.



چرا پاید پینمون فدق وجود داشته باش؟ این فاصله طبقاتی توی لینوکس چیه که راه اندامتین؟ لابد فردا حسن نژاد پرسنی تون کل می‌کنه و ما رو هم پیدون می‌کنید. اصلا اینو چوای پده؛ از یه کامپیوتر که فقط یه نفر استفاده می‌کنه چرا اکانت‌ها متفاوت پاید وجود داشته باش؟

برای اینکه کاربر عادی که هر روز از لینوکس استفاده می‌کند اشتباهها فایل‌های حیاتی سیستم را تغییر ندهد. در واقع این کار بیشتر برای آسایش است! در غیر این صورت یک اشتباه می‌تواند به طور کلی سیستم را به باد فنا دهد. به همین دلیل بجز کاربر عادی یک کاربر root هم دارید.

جدا از این این بحث کاربرها این امکان را فراهم می‌کند که اگر چندین نفر از یک سیستم لینوکس استفاده می‌کنند، بتوانند کاربرها در صورت تمایل، فایل‌هایشان را از دسترس قرار گرفتن یا تغییر داده شدن توسط کاربرهای دیگر محافظت کنند.

شبکه بندی احتمالا بهترین دلیل برای بکارگیری یک سیستم لینوکسی می‌باشد. این داشتن نام کاربری یا اکانت مجزا و منحصر به فرد چندین فایده می‌تواند داشته باشد. مثلاً اکانت شما می‌تواند با سایر منابع کامپیوترها در ارتباط باشد، مثل ارتباط داشتن با سیستم فایل‌ها، نرم‌افزارها، mailbox‌ها و ...

در اکثر مواقع، پیش می‌آید که شما اکانت خودتان را به اصطلاح شخصی‌سازی می‌کنید. برای مثال ممکن است برای اسمی فایل‌ها، نام‌های استعاری به کار ببرید. یا برنامه‌هایی



تنظیم کنید. با جمع‌آوری این اطلاعات در یک نقطه، انتقال دادن اکانتتان یا اضافه کردن یک اکانت جدید به یک کامپیوتر دیگر در آینده آسان می‌شود و سردرگم نخواهد شد.



نمی‌خوای ما رو پگردونی توی فضای دسکتاپ و چه  
کشتنی پذیریم؟

پارونم میاد؛ چه فضای عاشقانه می‌شد. من پاشم، لینوکس  
پاشه، توی حیاط دسکتاپ راه پریم... پارک هم پود... البته  
کشتن ارشاد مال ما نیست مال آدماس.



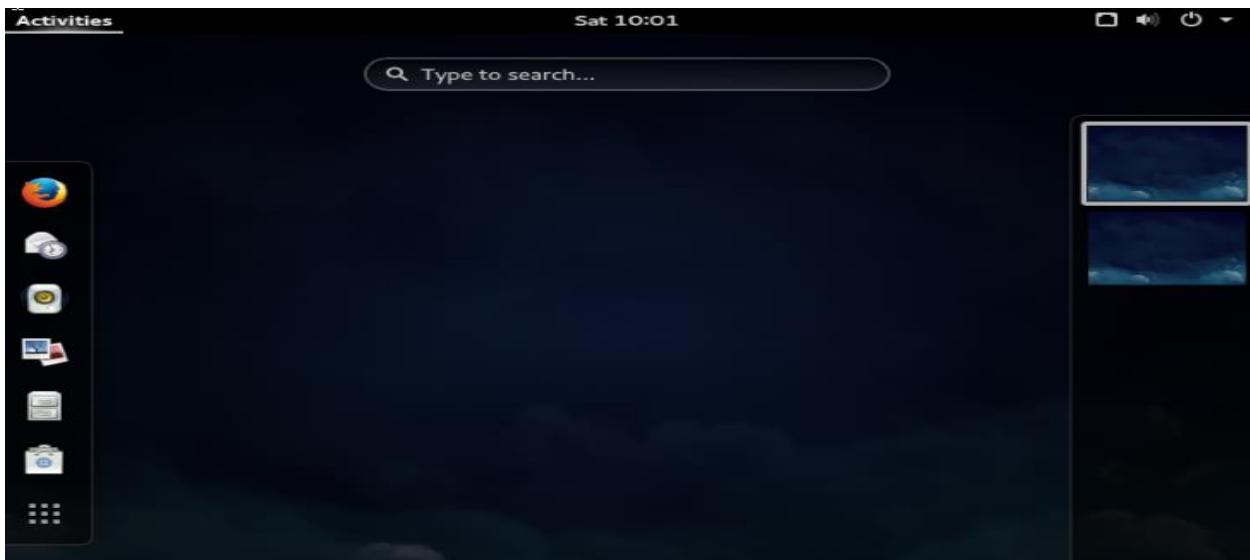
## قدم زدن در فضای دسکتاب

محیط گرافیکی یا GUI در واقع همان چیزی است که وقتی Desktop بالا می‌بینید. وقتی بحث محیط‌های گرافیکی می‌شود، برخلاف ویندوز که یک محیط گرافیکی واحد دارد، لینوکس این چنین نیست. در لینوکس‌های مختلف، این محیط‌های گرافیکی متفاوت است اما در ویندوز این چنین نیست. مثلاً اگر از ویندوز 10 استفاده می‌کنید محیط گرافیکی شما با تمام کسانی که از ویندوز 10 استفاده می‌کنند یکی است. اما دسکتاپ‌ها در دنیای لینوکس بسیار متنوع‌اند و این حق انتخاب برای تمامی کاربران وجود دارد.

برای بیان بهتر این موضوع، چندین تصویر از محیط‌های دسکتاپی لینوکس یا desktop enviroment در ادامه آورده شده است. اینکه کدام بهتر است بسته به سلایق افراد تغییر می‌کند. شاید کسی از محیطی خوش‌بیاید و همان محیط برای فرد دیگر مزخرف باشد.

GNOME 3



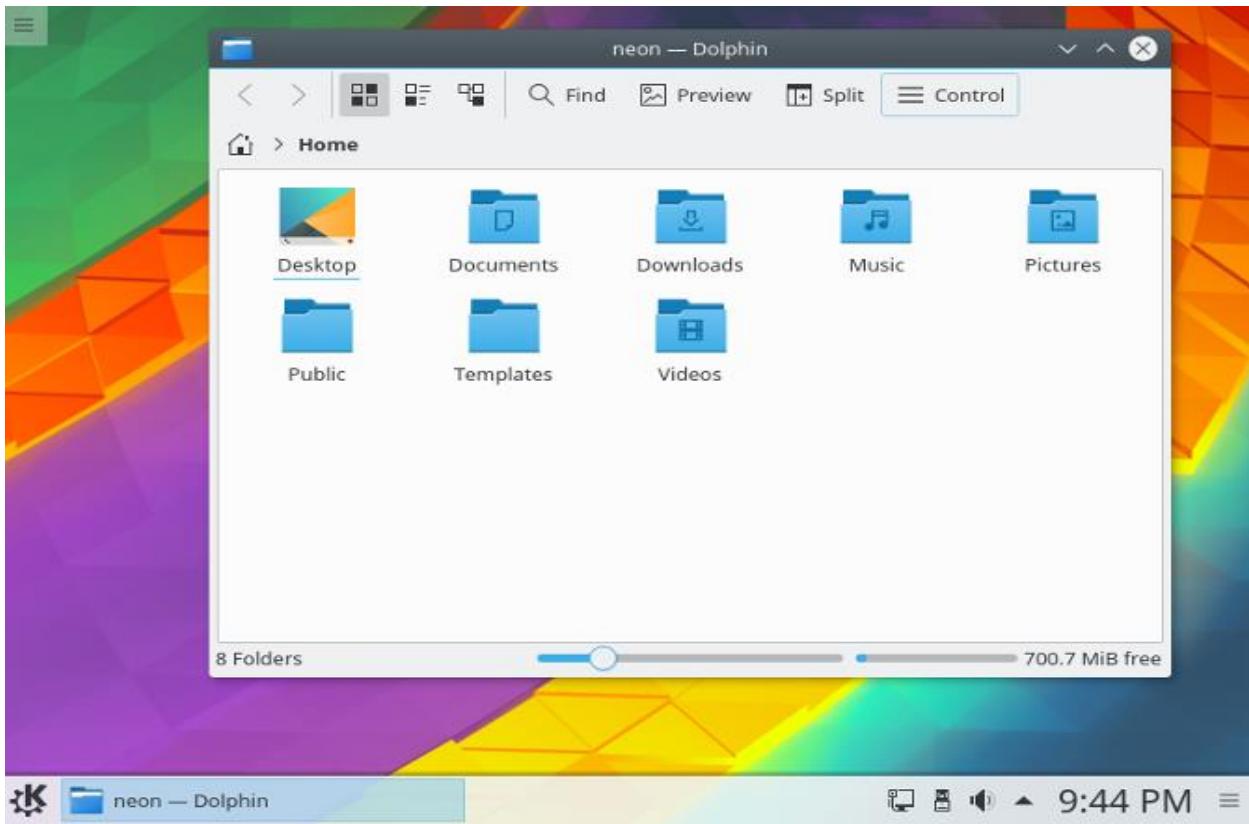


Ubuntu unity

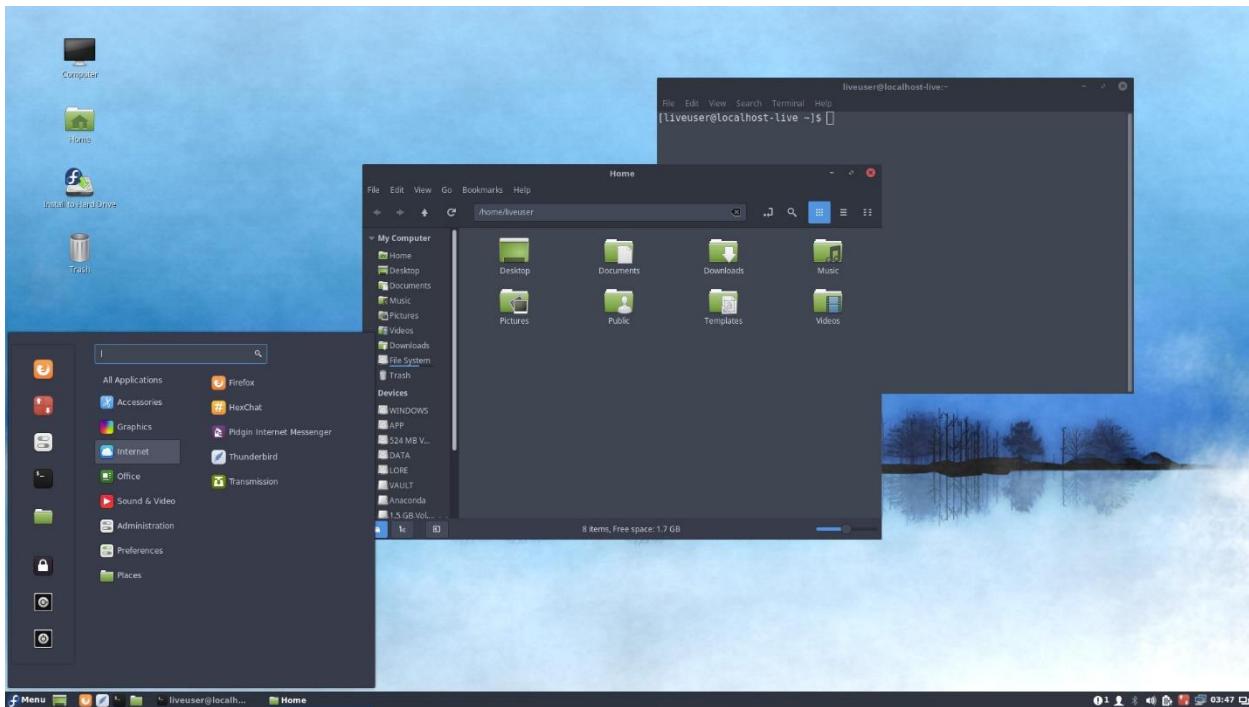


KDE



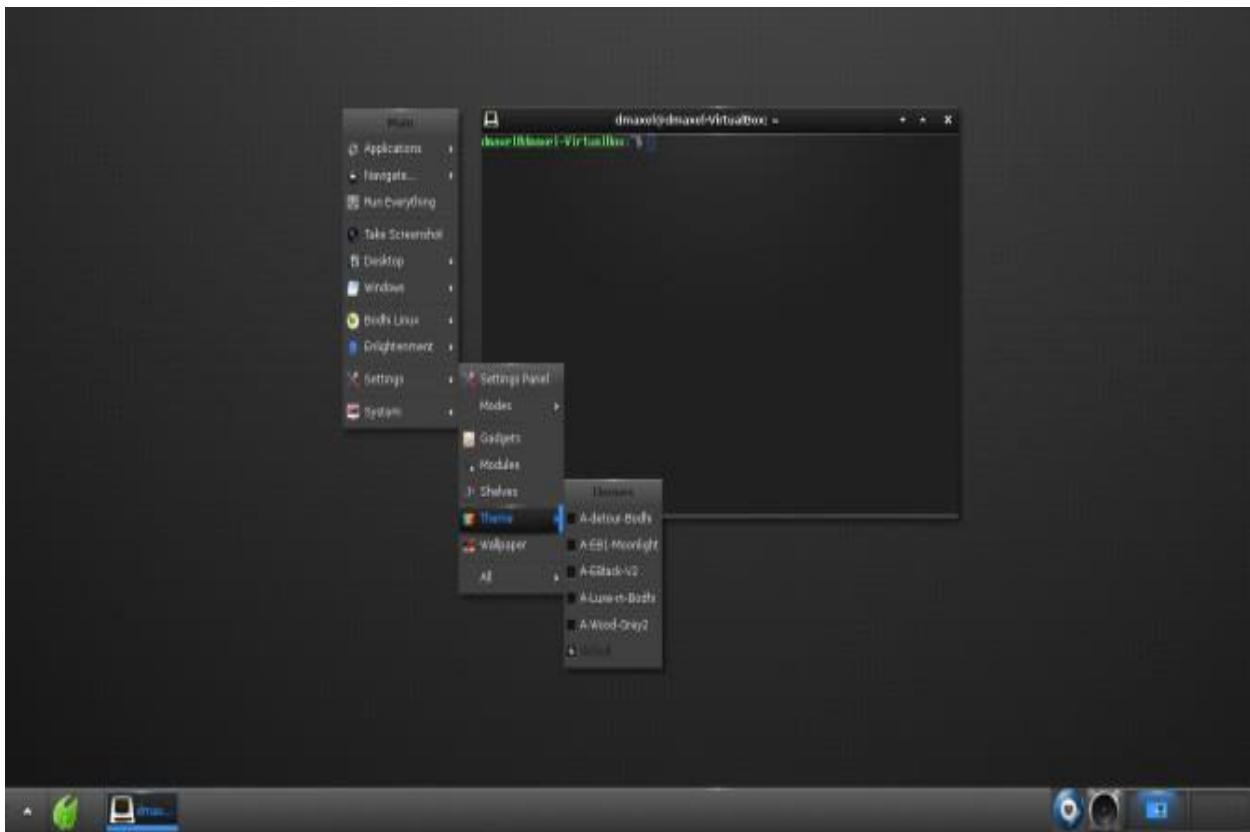


Cinnamon

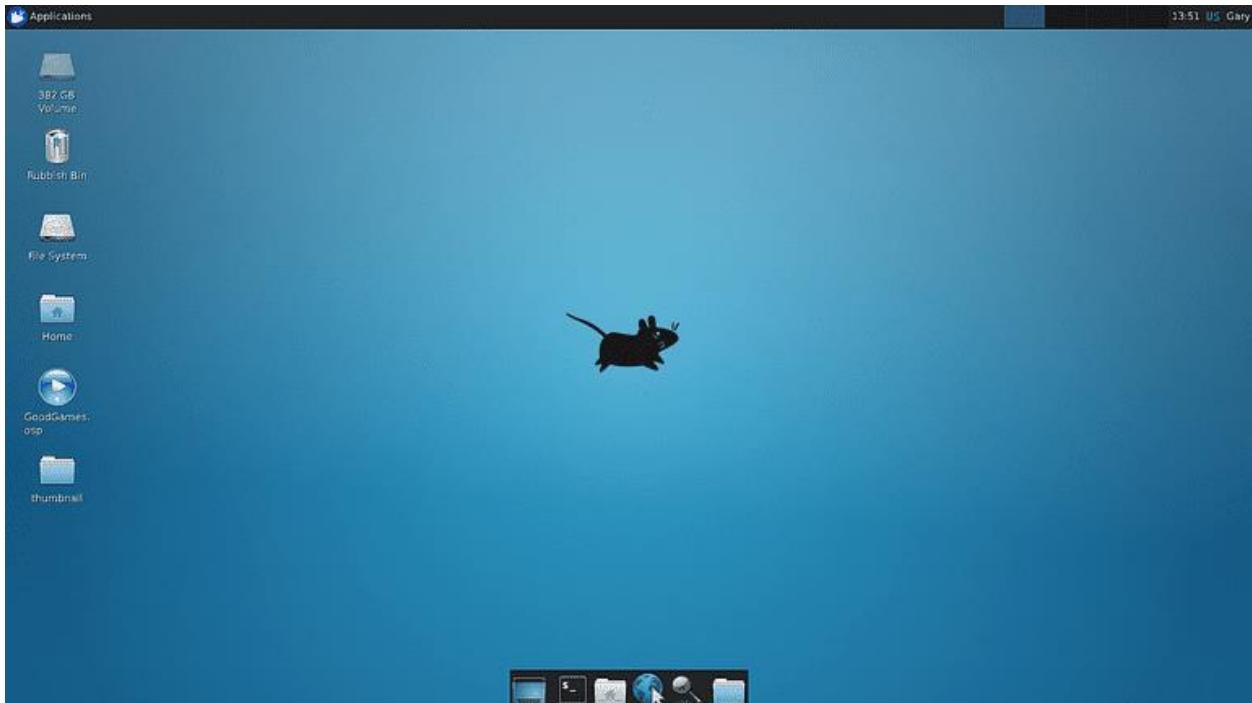


Enlightenment

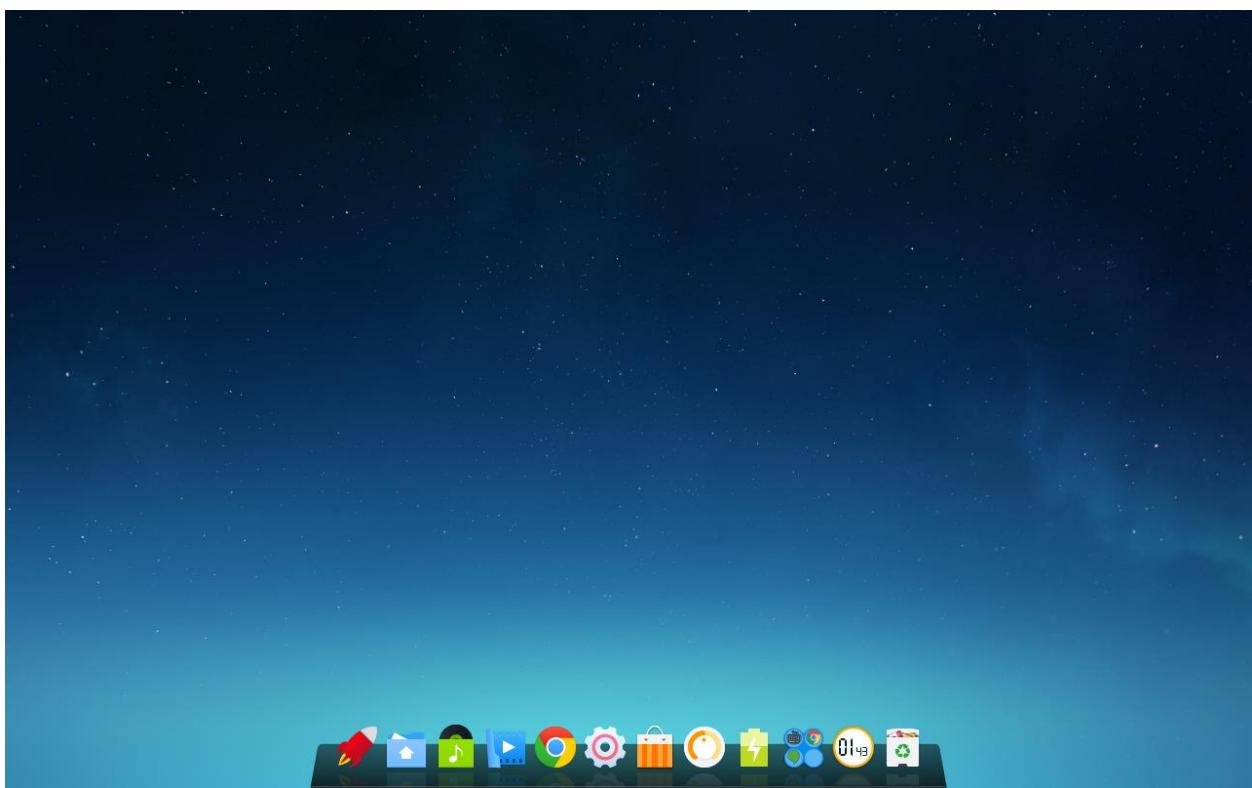




Xfce



Deepin

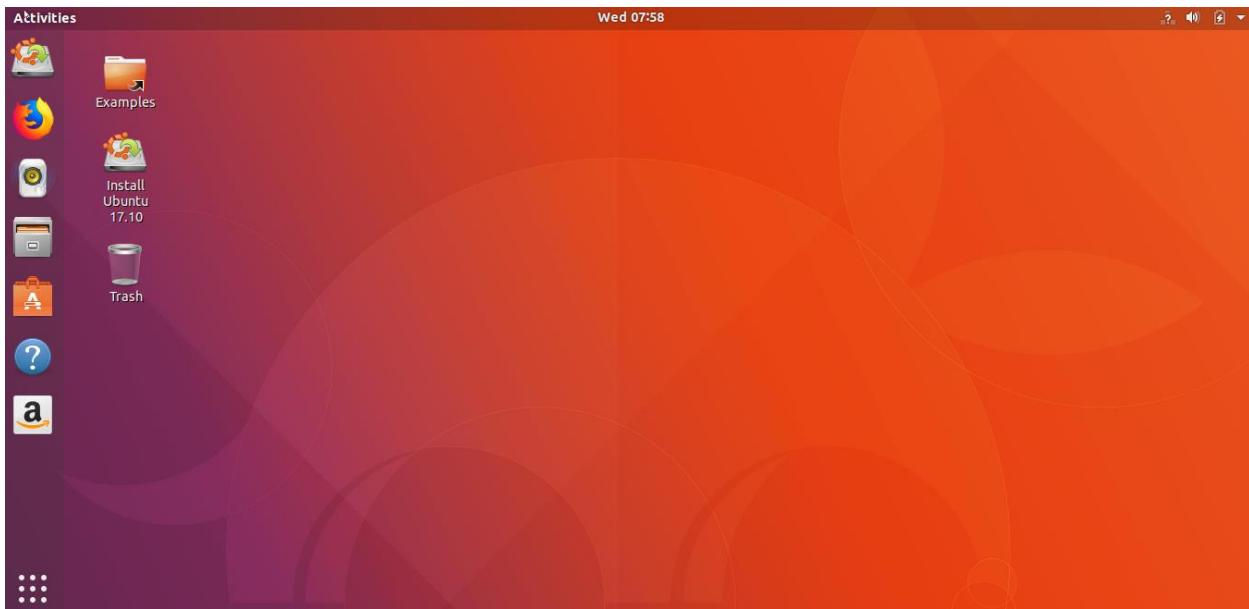


## Pantheon



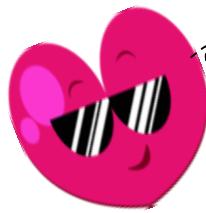
در نسخه اوبونتو 17، به جز اینکه ubuntu desktop با محیط Unity ارائه شده است، نسخه‌های مختلفی هم ارائه شده‌اند. مثل: Kubuntu که به جای محیط گرافیکی Unity، Lubuntu، ubuntu GNOME، KDE استفاده می‌کند و نسخه‌های دیگری مثل: Xubuntu، ubuntu Budgie ، ubuntu studio،Ubuntu Kylin





با کمی کارکردن با این محیط به راحتی می‌توانید شروع به استفاده از لینوکس کنید و کارهای ساده‌ای انجام دهید. استفاده از این محیط گرافیکی نیاز به توضیح خاصی ندارد. اما این محیط گرافیکی نمی‌تواند به تنها‌ی تمامی کارهای ما را انجام دهد و محدودیت‌هایی را دارد. پس باید از یک محیط قدرتمند دیگری استفاده کنیم.





## کرنل (kernel)

می‌دانیم که در قلب سیستم‌عامل، کرنل حضور دارد. کرنل در واقع نرم‌افزاری است که به application‌ها اجازه می‌دهد تا با سخت‌افزار یا Hardware کامپیوتر در ارتباط باشند و با سایر برنامه‌های درحال اجرا روی همان سیستم‌عامل همکاری کنند.

سخت‌افزار در واقع هر چیزی است که قابل لمس باشد مثل: مموری و ...

برای اینکه کرنل بتواند با سخت‌افزار کامپیوتر در ارتباط باشد باید چندین پیش‌نیاز را داشته باشد:

1- مطالبی را دربارهٔ درایورهای سخت‌افزاری واقع بر کامپیوتر را بداند: در لینوکس، کدی که به application‌ها اجازه می‌دهد تا با هر قطعه سخت‌افزار در ارتباط باشند، یا در داخل کرنل بصورت درایور ساخته شده است یا بعد از اجرای کرنل با استفاده از چیزی به نام ماژوال قابل بارگذاری، به کرنل اضافه می‌شوند.

2- باید از عهده مدیریت application‌ها بربیاد: کرنل باید تصمیم بگیرد که چه پردازش‌هایی اлан در حال اجرا هستند و این پردازش‌ها به کدام پردازنده کامپیوتر می‌توانند دسترسی داشته باشند و این دسترسی چقدر طول بکشد.

3- باید سیستم فایل‌ها و اجازه دسترسی را تشخیص دهد: کرنل مسئولیت‌های سنگینی را به عهده دارد و چون مسئول این است که به پردازش‌ها اجازه خواندن (reading)، نوشتن (writing)، run فایل‌هایی که در انواع مختلف سیستم‌فایل ذخیره شده‌اند را بدهد، پس باید ساختار این سیستم فایل‌ها را بداند. جدا از این باید اجازه هر کاربر را بررسی کند تا بداند هر کاربر اجازه دسترسی به کدام فایل‌ها را دارد.



حوالستو چمع کن از این په بعد مخنوم آیندم ایجاز تو تعیین  
می‌کنه. دسته از پا مخطا کنی میدم شیفت دیلیت کنه.



گیری کردیما. آرزو پر چوایان عیپ نیست. چه حساب کردی  
روی خانوم آیند!



حالا که توانایی لازم برای این مسئولیت بزرگ را به تنها یی نداریم (!) پس کرنل یک interface ای ارائه می‌دهد که با استفاده از آن توانایی این را داریم که با سخت‌افزار صحبت کنیم و ارتباط برقرار کنیم ولی غیرمستقیم.

کسی که این لطف را در حق ما می‌کند، shell نام دارد که رابط بین ما و کرنل است.



په قول بعضی‌ها، توی پلا دکف پهش می‌گن shell  
فارسیش چی می‌شه؟ ترکی پیلیر سنه؟



شل یا پوسنه هم پهش می‌گن

Shell یک مفسر کامند لاینی (command line) است که به شما اجازه دسترسی به برخی از حیاتی‌ترین ابزارهای لینوکس را می‌دهد.



آخ چون کی شروع می‌شه؟ می‌خواهم توش پنویسم I love  
الا و پقدستم پدای عیال شاید بینه



روشی برای اجرای برنامه‌ها، کار با سیستم فایل، کامپایل کد کامپیوتر و مدیریت کامپیوتر ارائه می‌نماید. اگرچه shell از GUI‌های مرسوم محبوبیت کمتری دارداما بخش زیادی از حرفه‌ای بودن یک فرد در لینوکس بستگی به استفاده او از کامندلاین‌ها در shell دارد. شاید در ابتدا قضیه کمی سخت جلوه کند اما اگر به درستی یاد بگیرید تبدیل به بزرگترین سرگرمی شما خواهد شد. شروع یک دنیای هیجان‌انگیز است که پایان ندارد.

Shell‌ها انواع مختلفی دارند. گرافیکی، CLI که به آن‌ها text یا command line یا گفته می‌شود. بسیاری از لینوکس‌ها هم گرافیکی shell دارند هم text تکست.

قبل‌اهم اشاره شد که منظور از گرافیکی shell، همان دسکتاپی است که اغلب بعد لاگین کردن می‌بینید و گفتیم که این محیط، در هر سیستم لینوکسی‌ای می‌تواند متفاوت باشد. اما چیزی که می‌خواهیم اضافه کنیم این است که اصلاً ممکن است سیستم لینوکسی شما محیط گرافیکی نداشته باشد مثل برخی از لینوکس سرورها. در این صورت باید حتماً از text shell استفاده کنید. بنابراین **تمرکز خود را روی یادگیری محیط‌های گرافیکی نگذارید**. چون اصولاً قدرت خاصی به شما نمی‌دهند.

### چگونه به یک shell دسترسی داشته باشیم؟

این بستگی دارد به اینکه سیستم شما محیط گرافیکی یا اصطلاحاً GUI دارد یا ندارد:

(مثل برخی از لینوکس سرورها) بعد از لاگین کردن، به طور مستقیم کارتان با shell شروع می‌شود و نیازی به زحمت خاصی نیست.

(مثل ubuntu که داریم استفاده می‌کنیم) باشد terminal With desktop (راست کلیک روی دسکتاپ و گزینه open terminal یا در بین application‌ها پیدا کنید یا از شورت کات ctrl+alt+t استفاده کنید.)





.Shell در واقع یک کلمه کلی است؛ درست مثل زمانی که می‌گوییم مرورگر یا browser یعنی یک مرورگر انواع مختلفی دارد مثل: کروم، فایرفاکس، اپرا و ...

Shell نیز داستانی این چنینی دارد. یعنی shell یا پوسته، انواع مختلفی دارند اما shell‌ای که به طور دیفالت روی همه لینوکس‌ها حضور دارد، bash است. Bourne Again Shell مخفف Bourne با اولین shell یونیکس سازگاری دارد. (یعنی پوسته Bourne که به وسیله دستور bash نمایش داده می‌شد).

حالا این حقیقت رو به ما نمی‌کفتی  
می‌ترکیدی؟ گیچمون کردی



نرم‌افزار ترمینال (terminal)، نرم‌افزاری است که در محیط گرافیکی اجرا می‌شود و در پشت این پنجره ترمینال، bash در حال اجراست. پس هر چیزی را که در ترمینال تایپ کنید به bash می‌رساند. هم آن را اجرا (run) کرده و نتیجه را در پنجره ترمینال به شما نشان می‌دهد.



خوبی ترمینال این است که چون در محیط‌های گرافیکی اجرا می‌شود توانایی این را داریم که اسکرول کنیم یا فونت و رنگ را عوض کنیم یا حتی چندین پنجره ترمینال باز کنیم.

پنجه‌هارو پاز کن عشق رو پیار په خونه ڦا که ڦماری عشق  
پخونه عاشقونه غمارو رها کن په فردا نگاه کن عشقو صدا  
کن... عشقو صدا کن



پا خرد گشتی؟

به طور دیفالت در اوبونتو، سه تا ترمینال نصب است:

Xterm -1

Uxterm -2

GNOME terminal -3

محیط‌های گرافیکی که GNOME است (مثل اوبونتو) ترمینال دیفالت آنها GNOME terminal است.

لینوکس‌هایی که محیط گرافیکی آنها KDE است، ترمینال دیفالت آنها، کنسول نام دارد و هر کدام ویژگی‌های خاص خود را دارد.

بعد از اینکه shell را بالا آوردید (مثلا ترمینال در اوبونتو)، اولین چیزی که می‌بینید، خط اعلان یا prompt نام دارد.

```
Terminal
File Edit View Search Terminal Help
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

sahar@dance-with-linux:~$
```



علامت prompt برای یک کاربر عادی، فقط یک علامت دلار یا \$ است. قبل اهم گفته بودیم که کاربر عادی قابلیت مدیریت سیستم را ندارد و تنها می‌تواند از سیستم استفاده کند.

علامت prompt برای یک کاربر root یا super user، علامت # است و می‌تواند به فایل‌های حیاتی دسترسی داشته باشد (کمی خطرناک است(!)) این کarakترها و علامت‌ها را خودتان هم می‌توانید تغییر دهید.

من می‌خواهم عکس سه در چهار لینوکس رو پذارم.  
پول که نداریم توی کیف پولمون پذاریم حداقل  
اینچه پذاریم



## چگونه کار می‌کند؟ Bash

شما به عنوان لینوکس کار که بیشتر وقت خود را پشت این سیستم‌عامل می‌گذرانید بهتر است تا حدودی به خوبی درک کنید که پشت پرده چه اتفاقاتی می‌افتد.

وقتی یک command line یا اصطلاحاً دستوری را در ترمینال (terminal) تایپ می‌کنید، بعد از زدن enter یا return آنرا اجرا (run) می‌کند و خروجی را به شما نشان می‌دهد.

Bash تنها نخستین کلمه در command line را به عنوان کامند یا دستور در نظر می‌گیرد. منظور از نخستین کلمه، یعنی تا وقتی که شما کلید space را فشار دهید. باقی command یا دستور را به عنوان آرگمان در نظر می‌گیرد.



وقتی یک دستور را وارد کردید و bash اجرا کرد، در پایان کار خود دوباره این خط prompt را به شما نشان می‌دهد. اگر نشان نداد یعنی هنوز مشغول اجرای کامند قبلی است.

برای شروع و آشنایی با shell برحی از این کامندها را بررسی می‌کنیم.

## بررسی لاغین خود

وقتی به یک سیستم لینوکس login می‌کنید، لینوکس شما را به عنوان یک موجودیت خاص نمایش می‌دهد یا به عبارتی دیگر به شما یک identity می‌گیرد. این identity شامل نام کاربری(user id)، نام گروه شما (group id) می‌باشد. برای نمایش خودتان دستور id را تایپ کنید.

```
sahar@dance-with-linux:~$ id
uid=1000(sahar) gid=1000(sahar) groups=1000(sahar),4(adm),24(cdrom),27(sudo),30(dip),46(plugdev),118(lpadmin),128(sambashare)
```

این خطوط نشان می‌دهد که نام کاربر sahar بوده که به وسیلهٔ عددی کاربر 1000 (uid) نشان داده است. در اینجا گروه اصلی که sahar به آن تعلق دارد در groups نوشته شده. همچنین sahar به گروههای دیگری مثل 4(gid) adm و بقیه گروهها که آمده، تعلق دارد. این اسمی و اعداد نشان‌دهندهٔ permission یا اجازه‌هایی است که sahar برای دسترسی به منابع کامپیوتر دارد.

## فهمیدن shell دیفالت

قبل اشاره کرده بودیم که shell دیفالت لینوکس‌ها با هم تفاوت دارد. برای اینکه بدانید shell دیفالت یوزر شما چیست از این دستور استفاده کنید:

```
sahar@dance-with-linux:~$ who
sahar    tty2      2018-05-09 19:28 (/dev/tty2)
sahar@dance-with-linux:~$ grep sahar /etc/passwd
sahar:x:1000:1000:sahar,,,:/home/sahar:/bin/bash
sahar@dance-with-linux:~$
```



دستور who لاین فعالی شما را نمایش می‌دهد و دستور grep تعریف اکانت شما را در فایل /etc/passwd نشان می‌دهد. در اینجا بجای sahar باید user خودتان را جایگزین کنید و در ادامه /etc/passwd را تایپ کنید. در فیلد آخر، /bin/bash، نشان می‌دهد که shell دیفالت شما bash می‌باشد.

(همان طور که می‌بینید، بعد از اجرای کامند، دوباره خط اعلان يا prompt به شکل چشمک زنان نشان داده شده است. یعنی کامند قبلی اجرا شده و حالا منتظر گرفتن دستور جدید است.)



په منم چشمک زد

با این حال، این امکان وجود دارد که با shell‌های دیگری نیز کار کنید) (با فرض اینکه نصب شده‌اند یا اگر نصب نشده‌اند می‌توانید نصب کنید). Shell‌های زیادی وجود دارد مثل dash ، sh.csh ، tcsh ، ksh و ...

برای کار با این shell‌ها کافی است اسم آن‌ها را وارد کنید و وقتی هم که کارتان با آن shell تمام شد، می‌توانید با تایپ دستور exit برگردید به shell دیفالت. (در اینجا bash دیفالت است).

وقتی بفهمید یک shell چگونه کار می‌کند، به راحتی کار با shell‌های دیگر را یاد می‌گیرید. یا می‌توانید از صفحه راهنمای آن shell استفاده کنید برای مثال با تایپ man bash لیستی از کامندها و مشخصات آن shell مورد نظر را می‌توانید پیدا کنید.

## بررسی دایرکتوری‌ها

برای هر shell مکانی در سیستم فایل لینوکس وجود دارد که به current directory یا working directory یا دایرکتوری کنونی شناخته می‌شود مثلاً وقتی برای اولین بار به داخل لینوکس لاین می‌کنید، کار با دایرکتوری home را آغاز می‌کنید.





این دایرکتوری په چه دردی می‌مُحوره؟

وقتی درخواست بازگردان یا ذخیرهسازی یک فایل را می‌کنید، shell شما از دایرکتوری کنونی یا همان current directory به عنوان نقطهٔ رجوع استفاده می‌کند. مثلاً برای save کردن یک فایل، کافی است نام فایل را بدهید و آن فایل در دایرکتوری کنونی شما ذخیره خواهد شد.

برای اینکه بفهمید دایرکتوری کنونی شما چیست، pwd را تایپ کنید:

```
sahar@dance-with-linux:~$ pwd
/home/sahar
```

دستور pwd دایرکتوری کنونی شما را نشان می‌دهد. دایرکتوری فعلی در اینجا /home/sahar می‌باشد. همچنین اگر دایرکتوری کنونی را تغییر دادید می‌توانید با دستور cd که مخفف change directory است به دایرکتوری home باز گردید. اگر به دنبال این دستور، نام یک دایرکتوری آورده شود، دایرکتوری کنونی شما به آن دایرکتوری انتخابی تغییر می‌کند ولی دستوری cd بدون هیچ دنباله‌ای شما را به دایرکتوری home می‌برد.

دستور ls لیست محتویات موجود در دایرکتوری فعلی شما را نشان می‌دهد.

```
sahar@dance-with-linux:~$ ls
Desktop   Downloads   Music   Public   Videos
Documents examples.desktop Pictures Templates
sahar@dance-with-linux:~$
```

## چجوری از یک shell خارج شویم؟

بعد از اینکه کارتان با shell به پایان رسید، با تایپ دستور exit یا ctrl+d می‌توانید خارج شوید. این‌ها برای دست‌گرمی بود و گرنۀ صدھا دستور دیگر در دایرکتوری‌هایی مثل bin یا bin/usr وجود دارد. همچنین دستوراتی برای مدیریت سیستم در داخل دایرکتوری‌های /sbin و یا /usr/sbin قرار دارند. در ادامه کتاب بیشتر در مورد دستورات صحبت می‌کنیم.



## نحوه نوشتن کامندها یا command syntax

برخی از command‌ها، هیچ آرگمان و یا آپشنی در دنباله خودشان ندارند. مثل: ls یا pwd

می‌توانیم یکسری آیتم‌ها به این دستورها اضافه کنیم. با این کار چگونگی کارکرد آن دستور تغییر می‌کند. این آیتم‌ها عبارت است از:



یه حسی پوهم می‌گه پا این آیتم‌ها، کامندهامون شاسی پلند  
می‌شه یا یه چیزی مثل اسفناچی که ملوان زیل می‌خورد البته  
امروزه علم اثبات کرده اون اسفناچ نبوده که اون می‌خورده. هر  
چیه همون چیزی بوده که این خبره هم می‌خوره

- 1- آپشن‌ها (options)
- 2- آرگمان‌ها (arguments)
- 3- متغیرهای محیطی (enviroment variables)
- 4- متاکاراکترها (metacharacters)

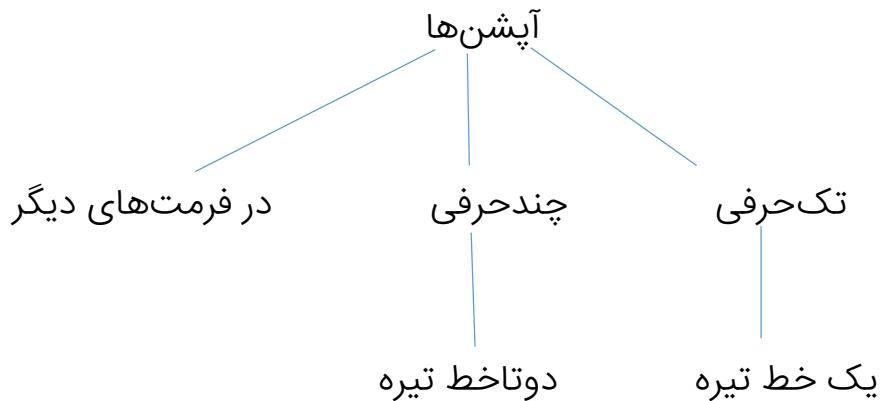
## آپشن‌ها (options)

با آپشن‌ها می‌توانیم رفتار یک کامند را تغییر دهیم. به کامند دقیقاً مشخص کنیم که چه می‌خواهیم، چه نمی‌خواهیم، چه کاری بکند، چه کاری نکند اکثر کامندها دارای یک یا چندین آپشن هستند.

آپشن‌های یک کامند را برنامه‌نویسی که آن کامند را نوشته، مشخص می‌کند.

همه کامندها آپشن دارند منتهی در فرمتهای مختلف. مثلاً برخی از برنامه‌ها، آپشن‌ش یک خط تیره یا dash یا - دارد، برخی دیگر دوتا خط تیره دارد و یا اصلاً برخی ندارند.





ولی در کل، آپشن‌ها یک لباسی دارند (یک الگویی دارند) که **اکثر** کامندها آن‌ها را می‌پوشند.  
در ادامه به شرح این لباس می‌پردازیم.

آپشن‌های تک حرفی معمولاً با یک خط تیره یا dash شروع می‌شوند.

برای مثال پیش از این اشاره کردیم که دستور ls محتویات دایرکتوری فعلی را نشان می‌دهد، اگر به دنبال این دستور آپشن - (خط تیره ای) تایپ کنیم، لیست کاملی از اطلاعات با جزئیات را ارائه می‌کند و همچنین تایپ آپشن a - فایل‌های مخفی (فایل‌های نقطه) را نشان می‌دهد.



با این کتاب نوشتن. فکر کنم تا حالا هیشکی از تجهیزه  
نگرفته. آخه خط تیره ال ؟؟؟ تمام پاورهای ریختی بهم

```
sahar@dance-with-linux:~$ ls -l -a
total 1528
drwxr-xr-x 16 sahar sahar 4096 Jun  5 07:40 .
drwxr-xr-x  3 root  root  4096 May  9 12:14 ..
-rw-r--r--  1 sahar sahar 1454080 May 11 04:46 backup.tar
-rw-----  1 sahar sahar    748 May 29 02:09 .bash_history
-rw-r--r--  1 sahar sahar   220 May  9 12:14 .bash_logout
-rw-r--r--  1 sahar sahar  3771 May  9 12:14 .bashrc
drwx----- 14 sahar sahar 4096 May 29 00:03 .cache
drwxr-xr-x 16 sahar sahar 4096 May 29 01:05 .config
drwxr-xr-x  2 sahar sahar 4096 May 11 05:45 Desktop
drwxr-xr-x  2 sahar sahar 4096 May  9 12:42 Documents
drwxr-xr-x  2 sahar sahar 4096 May  9 12:42 Downloads
-rw-r--r--  1 sahar sahar 8980 May  9 12:14 examples.desktop
drwx-----  2 sahar sahar 4096 May 29 00:03 .gnupg
-rw-----  1 sahar sahar 3496 Jun  5 07:39 .ICEauthority
drwxr-xr-x  3 sahar sahar 4096 May  9 12:42 .local
drwx-----  5 sahar sahar 4096 May 29 00:01 .mozilla
```

### توضیح درباره شکل:

دایرکتوری‌هایی که یک نقطه(.) دارند نشان‌دهنده دایرکتوری /home/sahar هستند یا همان /home/user/ یعنی دایرکتوری فعلی است. و آن‌هایی که دو نقطه(..) دارند، اصطلاحاً دایرکتوری والد گفته می‌شود. این دایرکتوری والد، همان /home/ است که تنها در اختیار کاربر root می‌باشد. تمامی فایل‌های دیگر در اختیار sahar نیز هست.

(اسامی فایل‌ها یا دایرکتوری‌هایی که در سمت راست آن‌ها یک نقطه(.) قرار دارد، فایل‌هایی را نشان می‌دهند که برای ذخیره‌سازی خاصیت‌های الG یا خاصیت‌های shell به کار برده می‌شوند).

در کل، فایل‌هایی که hidden (محفوی) هستند با یک نقطه شروع می‌شوند. آوردن آپشن -a- می‌گوید فایل‌های hidden را هم نشان بده.



پیچیه اطلاعات لیست مریپوت په انداره هر فایل پر حسب پایه، تاریخ و زمان آخرین پاری که دستتون په فایل مورده رو نشون می‌ده، پاتریش نکشید اینارم په



- (دشال) در کامند ls معنای long list را می‌دهد.

ممکن است در برنامه‌ای دیگر همین - معنی دیگری، مثلا login بدهد و کلا کار دیگری انجام دهد. یعنی آپشن‌ها بسته به چیزهایی که برنامه‌نویس نوشته است، می‌توانند متفاوت باشند و کاربردهای مختلفی داشته باشند.

اونجا که گفتی "پرندۀ‌ای دیگر" فکر کردم می‌خوای  
مارو په خدای پنرگ پسپاری



خپ می‌خوای من پسپارم؟

تا پرندۀ‌ای دیگر شما رو په خدای پنرگ می‌سپارم



می‌توانید چندین آپشن تک حرفی را در کنار یکدیگر بعد از یک خط تیره تایپ کنید. برای

-la :

می‌خواود پکه می‌شه آپشن‌های تک حرفی رو توی یه مخلوط کن  
پرینزید و قاطیش کنید و کنار هم بیارید مثل ls -la



یعنی این سه تا هیچ فرقی با هم ندارند:

ls -l -a

ls -a -l

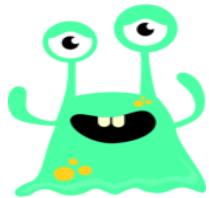
ls -la



```
sahar@dance-with-linux:~$ ls -la
total 84
drwxr-xr-x 13 sahar sahar 4096 May  9 12:45 .
drwxr-xr-x  3 root  root  4096 May  9 12:14 ..
-rw-----  1 sahar sahar   14 May  9 12:45 .bash_history
-rw-r--r--  1 sahar sahar  220 May  9 12:14 .bash_logout
-rw-r--r--  1 sahar sahar 3771 May  9 12:14 .bashrc
drwx----- 9 sahar sahar 4096 May  9 12:43 .cache
drwxr-xr-x 11 sahar sahar 4096 May  9 12:43 .config
drwxr-xr-x  2 sahar sahar 4096 May  9 12:42 Desktop
drwxr-xr-x  2 sahar sahar 4096 May  9 12:42 Documents
drwxr-xr-x  2 sahar sahar 4096 May  9 12:42 Downloads
-rw-r--r--  1 sahar sahar 8980 May  9 12:14 examples.desktop
-rw-----  1 sahar sahar   310 May  9 12:42 .ICEauthority
drwxr-xr-x  3 sahar sahar 4096 May  9 12:42 .local
drwxr-xr-x  2 sahar sahar 4096 May  9 12:42 Music
drwxr-xr-x  2 sahar sahar 4096 May  9 12:42 Pictures
-rw-r--r--  1 sahar sahar   675 May  9 12:14 .profile
drwxr-xr-x  2 sahar sahar 4096 May  9 12:42 Public
drwxr-xr-x  2 sahar sahar 4096 May  9 12:42 Templates
drwxr-xr-x  2 sahar sahar 4096 May  9 12:42 Videos
```

اما تعداد آپشن‌های تک حرفی کم است، پس اگر کامندهای نیاز به آپشن بیشتر داشت چه کنیم؟

در آن صورت، برنامه‌نویس یکسری آپشن‌هایی در نظر می‌گیرد که به آن آپشن‌های long گفته می‌شود. این آپشن‌های لانگ، دو تا خط تیره دارند. چرا؟ چون اگر یک خط تیره باشد، هر حرف آن آپشن را، یک آپشن مجزا در نظر می‌گیرد. ما با دو تا خط تیره گذاشتن به او می‌فهمانیم که چیزی که نوشتیم همگی باهم معرف یک آپشن است. مثل `ls --help`. اگر دو تا خط تیره تایپ نشود، `help` به صورت جداگانه `h`, `e`, `a`, `p`, حساب می‌شود.



خدا لگدش کنه که همه رو ټیکه ټیکه  
می کنه

مثلًا فرض کنید نمی‌دانیم دستور `ls` چه کاری انجام می‌دهد. می‌توانیم از آپشن `help` کمک بگیریم:



```
sahar@dance-with-linux:~$ ls --help
Usage: ls [OPTION]... [FILE]...
List information about the FILEs (the current directory by default).
Sort entries alphabetically if none of -cftuvSUX nor --sort is specified.

Mandatory arguments to long options are mandatory for short options too.
-a, --all                  do not ignore entries starting with .
-A, --almost-all           do not list implied . and ..
--author                   with -l, print the author of each file
-b, --escape                print C-style escapes for nongraphic characters
--block-size=SIZE           scale sizes by SIZE before printing them; e.g.,
                           '--block-size=M' prints sizes in units of
```

در اینجا آپشن‌هایی که کامند ls می‌تواند بپذیرد را نشان داده است.

با تایپ less خروجی به طور صفحه به صفحه نشان داده خواهد شد. برای خروج می‌توانید q را فشار دهید:

```
sahar@dance-with-linux:~$ ls --help | less
Usage: ls [OPTION]... [FILE]...
List information about the FILEs (the current directory by default).
Sort entries alphabetically if none of -cftuvSUX nor --sort is specified.

Mandatory arguments to long options are mandatory for short options too.
-a, --all                  do not ignore entries starting with .
-A, --almost-all           do not list implied . and ..
--author                   with -l, print the author of each file
-b, --escape                print C-style escapes for nongraphic characters
--block-size=SIZE           scale sizes by SIZE before printing them; e.g.,
                           '--block-size=M' prints sizes in units of
                           1,048,576 bytes; see SIZE format below
-B, --ignore-backups       do not list implied entries ending with ~
-c                         with -lt: sort by, and show, ctime (time of last
                           modification of file status information);
                           with -l: show ctime and sort by name;
                           otherwise: sort by ctime, newest first
-C                         list entries by columns
--color[=WHEN]              colorize the output; WHEN can be 'always' (default
                           if omitted), 'auto', or 'never'; more info below
-d, --directory            list directories themselves, not their contents
-D, --dired                 generate output designed for Emacs' dired mode
-f                         do not sort, enable -aU, disable -ls --color
:|
```



## آپشن‌هایی که مقدار می‌پذیرند

برخی از آپشن‌ها نیز وجود دارند که به وسیله آن‌ها ما "مقداری" یا یک "value" ای به پراسس پاس می‌دهیم.

مثالاً کامند `head` با آرگمان `n`- با آپشن `/etc/samba/smb.conf` - چند خط اول فایل `smb.conf` را نشان می‌دهد. برای این که بگوییم چند خط را نشان دهد، عدد یا آن `value` را در کنار `n` می‌نویسیم.

مثالاً من یک فایل تکستی در دایرکتوری `/HOME/SAHAR` ساختم به اسم `EXAMPLE.TXT` و داخل آن متنی گذاشتم. می‌خواهم با استفاده از کامند `HEAD` تنها دو خط اول این فایل را بخوانم:

```
sahar@dance-with-linux:~$ head /home/sahar/example.txt -n2
Dance with linux
learn gnu/linux
sahar@dance-with-linux:~$
```

اگر بخواهم شش خط اول را بخوانم:

```
sahar@dance-with-linux:~$ head /home/sahar/example.txt -n6
Dance with linux
learn gnu/linux
the fun way
sahar shaker
web:
saharshaker.com
sahar@dance-with-linux:~$
```

شیش چشنه که می‌گید شش؟



یعنی با آوردن آپشن `n`- و بعد مقدار عددی 6 مشخص کردم که شش خط اول را نشان بدھ.



در برخی آپشن‌ها این value به آپشن (اینجا  $n$ ) می‌چسبد ولی در برخی هم یک اسپیس بین آن‌ها وجود دارد. در برخی موقع هم باید یک مساوی قرار بگیرد:  $n=6$ -

ولی این وسط داریم کامندهایی را که سرکشی کردند یا به عبارتی حال کرده‌اند طوری دیگری آپشن داشته باشند. مثلًا کامند ip آپشنی ندارد که با dash بتوان با او رفتار کرد. یعنی آپشن‌های ip هیچکدام خط تیره یا dash ندارد.

اگر بخواهیم ip یک interface را ببینیم،

```
sahar@dance-with-linux:~$ ip addr show dev lo
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default
    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
            valid_lft forever preferred_lft forever
sahar@dance-with-linux:~$
```

توی اینجا بعد از ip هر چه آمده، آپشن است اما هیچکدام خط تیره ندارد!



همه این‌ها رو پاپید حفظ کنیم؟ ۰\_۰

نه. بلکه بیشتر باید به آن کامند بروید تا متوجه شوید که چه آپشن‌هایی و به چه شکلی وجود دارد؛ رفته رفته با تکرار یادتان می‌ماند، مثل اولین باری که مشتق می‌گرفتید فرمول حفظ می‌کردید ولی بعدها ذهنی انجام می‌دهید.



معچتو گرفتم، مشتق کیه؟

منتھی اکثر کامندها، آپشن‌هایشان همین لباس یدونه dash یا دوتا dash را پوشیده‌اند.



## پیدا کردن اطلاعات در مورد کامندها

دستوراتی دیگری که اطلاعات یا information در مورد کامندها در اختیار شما می‌گذارند، یکی man و دیگری info است. اما کامند help خلاصه‌ای از اطلاعات دستورات را نشان می‌دهد. پیش از این از آپشن help هم استفاده کردیم.

وقتی اولین بار از shell استفاده می‌کنید، شاید فکر کنید هیولاست! شما فقط یک خط می‌بینید که منتظر است تا دستورات شما را اجرا کند.



خیلی هم دلش بخواه هیولا باشه

از کجا بفهمیم اصلاح کامندهایی هست؟ چه آپشن‌هایی می‌گیرن؟ یا چچوری پیش‌رفته تر ازشون استفاده کنیم و اقرودنی‌های معجازشون چیه؟



مکه کلاس آشپزی با مانع است؟ بقیه قصه برای  
چوای سوالنه



## 1- چک کردن PATH

در shell تایپ کنید:

```
echo $PATH
```

نُشْد



پُزْرَگْ ٹَیپُوْ کُنْ PATH ڦپلشِم یه علامت دلار پُذْدار.

Every body listen and repeat PATH

یک لیست از دایرکتوری‌های یا اتاق‌هایی که کامندها در آن حضور دارند به شما نشان داده خواهد شد. محتویات داخل این دایرکتوری‌ها اکثراً دستورات استاندارد لینوکس را نشان می‌دهد، یعنی داخل این اتاق‌ها، دستورات حضور دارند. مثلاً می‌خواهیم بفهمیم در دایرکتوری یا اتاق bin چه دستوراتی زندگی می‌کنند:



```
sahar@dance-with-linux:~$ echo $PATH
/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/games:/usr/local/games:/snap/bin
sahar@dance-with-linux:~$ ls /bin
bash          fusermount    ntfs-3g        su
bunzip2       getfacl      ntfs-3g.probe  sync
busybox       grep         ntfscat       systemctl
bzcat         gunzip      nftscluster   systemd
bzcmp         gexe        ntfscmp      systemd-ask-password
bzdiff        gzip        ntfsallocate  systemd-escape
bzegrep       hciconfig   ntfsfix       systemd-hwdb
bzexe         hostname    ntfsinfo     systemd-inhibit
bzfgrep       ip          ntfsls       systemd-machine-id-setup
bzgrep        journalctl ntftsmove    systemd-notify
bzip2         kbd_mode    ntfsrecover   systemd-sysusers
bzip2recover kill        ntfsseaudit  systemd-tmpfiles
bzless        kmmod       ntfstruncate  systemd-tty-ask-password-agent
bzmore        less        ntfsusermap  tar
cat           lessecho    ntfswipe     tempfile
chacl         lessfile    open         touch
chgrp         lesskey     openvt       true
chmod         lesspipe    pidof       udevadm
chown         ln          ping        unlockmgr_server
chvt          loadkeys   ping4      umount
```

## 2- استفاده از کامند help

بعضی از کامندها هم در دایرکتوری‌ها یا همان اتاق‌ها دیده نمی‌شوند. در واقع این کامندها داخل خود shell از پیش ساخته شدند. برای دیدنش از کامند help می‌توانید استفاده کنید. منتهی خروجی این کامند چون خیلی زیاده، می‌توانید با نوشتן کامند زیر، به صورت صفحه به صفحه ببینید.

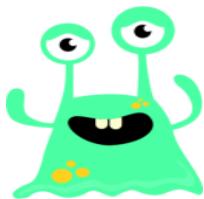
help | less



Use 'info bash' to find out more about the shell in general.  
Use 'man -k' or 'info' to find out more about commands not in this list.

A star (\*) next to a name means that the command is disabled.

```
job_spec [&]
(* expression )
. filename [arguments]
:
[ arg... ]
[[ expression ]]
alias [-p] [name[=value] ... ]
bg [job_spec ...]
bind [-lpsvPSVX] [-m keymap] [-f file>
break [n]
builtin [shell-builtin [arg ...]]
caller [expr]
case WORD in [PATTERN [| PATTERN]...)>
cd [-L|[-P [-e]] [-@]] [dir]
command [-pVv] command [arg ...]
compgen [-abcdefgjksuv] [-o option] [>
complete [-abcdefgjksuv] [-pr] [-DE] >
compopt [-o]+o option] [-DE] [name ..>
history [-c] [-d offset] [n] or hist>
if COMMANDS; then COMMANDS; [ elif C>
jobs [-lnprs] [jobspec ...] or jobs >
kill [-s sigspec | -n signum | -sigs>
let arg [arg ...]
local [option] name[=value] ...
logout [n]
mapfile [-d delim] [-n count] [-O or>
popd [-n] [+N | -N]
printf [-v var] format [arguments]
pushd [-n] [+N | -N | dir]
pwd [-LP]
read [-ers] [-a array] [-d delim] [->
readarray [-n count] [-O origin] [-s>
readonly [-aAf] [name[=value] ...] o>
return [n]
select NAME [in WORDS ... ;] do COMM>
set [-abefhkmnptuvxBCHP] [-o option->
```



قپول نیست چیزایی که ما نمی‌فهمیم رو پنویسی و ما فقط تایپ کنیم. اصلاً من بازی نمی‌کنم

قرار نیست اینچوری تمام شه، و گرنه غم انگیزه. همشو می‌گه ای خدا قپول کنه. تو فقط تحمل داشته پاش. اینا رو یاد پگیر که چایی موندی هی گیج نشی



یعنی چی هی می‌گه صفحه صفحه؟



یعنی موسٹو اسکرول کنی میاد پایین پاڑ، می توںی ۹ رو پذرنی ٹموم  
کنی این پازی کشیفو



این دستور فقط با shell یا پوستہ bash کار می کند اگر با دیگری کار می کنید، این دستور اجرا نخواهد شد.

### 3- می توانید help- را با دستوری که می خواهید به کار ببرید

خیلی از کامندهایی که وجود دارند، آپشنی به اسم help هم گذاشته اند تا کاربر گیج نشود.  
می توانید کامندي را که می خواهید بغلش help-- را استفاده کنید.

مثلث فرض کنید نمی دانیم ls چه آپشن هایی را قبول می کند:

```
sahar@dance-with-linux:~$ ls --help
Usage: ls [OPTION]... [FILE]...
List information about the FILES (the current directory by default).
Sort entries alphabetically if none of -cftuvSUX nor --sort is specified.

Mandatory arguments to long options are mandatory for short options too.
-a, --all                         do not ignore entries starting with .
-A, --almost-all                   do not list implied . and ..
--author                          with -l, print the author of each file
-b, --escape                        print C-style escapes for nongraphic characters
--block-size=SIZE                  scale sizes by SIZE before printing them; e.g.,
                                   '--block-size=M' prints sizes in units of
                                   1,048,576 bytes; see SIZE format below
-B, --ignore-backups              do not list implied entries ending with ~
-c                               with -lt: sort by, and show, ctime (time of last
                                   modification of file status information);
                                   with -l: show ctime and sort by name;
                                   otherwise: sort by ctime, newest first
-C                               list entries by columns
--color[=WHEN]                     colorize the output; WHEN can be 'always' (default
                                   if omitted), 'auto', or 'never'; more info below
-d, --directory                   list directories themselves, not their contents
-D, --dired                        generate output designed for Emacs' dired mode
-f                               do not sort, enable -aL, disable -ls, --color
```



من این دارو توی خواپیم می پینم. چن دا کامند دیگه ای بله  
نیستی؟ یکم مخشنالموں کن دیگه چیه همش تکراره تازه  
بالا هم اینو مثل زدی.



تو دیگه از تنوع حرف نهن. توی این هزار سالی که از خدا عمر  
گرفتم فقط قیافت همین شکلی بود. ما هر شونخی ای کردیم درین از  
یه انجنا روی صورتت



هر ))))))(( ارسالته؟؟؟ مُجالت پکش از هیکلت.

پایام خدای ام ر عادت داشت ما هر خدیتی که می کردیم می گفت  
چهل ساله از هیکلت مُجالت پکش. اگر تو رو می دید که هزاره ای،  
خیلی خبر کیف می شد.



پایام مُرد و تو رو پرای ما به ارث گذاشت، همه خدیت ها نسل به نسل  
چرخیده و انباسنه شده توی تو. تو هم خداب شدی روی سر ما. یه ذره از  
خدیت تو پا مُودش به گور نبرد یا ارث می ڈاشت پرای بقیه.

مثلا با دستور `mkdir` می توان یک دایرکتوری ساخت، برای اینکه اطلاعات بیشتری از او  
داشته باشیم:

این دستور آمار همه رو می ده؟ پنچن پینیم مخونه  
لینوکس کجاست؟



```
sahar@dance-with-linux:~$ mkdir --help
Usage: mkdir [OPTION]... DIRECTORY...
Create the DIRECTORY(ies), if they do not already exist.

Mandatory arguments to long options are mandatory for short options too.
-m, --mode=MODE    set file mode (as in chmod), not a=rwx - umask
-p, --parents      no error if existing, make parent directories as needed
-v, --verbose       print a message for each created directory
-Z                 set SELinux security context of each created directory
                  to the default type
--context[=CTX]   like -Z, or if CTX is specified then set the SELinux
                  or SMACK security context to CTX
--help            display this help and exit
--version         output version information and exit

GNU coreutils online help: <http://www.gnu.org/software/coreutils/>
Full documentation at: <http://www.gnu.org/software/coreutils/mkdir>
or available locally via: info '(coreutils) mkdir invocation'
sahar@dance-with-linux:~$ █
```

توی واقعیت پرای مخونه ساختن پاس عین این خد پول چمع  
کنی، صد نقدرو راضی کنی پیان پسازن بعد (ینجا) پا یه  
په اتاق می زنی زندگیتو می کنی



مگه شما خرها هم مخونه می سازین؟



آره مگه مثل شما آواره سحر نامیم؟ ما مخونه زندگی  
می سازیم روی پای خودمون و امیسیم. بیدون از (ینجا)  
واقعیت محظوظ



کم موئده پود تو یکی پهمنون درس احلاق پدی



#### man - دستور 4

دستورات قبلی، خلاصه‌ای از اطلاعات را در اختیار شما می‌گذارند. برای اطلاعات بیشتر از دستور man استفاده کنید.

نکنین. خانواده نشسته. نشون می‌ده لباس چی پوشیده.



می‌خواستم پنهان برای اطلاعات پیشتر به روزنامه‌های کثیرالانتشار  
مرا جمعه کنید. این همیش می‌خواه پیچونه. حالا پیش کی گفتم

یعنی چی هی کلمه man کهار کامند مردم پیاده ملته  
ناموس دارن. چه وضعش؟



تایپ کردم این man رو.  
می‌گ  
what manal page do you want?  
وقتنه پدم پیلمو پیارم؟



پاید کامندی که می‌خوای رو هم پُغش پیاری.



برای مثال بزنید man mkdir

دوپاره قفل کرد روی یه دستور جدید. پاپا ٹنونونوونوونو



```
Mkdir(1)                         User Commands                     Mkdir(1)

NAME
    mkdir - make directories

SYNOPSIS
    mkdir [OPTION]... DIRECTORY...

DESCRIPTION
    Create the DIRECTORY(ies), if they do not already exist.

    Mandatory arguments to long options are mandatory for short options
    too.

    -m, --mode=MODE
        set file mode (as in chmod), not a=rwx - umask

    -p, --parents
        no error if existing, make parent directories as needed

    -v, --verbose
        print a message for each created directory

Manual page mkdir(1) line 1 (press h for help or q to quit)
```



## 5- استفاده از info

Info هم یکی از دستورات دیگری است که به دادن اطلاعات یا information به شما کمک می‌کند. منتهی تمام کامندها در دیتابیس info ندارند. اما اگر داشته باشند، اطلاعات بیشتری نسبت به man می‌دهند.



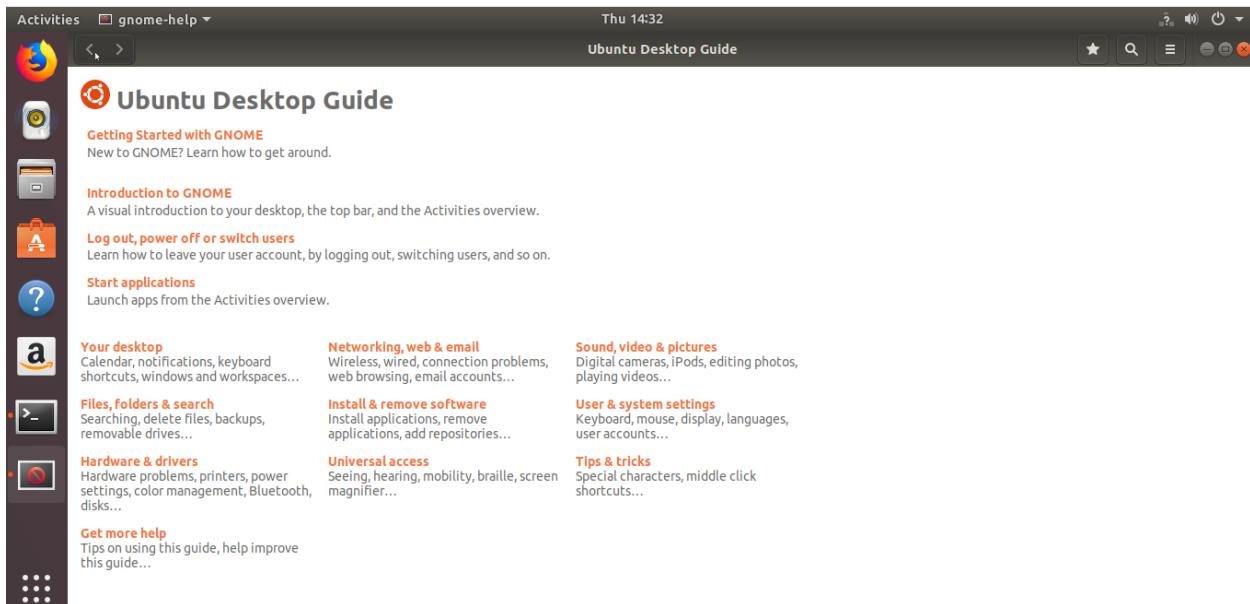
مثلاً اینکه دیدور هم چی پوشیدن

تو نمی‌خواهی پس کنی؟



## 6- استفاده از دستور gnome-help

مواردی که پیش از این گفتیم، در shell اجرا می‌شد اما با تایپ دستور gnome-help در محیط گرافیکی باز می‌شود که از آن هم می‌توانید استفاده کنید. اپلیکشن help در محیط گرافیکی باز می‌شود که از آن هم می‌توانید استفاده کنید.



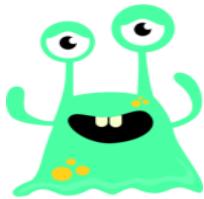
## آرگمان‌ها (arguments)

خیلی از کامندها می‌توانند جدا از آپشن‌ها، آرگمان نیز بپذیرند.



ملت چه توانی یی هایی دارن. نمی‌شه منم به فرزند  
مُوندگی قبول کنن؟

یک آرگمان، یک بخش اضافی از اطلاعات به شمار می‌رود.



يعنى كلا اضافي ان؟

اطلاعاتی مثل اسم یک فایل، دایرکتوری، username و آیتم‌های دیگری که از کامند درخواست می‌کند آن‌ها را نشان دهد.

عين پليسا رفتار مي کنه. چاسوسه؟



مثلا دستور cat /etc/passwd محتويات فایل /etc/passwd را نشان می‌دهد. در اينجا آرگمان به شمار می‌رود /etc/passwd.



```
sahar@dance-with-linux:~$ cat /etc/passwd
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
proxy:x:13:13:proxy:/bin:/usr/sbin/nologin
www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
backup:x:34:34:backup:/var/backups:/usr/sbin/nologin
list:x:38:38:Mailing List Manager:/var/list:/usr/sbin/nologin
irc:x:39:39:ircd:/var/run/ircd:/usr/sbin/nologin
gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/usr/sbin/nologin
```

## جایی که آرگمان‌ها با آپشن‌ها قاطی می‌شوند

آپشن‌ها در واقعی نوعی آرگمان‌اند.

گاهی اوقات یک آرگمان با یک آپشن شریک می‌شود (!) در این حالت، آرگمان فوراً بعد از آپشن قرار می‌گیرد.

وقتی می‌خواهیم به دنبال آپشن‌های تک حرفی، آرگمانی بیاوریم، بعد از یک space می‌آید. اما اگر آپشن مد نظر، تک حرفی نبود، برای اینکه آرگمان را به دنبال آن بیاوریم، از علامت تساوی = استفاده می‌کنیم.

مثال: دستور

```
sahar@dance-with-linux:~$ ls --hide=Desktop
Documents examples.desktop Pictures Templates
Downloads Music Public Videos
sahar@dance-with-linux:~$
```

--hide آپشنی است که روی کامند ls نشسته، از طرفی می‌دانیم که کامند ls کارش لیست کردن محتویات فایل یا دایرکتوری‌ای است که به آن می‌سپارند، --hide به کامند ls می‌گوید که موقع لیست کردن محتویات، دایرکتوری‌ها و یا فایل‌هایی که اسم آن‌ها Desktop است را نمایش نده.



چه خبیث. مگه اون ٹو چخبره؟



فقط نکته اینجاست که بین دو طرف تساوی هیچ space یا فاصله‌ای نیست. یعنی اول آپشن اومنده بعد هیچ space‌ای نیست و تساوی اومنده، دوباره هیچ space‌ای نیست و آرگمان آمده.

کشتنی خود تو فهمیدیم اسپیس نداره دیگه



مثال زیر، یک آپشن تک‌حرفی است که به دنبال آن یک آرگمان آمده است:

```
$ tar -cvf backup.tar /home/sahar
```

```
sahar@dance-with-linux:~$ tar -cvf backup.tar /home/sahar
tar: Removing leading '/' from member names
/home/sahar/
/home/sahar/Downloads/
/home/sahar/Documents/
/home/sahar/.config/
/home/sahar/.config/dconf/
/home/sahar/.config/dconf/user
/home/sahar/.config/yelp/
/home/sahar/.config/pulse/
/home/sahar/.config/pulse/45babca5877946558460739b7500ae5e-stream-volumes.tdb
/home/sahar/.config/pulse/45babca5877946558460739b7500ae5e-device-volumes.tdb
/home/sahar/.config/pulse/45babca5877946558460739b7500ae5e-default-source
/home/sahar/.config/pulse/45babca5877946558460739b7500ae5e-card-database.tdb
/home/sahar/.config/pulse/45babca5877946558460739b7500ae5e-default-sink
/home/sahar/.config/pulse/cookie
```

توی اینجا آپشن cvf - می‌گوید یک فایل به اسم backup با پسوند tar بساز و همه محتويات داخل دایرکتوری /home/sahar را داخل آن بگذار.



(ونچا) که می گوییم پساز منظورش از create یا مخففش آپشن. اونچایی هم که می گوییم "فایل" منظورش file که توی آپشن مخففش f اومده. اون حرف l هم که وسط اومده، منظورش (به معنی طولانی، دراز) که یه پیام متندی میده که پنهان بک آپی که میخواستی ساختم. حالا اگر پرید توی دایرکتوری /home/user یوشه هر را میبینید.



## متاکاراکترها

یکسری کاراکترهای خاص وجود دارند که نمیتوانید آنها را در کامندها استفاده کنید. به اینها متاکاراکتر گفته می‌شود.

هر کدام از این متاکاراکترها برای bash معنی خاصی دارد. انواع مختلفی هم دارند. یعنی از طریق این کاراکترها ما میتوانیم با bash راحت‌تر حرف بزنیم و چیزهای بیشتری ازش بخواهیم. در ادامه به معرفی بعضی از این متاکاراکترها می‌پردازیم.

## متاکاراکتر نقطه‌ویرگول

نقطه‌ویرگول یا semicolon یا ; کاراکتری است که دو دستور را از هم جدا می‌کند. اگر بنویسم:

date ; date ; date

سه تا خروجی جدا در bash می‌بینید.

```
sahar@dance-with-linux:~$ date;date;date
Tue Jun  5 09:46:46 MDT 2018
Tue Jun  5 09:46:46 MDT 2018
Tue Jun  5 09:46:46 MDT 2018
```





یعنی نقطه وید گول نمی‌ذاره بقیه په عنوان آرگمان پاس داده پشن،  
هر کدوم رو یه کامند جدا می‌دونه

چه فهمیده. کار ایشون رو دوست دارم. بیاد من و لینوکس رو  
linux ; khare ; هم په هم وصل کنه، مثلا پزنه



شتبه فهمیدی، کار نقطه وید گول چدا کردن دوئی کامند از  
همدیگرس نه وصل کردنش. یعنی می‌شه توی یه خط چندین کامند  
رو نوشته و پا نقطه وید گول چدا کرد.



لینوکس هم پرای من معنی خاصی می‌ده، یه چیزی  
بیشتر از یه سیستم عامل

منم همینطور



## متاکاراکتر space

Space یا فاصله، خودش یکی از متاکاراکترها به شمار می‌رود که کامند را از آرگمان جدا می‌کند. مثلا وقتی تایپ می‌کنیم la- la بین کامند la و آپشن la- فاصله گذاشتیم.



## متاکاراکتر امپرسند &

بعضی از کامندها برای اینکه کامل اجرا شوند، طول می‌کشد (ماهم نسل بی‌حوصله) برای اینکه منتظر آن نمانیم، با کاراکتر امپرسند می‌توانیم دستور بعدی را در بک‌گراند run کنیم. به عبارتی این متاکاراکتر به شما اجازه می‌دهد تا به کارتان ادامه دهید و منتظر تکمیل کامند نمانید.

امپرسند معمولاً در پایان یک برنامه می‌آید.

کامندهایی که مربوط به فرمت متن می‌باشند یا به اصطلاح text formatting کامندهایی مثل troff یا nroff که بعداً توضیح می‌دهیم، از این دست کامندهایی هستند که در background اجرا می‌شوند.

حتی ممکن است شما shell script هایی نوشته باشید که در background اجرا می‌شوند و یک اتفاقی که مدام در اتفاق افتادن است را چک می‌کند. مثلاً میزان پرشدن هارد دیسک شما یا کاربران خاصی که لاغین کردند.



شل اسکرپت چیه؟



پیکسری پرنامه های کوتاه که پرای یه هدفی مینویسی.

مثلًا فرض کنید سه تا shell script با پایتون نوشته‌اید، برای استفاده از امپرسند اینگونه تایپ می‌کنید:

`./script.py & ./script2.py & ./script3.py &`



توجه کنید که & در پایان هر کدام آمده است.

### متاکاراکتر دبل امپرسند &&

این کاراکتر جدا کننده دو تا کامند است.

Date && cal

```
sahar@dance-with-linux:~$ date && cal
Tue Jun  5 09:51:20 MDT 2018
      June 2018
Su Mo Tu We Th Fr Sa
          1  2
3  4  5  6  7  8  9
10 11 12 13 14 15 16
17 18 19 20 21 22 23
24 25 26 27 28 29 30
```



عادت داری حرف تکراری پنجمی؛ فرقش با  
نقطه و پرگول چیه؟

هر کامندی وقتی کارش تمام می‌شود، یک status code به bash می‌فرستد تا وضعیت حال خودش را گزارش دهد. یعنی بگه من کامندو به خوبی اجرا کردم یا قاطی کردم و مشکلی پیش‌آمد

مثلًا اگر صفر بده، یعنی برنامه رو اجرا کردم. عملیات با موفقیت انجام شد!

توی اینجا، وقتی bash متوجه شود که status code اولی، یعنی date صفر بوده، یعنی با موفقیت انجام شده، دومی یعنی cal را اجرا می‌کند.



اگر کامند اول حالت خوب نباشد و درست اجرا نشود، سراغ دومی هم نمی‌رود. به عبارتی با استفاده از دابل امپرسند، ما شرط می‌گذاریم که اگر اولین کامند اجرا نشد بقیه را هم اجرا نکن.

### متاکاراکتر پایپ، pipe با علامت |

متاکاراکتر پایپ یک خط عمودی است، معمولاً در کیبوردها با گرفتن شیف+بکسلش ظاهر می‌شود. این متاکاراکتر، خروجی حاصل از یک دستور را به عنوان ورودی دیگری ارسال می‌کند.



یعنی از اولی می‌گیرد و به دومی می‌دهد

با این‌کار فرصت این را داریم که دستور اول روی برخی داده‌ها کار کرده، و دستور دوم روی خروجی دستور اول کار کند.



نفهمیدم

فکر کن خروجی دستور اول یه چیزی می‌شه مثل X، دستور دوم روی این خروجی یعنی X کار می‌کنه. انگار یکی قبل از خروجی فقط برمی‌داره می‌خوره و کارهایی که بهش سپردن رو روی اون انجام می‌دهد.

الگویی شبیه به این دارد:

Command1 | command2

```
sahar@dance-with-linux:~$ ls /usr/bin | sort | less
sahar@dance-with-linux:~$
```



این دستور، محتویات دایرکتوری `/usr/bin` را لیست کرده، محتویات را بر اساس حروف الفبا (بدون در نظر گرفتن کوچکی و بزرگی حروف) مرتب می‌نماید و خروجی را به دستور `less` ارسال می‌کند. با این کار شما می‌توانید نتایج را به صورت صفحه به صفحه ببینید. برای خروج هم کلید `q` را فشار دهید.

یک مثال دیگر:

```
sahar@dance-with-linux:~$ cat /etc/passwd | sort | less
sahar@dance-with-linux:~$
```

این دستور محتویات فایل `/etc/passwd` را لیست می‌کند و خروجی آن را به دستور `sort` وصل می‌کند، دستور `sort` اسمی کاربرانی که در هر فایل `/etc/passwd` شروع می‌شوند را گرفته و آنها را بر حسب حروف الفبا مرتب می‌کند و در آخر خروجی آن را به `less` ارسال می‌کند. دستور `less` هم خروجی را به صورت صفحه به صفحه نمایش می‌دهد.

پایپ‌ها به خوبی این حقیقت را نشان می‌دهند که یونیکس همان جد لینوکس (!) چجوری از بلوک‌های مختلف ساخته شده‌اند.

## متاکاراکتر دابل پایپ‌لاین

`date || cal`

```
sahar@dance-with-linux:~$ date || cal
Tue Jun  5 09:55:25 MDT 2018
sahar@dance-with-linux:~$ date2 || cal
No command 'date2' found, did you mean:
  Command 'date' from package 'coreutils' (main)
date2: command not found
      June 2018
Su Mo Tu We Th Fr Sa
              1  2
 3  4  5  6  7  8  9
10 11 12 13 14 15 16
17 18 19 20 21 22 23
24 25 26 27 28 29 30
```



در این مตاكاراكتر اگر date درست اجرا شود یعنی status code آن صفر باشد، دومی دیگر اجرا نخواهد شد. به عبارتی اين متاكاراكتر، هر کدام که صحيح بود را اجرا می‌کند. "این یا آن"

اگر هر دو صحیح بود، اولی را اجرا می‌کند، در غیر این صورت سراغ دومی می‌رود و الى آخر.

به عبارتی این شرط را می‌گذاريم که هر کدام از کامندها که حالت خوب بود آنرا اجرا کن. حتی یکی! فقط انجامش بده.

همانطور که در تصویر می‌بینید، کامند date2 اجرا نمی‌شود پس سراغ کامند بعدی رفته و cal را نمایش داده است.

## متاكاراكتر > یا خروجي استاندارد برگردیم سر bash

وقتی یک کامنلاین تایپ می‌شود تا bash آنرا اجرا کند، برنامه شما را اجرا کرده یا به عبارتی execute می‌کند و بعد، آرگمان‌ها را به برنامه پاس می‌دهد. حالا برنامه ممکن است از آن‌ها استفاده بکند یا نکند. اینجا پراسس یا process ایجاد می‌شود. یک پراسس در طول عمر خود، هر وقت که تمایل داشت می‌تواند خروجي ارسال کند. مثل زمان که در برنامه پایتون با دستور print می‌گفتیم خروجي را ارسال کن و مثلًا روی صفحه یک سلامی به ما بده hello را نشان بده. مشابه همین، این خروجي به دست bash می‌رسد، bash به طور دیفالت این خروجي را روی صفحه نمایش، نشان می‌دهد، اما کاراكتری وجود دارد به اسم "بزرگتر از" یا > یا greater than به طور مخفف .gt.



پادم نمی‌مونه، من همیشه <پا> گاتی می‌کنم. از کجا پفهتم؟

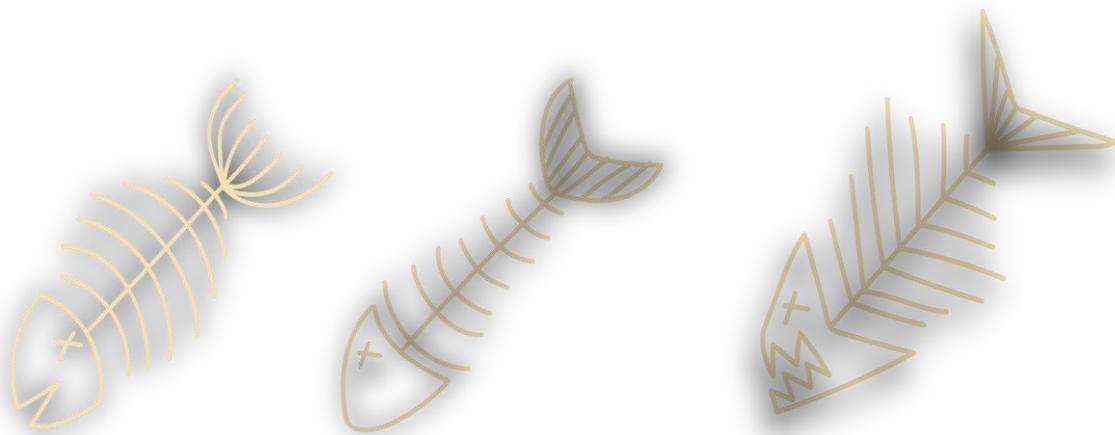


پیین دهن این ماهیه ب کدوم طرفه هر چا که غذا پیشتر پاشه ماهی  
به اون سمت دهنش بازه.



ماهیه چرد خید چی؟ اصن دهن ماهیه های من ^ این شکلین.





عاشق اوئیم کە دھنپش پستس. خدەچان سختش نکن دىگە. يە شكلو ياد پىگىر دىگە قاطى نكى.



اصلن منم مى خواام ياد پىگىرم.

منم بازى

نە تو دىگە. هەر چى چك و چونورە اينچاسەت. بە اندازە كافى از خدە مى كشىيم. تو يىكى وارد نشۇپا اون حافظە سە ئانىيەايت



اين علامت، يك متاكاراكتر به شمار مى رود و هر وقت bash اين كاراكتر را ببىند متوجه مى شود كە باید خروجى را در يك فاييل برىزد. چە فايلى؟ فايلى كە شما نام آن را بعد از علامت بزرگتر مى آوريد.

مثلا در ترمinal مى نویسم: cal

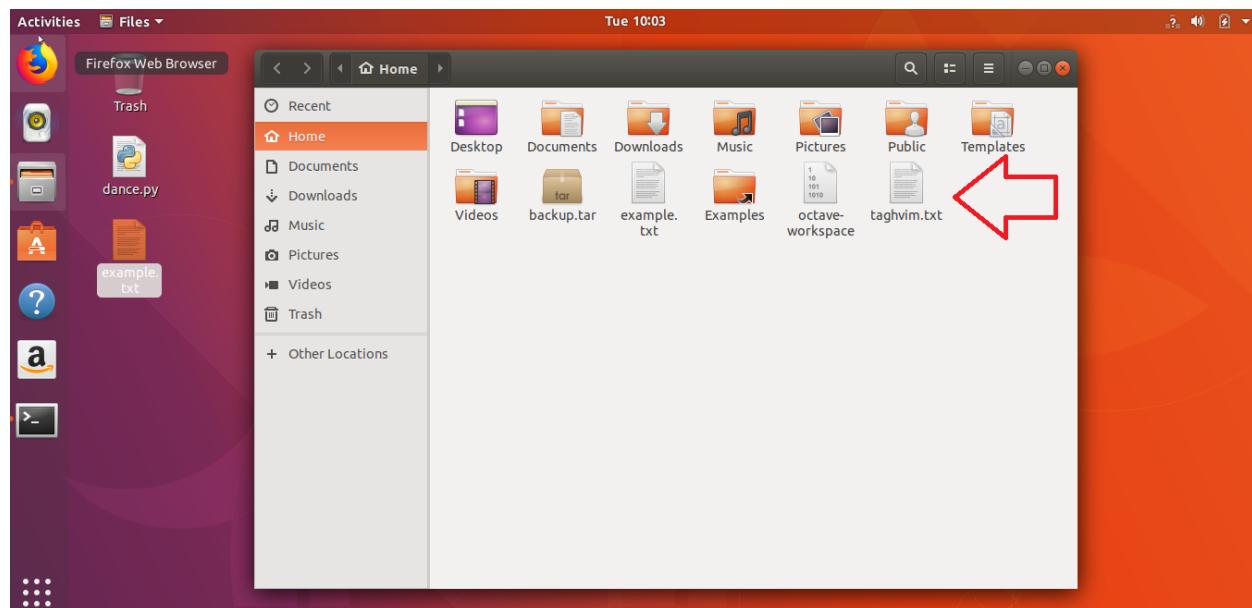
با اين كار تقويم را روی صفحه نمايش به من نشان مى دهد، اما اگر بنویسم:



cal > taghvim.txt

```
sahar@dance-with-linux:~$ cal > taghvim.txt
sahar@dance-with-linux:~$
```

دیگه خروجی‌ای روی صفحه نمی‌بینید، و خط prompt برای گرفتن دستور بعدی مدام چشمک می‌زند و این یعنی دستور قبلی اجرا شده. اما خروجی کجا رفته؟ خروجی درون فایل taghvim.txt درون دایرکتوری جاری حضور دارد. دایرکتوری جاری ما الان فolder هستیم) / است. (یادمون نره که با pwd هم می‌تونستیم ببینیم توی چه دایرکتوری



نکته مهم اینجاست هر بار که این علامت بزرگتر را استفاده می‌کنیم، آن فایل مورد نظر (اینجا taghvim.txt) خالی می‌شود، پاک می‌شود و خروجی دستور جدید در آن قرار می‌گیرد. اصطلاحاً به این کار می‌گوییم overwrite شدن.

اگر بخواهیم Overwrite نشود و خروجی به انتهای فایل اضافه شود، به جای یک بزرگتر، از دو تا علامت بزرگتر استفاده می‌کنیم.

فرض کنید می‌خواهیم date هم به آخر فایل تقویم اضافه شود:



date >> taghvim.txt

```
sahar@dance-with-linux:~$ cal > taghvim.txt
sahar@dance-with-linux:~$ date >> taghvim.txt
sahar@dance-with-linux:~$
```

```
Open + taghvim.txt ~/
Save ⌂ ⎓ ⎔ ⎕ ⎖ ⎗
June 2018
Su Mo Tu We Th Fr Sa
      1  2
3  4  _  5  6  7  8  9
10 11 12 13 14 15 16
17 18 19 20 21 22 23
24 25 26 27 28 29 30

Tue Jun 5 10:11:51 MDT 2018
```

اصطلاحا به این کار **output redirection** گفته می‌شود.

### متاکاراکتر کوچکتر یا < یا ورودی استاندارد

قبلما هم اشاره کردیم که بسته به برنامه‌نویس یک برنامه، ممکن است آن برنامه خروجی داشته باشد یا نداشته باشد. این برای ورودی استاندارد نیز صدق می‌کند. یعنی با توجه به نظرات و تصمیمات (!) برنامه‌نویس محترم، ممکن است یک برنامه برای ادامه کارش از کاربر یا همان یوزر، ورودی درخواست کند.



مثُل موقعی که په این یعنی چال گنده‌ها پول می‌ندازی تو شش  
ت پهت آپ معدنی پده. تا پولو ندی ادامه نمیده.



اگر برنامه‌ای داشته باشد که از ورودی یا input استفاده می‌کند، bash مستقیماً کیبورد شما را به آن برنامه متصل می‌کند.



نقش دستمون رو داره موقع پول اندامتن توی یخچاله، نه؟

یعنی از اینجا به بعد هر چی با کیبورد بنویسید و یک Enter بزنید، آن اطلاعات به عنوان ورودی یا input فرستاده می‌شود. دیگه با آن ورودی، برنامه چه کاری انجام می‌دهد به ما ربطی نداره.



چطور ربطی نداره؛ پاسخگو پاش



مثلاً برنامه یه چای محالی داده این شکلی: -----\*=---- و از تو یه عددی که توی چای محالی اول می‌خوای قدرار پکیده رو می‌پرسه. این مسئله فرقی توی نحوه خذینت تو داره عایا؟

اره. اصلاً من دوست دارم برنامه چوری کار کنه که دو دوڑا رو پذرن 5.7



هفت و نیم ۷۷۷۷۷ چطور ممکنه؟

برنامه خودمه دوست دارم اینچوری پاشه. حیلی دخالت می‌کنیا. می‌گم از سناریو پندارنیت پیدون. حدیم شخصی حالت نیست؟



برنامه تا وقتی که از یوزر ورودی می‌خواهد، متوقف می‌شود. برای مثال، در مثال دو دوتا می‌شه هفت و نیم، برنامه منتظر می‌ماند آن ۲ یا هر عددی دیگر را وارد کنید تا ضرب انجام گیرد و خروجی نشان داده شود.

حتی می‌توان با bash صحبت کرد و گفت که اگر ورودی یا input احتیاج داشتی، بجای آنکه مستقیم به کیبور وصل شوی، این input را از یک فایل بگیر.

خوبه چدیداً با bash هم حرف می‌زنی! چشم م روشن. فشار  
امتحاناتیه دیگه.



مثلا فرض کنید برنامه‌ای نوشتم (با اسم danceWithLinux) و در محیط bash اجرا کردیم. و می‌دانیم که این برنامه از یوزر ورودی می‌خواهد. اگر بخواهیم بگوییم ورودی را از یک فایل (فایلی با اسم data) بگیر، باید بنویسیم:

`danceWithLinux < data`

در این حالت برنامه danceWithLinux ورودی خود را از data می‌گیرد.

برای درک بهتر، فکر کنید برنامه‌ای به زبان پایتون نوشته شده که می‌خواهد معدل دانشآموزان یک کلاس را محاسبه کند. و حالا نمرات این دانشآموزان درون فایلی به اسم nomarat وجود دارد. برای آن که به برنامه بفهمانید که nomarat را به عنوان ورودی بپذیرد باید بنویسیم:



moadel < nomarat

یکم بازی کنیم؛ اگر این را بنویسم چه اتفاقی خواهد افتاد؟

moadel < nomarat > final.txt



عجیب مُخْرِّ تُو مُخْرِّ شد



پی ادب! در این حالت نمودار رو به عنوان ورودی به برنامه moadel می‌ده، پس از اینکه معدل رو حساب کرد، خروجی رو تُوی final.txt وارد می‌کنه.

یعنی هم از ورودی استاندارد استفاده کردیم، هم از خروجی استاندارد. و مهم‌تر اینکه به شما در خود bash نه چیزی نشان خواهد داد، نه چیزی خواهد پرسید.

فقط اینکه اینجا فایل final.txt به اصطلاح overwrite می‌شود. اگر نخواهیم این اتفاق بیافتد، دو تا علامت بزرگتر استفاده می‌کنیم:

moadel < nomarat >> final.txt



پچه پچه مُوبیه. سر په راهه. دوسُشن دارم.  
کار مُخودشو می‌کنه.



با هم به مورد عینی را امتحان کنیم. در bash برنامه‌ای داریم به اسم WC که مخفف word است. این برنامه از یوزر، ورودی می‌خواهد. برای همین متوقف می‌شود تا یوزر، ورودی‌های خودش را از طریق کیبورد وارد کند.

کاربر (که من و شما باشیم) برای اینکه به برنامه بفهمانیم ورودی‌هایی که می‌خواستیم را ارسال کردیم و تموم شد و خلاصه ولمون کن بابا و اجرаш کن، باید آخرش Ctrl+d را بزنیم:

```
sahar@dance-with-linux:~$ wc
Dance with linux
sahar shaker
bug blog
```

اینجا باید Ctrl+d را بزن:

```
sahar@dance-with-linux:~$ wc
Dance with linux
sahar shaker
bug blog
      3      7     39
sahar@dance-with-linux:~$
```

بهم می‌گه: سه خط نوشته، هفتا کلمه بود، و 39تا حرف.

فان: دروغ می‌گه من می‌خوام پشیتم پشمارمشن.





دست شما درد نکنه دیگه شدیدم دروغگو

ای چانم! کجا پودین شما تا الان؟ خودش حالی بند  
پایه چلو خودم، پیستا کارت پستال "عذیزم تو تنها  
عشق منی" خرید. شما غم به دلت راه نده.

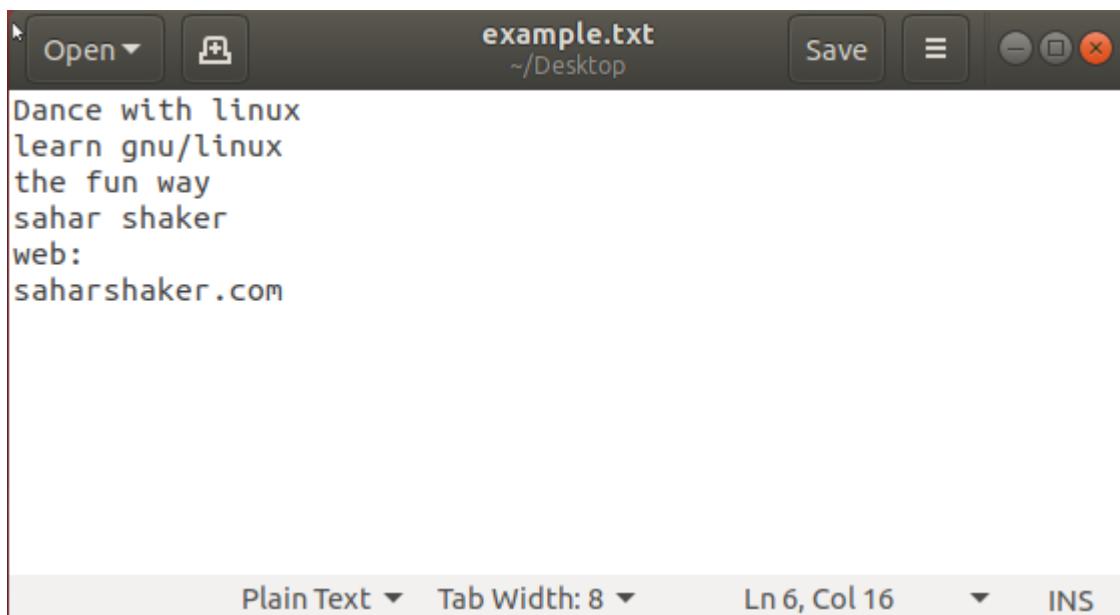


کامند WC هم مانند سایر کامندها، آپشن‌های زیادی دارد.

بازی خودمان را ادامه دهیم، به WC بگوییم ورودی را از یه فایل مثلا به اسم example بگیر و خروجی را در note.txt بریز:

فرض کنید فایل example.txt محتویاتی این چنینی دارد:

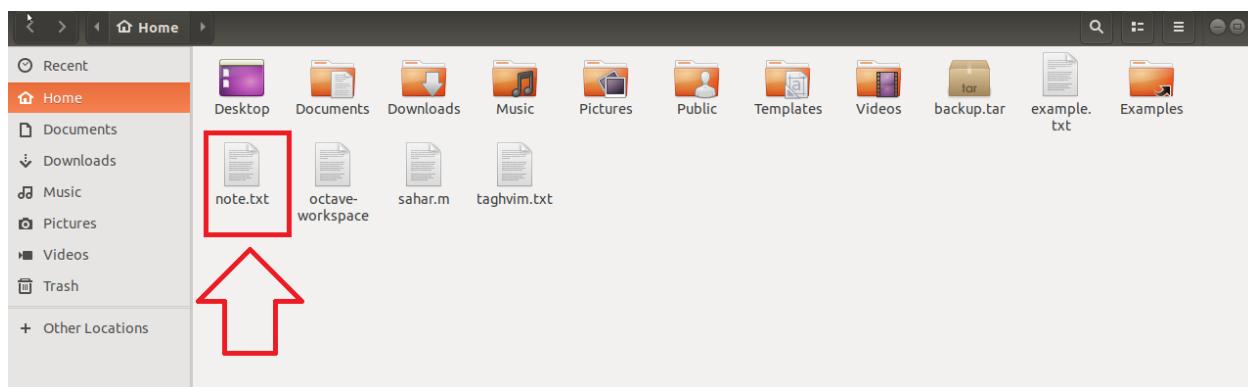




می خواهیم:

```
sahar@dance-with-linux:~$ wc < example.txt >> note.txt
sahar@dance-with-linux:~$
```

به طور خودکار، فایلی به اسم note.txt در دایرکتوری جاری ایجاد می شود که نتیجه را در آن نوشته:





تا اینجا یاد گرفتیم که چگونه به bash بفهمانیم تا مسیر ورودی استاندارد و خروجی استاندارد را عوض کند.

نکته: ما با متاکاراکتر پایپ‌لاین نیز یاد گرفته بودیم که خروجی یک برنامه را به ورودی برنامه دیگر وصل کند. بالاتر بروید پیدایش می‌کنید ():

گفته بودیم پایپ‌لاین با علامت |، خروجی برنامه اول را به ورودی برنامه دوم متصل می‌کند. مثلا در شکل زیر، با دستور ls محتويات داخل دایرکتوری dev/ را لیست کردیم، حالا لیستی که به دست آمده را به عنوان ورودی کامند WC ارائه دادیم، نشان می‌دهد:



فان: گلاب په روتون می‌شه انقدر WC نگید

شازده قراچه حساسه



```
sahar@dance-with-linux:~$ ls /dev | wc
    191      191     1210
sahar@dance-with-linux:~$
```

با bash کارهای خفن‌تری (!) می‌توانید انجام دهید. اما همگی بستگی به کارکردن شما دارد. با مدام خواندن هیچ کتابی و سایتی برنامه‌نویس نخواهید شد. کار و کار و کار.



پله از قدیما گفتن کار چوهر مرده

هر چی شما پیگید



این چش شد؛ پیرا کمنگ؟

فکر کنم چوهرش تموم شد. آخه پیکار تدر از این موجود وجود نداره!



در واقع کار پایپ‌لاین شبیه زنجیری است که کامندها را به هم وصل می‌کند.

## متغیرهای محیطی (enviroment variables)

کامندها به طور کلی به دو دسته تقسیم می‌شوند:

۱- کامندهای داخلی یا internal

۲- کامندهای خارجی یا external

کامندهای داخلی به دستوراتی گفته می‌شود که کامپایل شده‌اند و درون خود bash وجود دارند. اما تعداد آن‌ها زیاد نیست. ممکن است کامندي را تایپ کنید که داخلی نباشد در این صورت bash به دنبال فایل اجرایی آن دستور می‌رود.

یعنی عملاً می‌فرستیم دنبال نخود سیاه؟



بنابراین اگر کامندي داخلی نباشد، باید یک فایل یا برنامه کامپایل شده‌ای باشد که shell آن را پیدا کند و توانایی خواندنش را داشته باشد.



کد کامپایل شده یعنی اوّن توی زبان مثلاً توی python نوشته شیم  
و اوّن چویده و فایل‌ش رو می‌ده. دیگه shell می‌تونه اوّن پخونه

اینجا مهم است که shell در کدام فolderها به دنبال این فایل‌های کامپایل شده می‌گردد.

مثلاً فرض کنید یک برنامه‌ای با زبان پایتون نوشته‌ایم به اسم dance که درون آن خیلی ساده گفته‌ایم print('dance with linux') حالا کاری که باید انجام دهیم این است که کد کامپایل شده این برنامه dance را جایی قرار دهیم که bash بتواند شناسایی کند. بعد از اینکه درون اتاقی (پوشه‌ای، فایلی) مناسب قرار دادیم، می‌توانیم موقع داشته باشیم که با تایپ دستور dance و اینتر زدن داخل bash، دستور اجرا شده و عبارت dance with linux روی اسکرین ظاهر شود.



اسم این فایل اجرایی باید با کلمه اول کامند یکی باشد.

اینجاست که چیزی به اسم متغیر محیطی مطرح می‌شود. مثل خیلی از زبان‌ها که اجازه داریم درون آن‌ها چیزی به اسم متغیر تعریف کنیم، چنین امکانی در bash نیز وجود دارد که این متغیر در مموری ذخیره می‌شود.

در واقع bash از متغیر محیطی path لیست یکسری دایرکتوری را می‌خواند؛ دایرکتوری‌ها را به ترتیب و یکی‌یکی می‌گردد و اگر فایل باینری که اسمش با کلمه اول کامند شما یکی بود پیدا کرد آنرا اجرا می‌کند و بقیه کامند شما را که با space از هم جدا شده‌اند را به صورت آرگمان اجرا می‌کند.



من یه چیز می‌دونم، اینجا پهش می‌کن آرگمان رو په اون پراسسی که داره اچداش می‌کنه، پاس می‌ده



پو گو پینیم کی گل می‌شه؟



گلش پستگی په خودت داره و راویه شوتت! تو هم که کلا شوتی  
مطمئن پاش روی تو یکی چواب می‌ده

مثلًا کامند ping یک کامند اکسترناł است و bash آنرا اجرا می‌کند. پس bash توانسته فایل اجرایی ping را پیدا کند.



```
sahar@dance-with-linux:~$ ping
Usage: ping [-aAbBdDfhLnOqrRUvV64] [-c count] [-i interval] [-I interface]
           [-m mark] [-M pmtudisc_option] [-l preload] [-p pattern] [-Q tos]
           [-s packetsize] [-S sndbuf] [-t ttl] [-T timestamp_option]
           [-w deadline] [-W timeout] [hop1 ...] destination
Usage: ping -6 [-aAbBdDfhLnOqrRUvV] [-c count] [-i interval] [-I interface]
           [-l preload] [-m mark] [-M pmtudisc_option]
           [-N nodeinfo_option] [-p pattern] [-Q tclass] [-s packetsize]
           [-S sndbuf] [-t ttl] [-T timestamp_option] [-w deadline]
           [-W timeout] destination
```

در مورد چگونگی نوشتن متغیرهای محیطی و تنظیم آنها در فرصتی جدا بحث خواهیم کرد.



یعنی دستی دستی پیچوندی مارو سحر؟



نه نوشتن متغیرهای محیطی خودش به برنامه نویسی نیاز دارد و چند تا کامند دیگه.



پاشه ولی من نخود سیاهمو همین چا گم کردم.  
خواستم پگم حواس است هست.

## کامندهای پایه‌ای

پیش از این، در بین مثال‌ها به چندین کامند پایه‌ای اشاره شد. مثل `ls` در اینجا سعی می‌کنیم به چندین کامند دیگر اشاره کنیم. تمامی این کامندها حکم جعبه‌ابزاری را دارند که باعث می‌شوند کارهایتان را سریعتر و راحت‌تر انجام دهید.



قبل از اینکه بحث را شروع کنیم، خوب است اشاره کنیم که در بخش "پیدا کردن اطلاعات در مورد کامندها" به خوبی توضیح دادیم چگونه می‌توانیم کار کرد یک کامند را بفهمیم و همچنین آپشن‌ها و آرگمان‌هایی که هر کامند می‌تواند بگیرد را چگونه استخراج کنیم. برای همین خوب است نیم نگاهی به آن بخش کنید.

### کامند mkdir

با این کامند می‌توان فolder یا دایرکتوری ساخت.



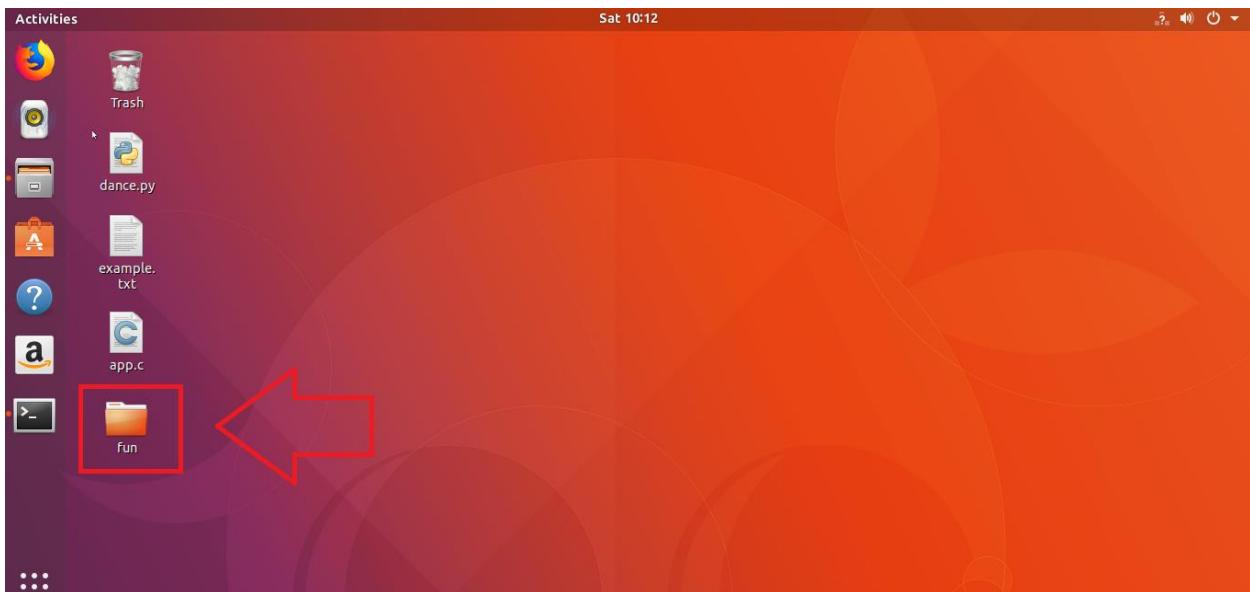
یعنی خونه دار می‌شیم؟

اول باید به این دستور، یک دایرکتوری بگیم، این دایرکتوری باید وجود داشته باشد. مثلاً می‌توانیم بنویسیم:

```
mkdir /home/sahar/Desktop/fun
```

```
sahar@dance-with-linux:~$ mkdir /home/sahar/Desktop/fun
sahar@dance-with-linux:~$
```





یعنی طرف توقع داره دیگه حضرن و اله، تا قسمت رو desktop داشته باشین و ایشون فقط بیاد اتاق رو فرشش کنه. یعنی اون قسمت fun رو خودش میسازه.



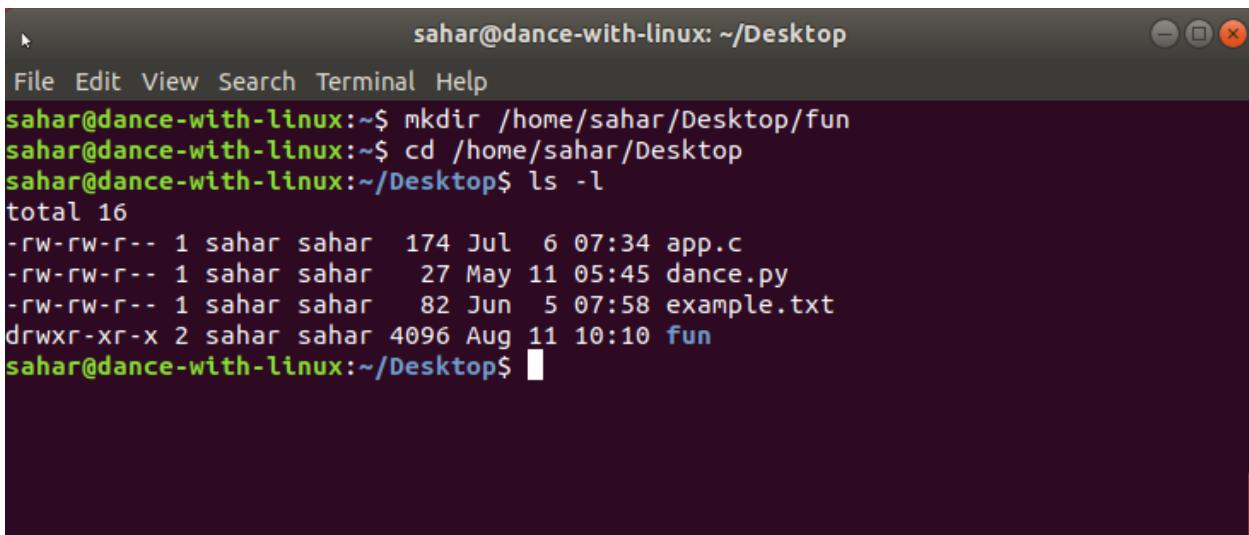
از کجا بفهمیم که تشکیل شده است؟

دستور ls محتویات داخل دایرکتوری جاری را لیست میکند. اگر در دایرکتوری جاری حضور خود را رسانده باشیم، میتوان با bash صحبت کرد و راضی به نشان دادنش کرد.

یعنی بنویسیم:

|s -|





```
sahar@dance-with-linux: ~/Desktop
File Edit View Search Terminal Help
sahar@dance-with-linux:~$ mkdir /home/sahar/Desktop/fun
sahar@dance-with-linux:~$ cd /home/sahar/Desktop
sahar@dance-with-linux:~/Desktop$ ls -l
total 16
-rw-rw-r-- 1 sahar sahar 174 Jul  6 07:34 app.c
-rw-rw-r-- 1 sahar sahar 27 May 11 05:45 dance.py
-rw-rw-r-- 1 sahar sahar 82 Jun  5 07:58 example.txt
drwxr-xr-x 2 sahar sahar 4096 Aug 11 10:10 fun
sahar@dance-with-linux:~/Desktop$
```

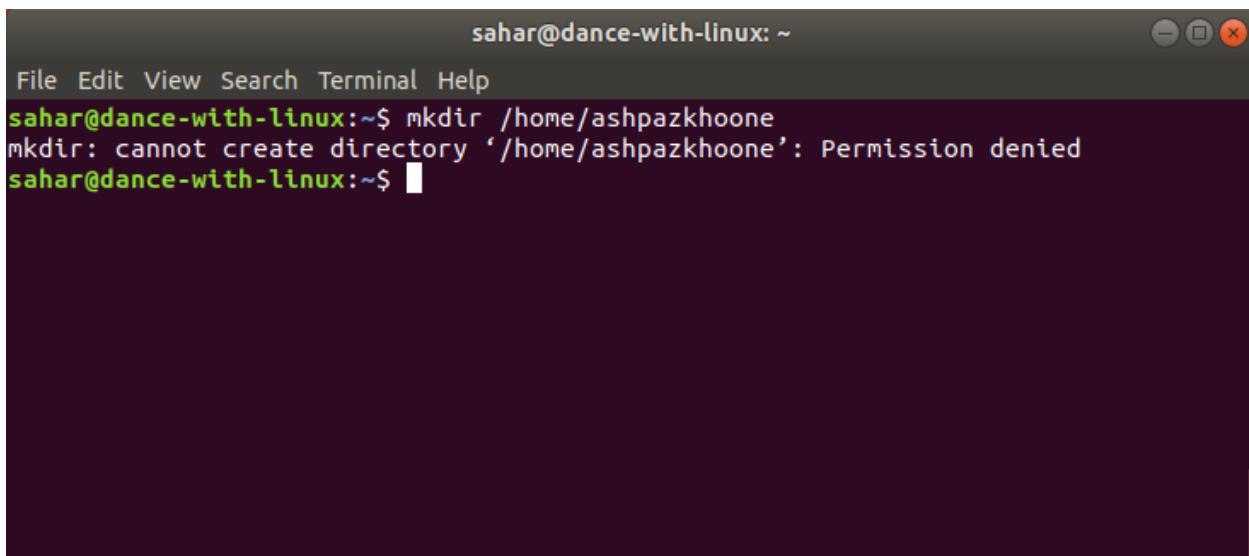
به محض لیست گرفتن، می‌بینید که فولدری به اسم `fun` ساخته شده است.



اگر بچای sahar توی دایدکتوئی پنویسم  
هیچی می‌شنهشاید دلم پخواه پره  
آشپزخونه

هیچی. ارور می‌ده. چون توقع داره حداقل وسعت پکشە و ٹا قسمت آشپزخونه رو تو داشته پاشی که نداری. یعنی یه مسیر، یه اتاق درست پاید پهش پدی ٹا پتوونه اونجا زندگی کنه





```
sahar@dance-with-linux: ~
File Edit View Search Terminal Help
sahar@dance-with-linux:~$ mkdir /home/ashpazkhoone
mkdir: cannot create directory '/home/ashpazkhoone': Permission denied
sahar@dance-with-linux:~$ █
```

سرویس چی؟ اوْ نو دارم



### آپشن p در mkdir

این دستور، یک آپشنی دارد به اسم p- که می‌توان با استفاده از آن، دایرکتوری‌های تو در تو ساخت.

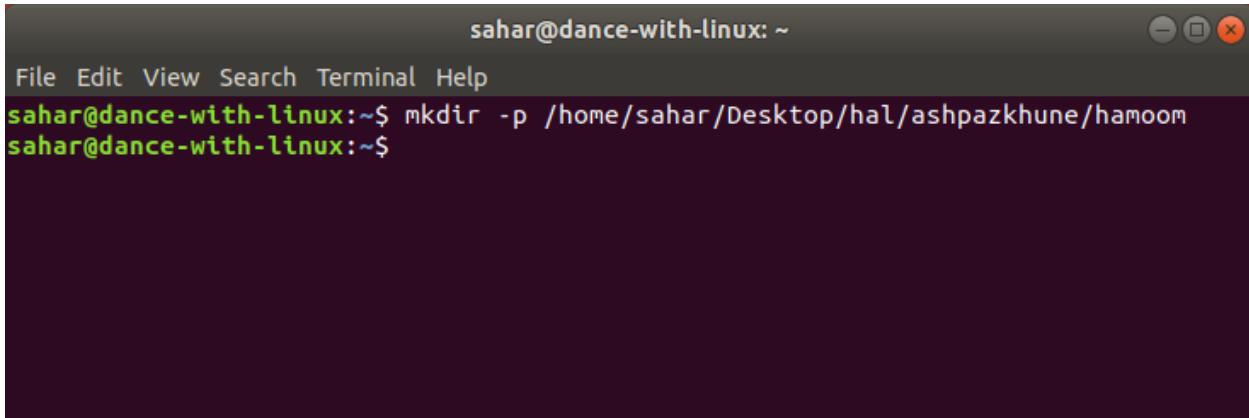


یعنی توی حال، آشپزخونه پسازیم، توی آشپزخونه، حموم پسازیم و پرو تا آخر

مثلا:

`mkdir -p /home/sahar/Desktop/hal/ashpazkhune/hamoom`





```
sahar@dance-with-linux: ~
File Edit View Search Terminal Help
sahar@dance-with-linux:~$ mkdir -p /home/sahar/Desktop/hal/ashpazkhune/hamoom
sahar@dance-with-linux:~$
```

حالا اگر از دایرکتوری، لیست بگیریم:

ls

نشان می‌دهد که اول حال را ساختیم، بعد داخل حال، آشپزخونه ساختیم و داخل آشپزخونه حموم ساختیم.

یعنی اگر از حال، ls بگیریم:

ls hal

آشپزخونه را نشان می‌دهد

اگر از آشپزخونه ls بگیریم:

ls hal/ashpazkhune

حموم را نشان می‌دهد.



```
sahar@dance-with-linux:~/Desktop$ ls
hal
sahar@dance-with-linux:~/Desktop$ ls hal
ashpazkhune
sahar@dance-with-linux:~/Desktop$ ls hal/ashpazkhune
hamoom
sahar@dance-with-linux:~/Desktop$ 
```

## نقش mkdir tree

اگر tree را هم تایپ کنیں (حالی تایپ کنیں)، فایل‌های تو در تو را به صورت درختی نشان می‌دهد.

اینجوری خونه رو از نمای پالاتر میتوینین بینین،  
ویوتون هم عالی میشه. فقط مواظب پاشین کسی  
دستشویی نپاشه



حالا نوبت شماست. چهارتا فolder تو در تو پسازین بینیم چقدر  
خونه سازی پلدين؟ مثلاً اسم فolder اول رو پذاريid 1، بعد داخل  
يک، فolder 2 رو پسازين، بعد داخل 2، فolder 3 رو پسازين.  
چو اپهارو مي تونيد توی این آدرس، کامنت کنین ٹا پقیه هم  
بینین



راستی mkdir معلومه مخفف make directory است.



چطوری چندتای دایرکتوری رو پاهم توی  
یک خط کامند پسازیم؟

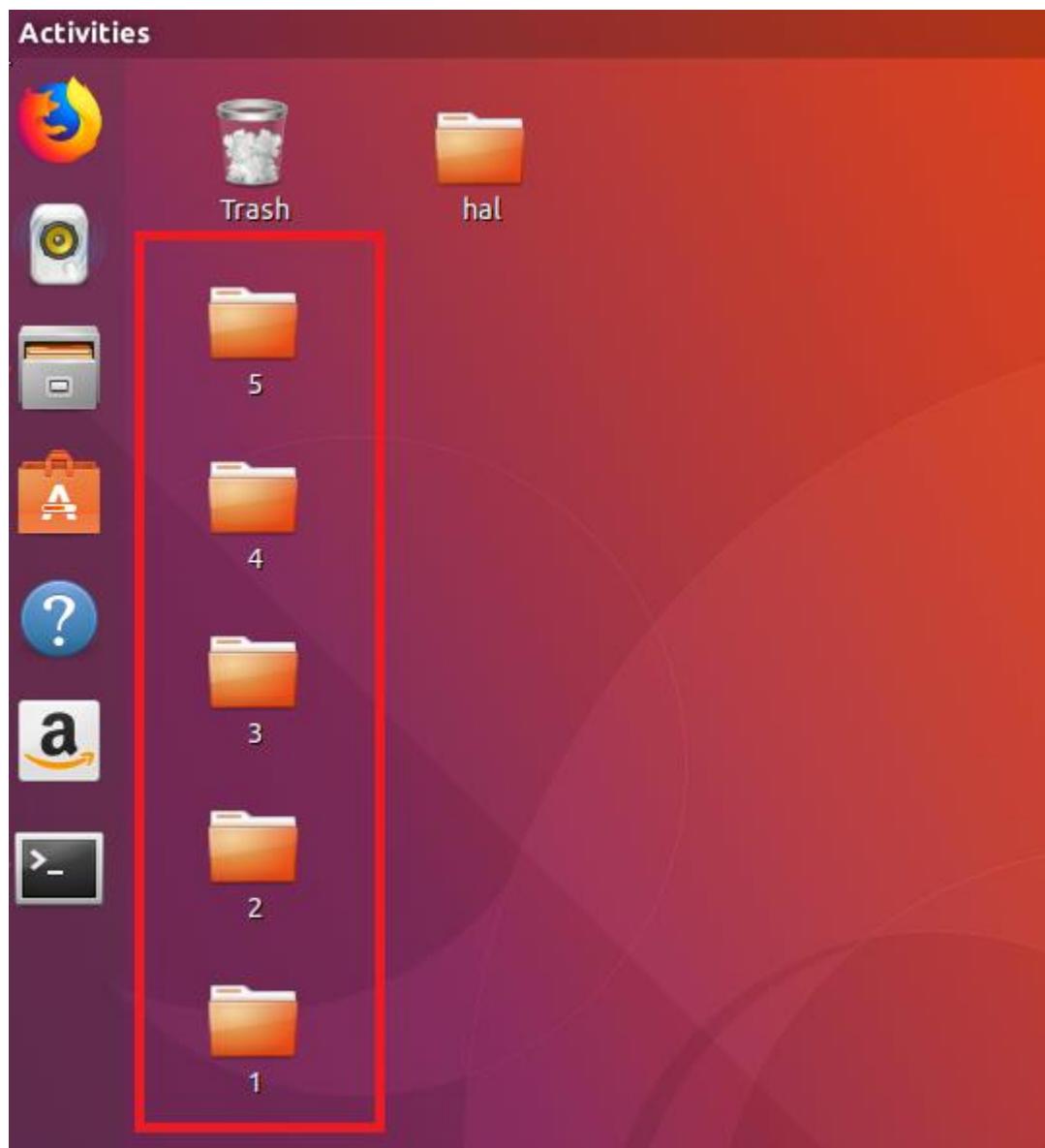
برای این کار، کافی است بین اسامی، فاصله یا space قرار دهید:

`mkdir 1 2 3 4 5`

```
sahar@dance-with-linux: ~/Desktop
File Edit View Search Terminal Help
sahar@dance-with-linux:~/Desktop$ mkdir 1 2 3 4 5
sahar@dance-with-linux:~/Desktop$
```

در این صورت 5 فولدر، در دایرکتوری جاری شما (اینجا دسکتاپ) ایجاد می‌شود.





پا چی می فهمیدیم ټوی کدوم اټاقيم یا همون  
دايدكتوري چاري رو از کجا پيدا مي کردیمه؟



پا ټایپ pwd



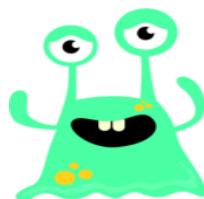
چرا هر چی می‌نویسم، صدایی ازش درنمیاد؟  
یعنی چرا هیچ چیزی روی صفحه نشون نمیده؟



این کامند، مانند بسیاری از کامندهای دیگر، تا زمانی که به error و مشکلی نخورد، صدایش در نمیآید. آهسته کار خود را می‌کند و دستوری که به آن داده‌اید را به خوبی گوش می‌دهد.

در این مثال، کافی است به دایرکتوری جاری خود، مثلًا اینجا desktop بروید تا ببینید که 5 فolder به اسم‌های 1 و 2 و 3 و 4 و 5 ساخته شده است.

اصلًا می‌شه توی bash موئد از دایرکتوری چاری،  
اگر قت ٹا پیغامیم که کارشو کرده پا نه.



خوب اینچوری اذیت می‌شم، پیکار دیگه کن  
که من خروجی رو پیینم.



نمی‌شه زد تو گوشش ۳ صداسن درپیاد و اعتراض  
کنه چه کارایی کرده؟



چه مخشن

با اضافه کردن آپشن -v که مخفف verbose به معنای پر حرف است، می‌توان خروجی را در همان صفحه مشاهده کرد.

یعنی در این حالت، به شما یک پیام نشان می‌دهد که من ساختم.

`mkdir -v man na manam`

```
sahar@dance-with-linux: ~/Desktop
File Edit View Search Terminal Help
sahar@dance-with-linux:~/Desktop$ mkdir -v man na manam
mkdir: created directory 'man'
mkdir: created directory 'na'
mkdir: created directory 'manam'
sahar@dance-with-linux:~/Desktop$
```

همان‌طور که در تصویر می‌بینید به شما اطلاع داده است که سه فolder به اسم "من نه منم" ساخته است.



## پاک کردن فایل، کامند rmdir

برای پاک کردن یک فایل، از کامند `rmdir` استفاده می‌کنیم.

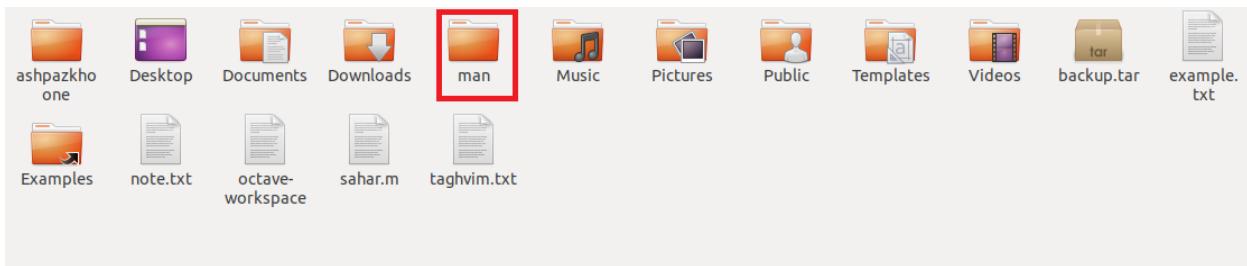


اون rm اولش و اسه remove هست؟



یاه! ا-neckدر در کش سخته؟

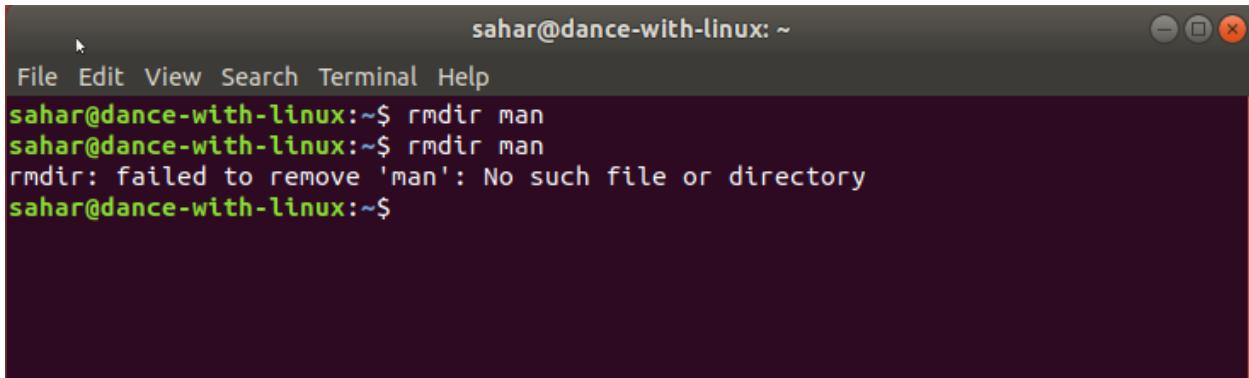
مثلا برای پاک کردن فایل `man` می‌توان نوشت:



```
sahar@dance-with-linux: ~
File Edit View Search Terminal Help
sahar@dance-with-linux:~$ rmdir man
sahar@dance-with-linux:~$
```



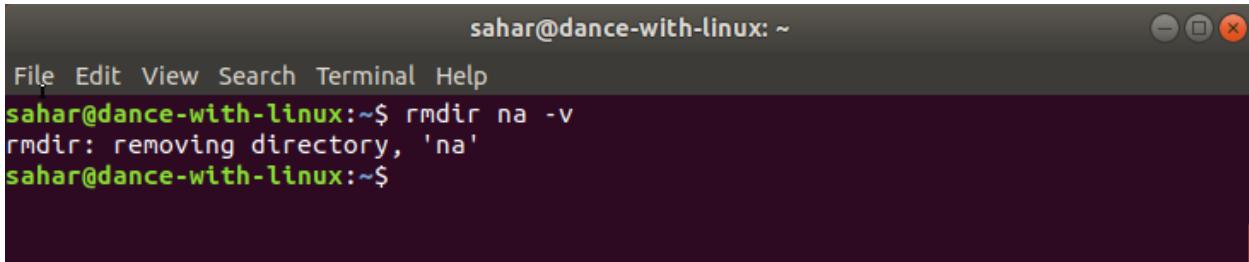
اگر دوباره بزنم:



```
sahar@dance-with-linux: ~
File Edit View Search Terminal Help
sahar@dance-with-linux:~$ rm man
sahar@dance-with-linux:~$ rm man
rm: failed to remove 'man': No such file or directory
sahar@dance-with-linux:~$
```

ارور می‌ده که همچین فایلی کلا نیست که بخواهم پاکش کنم.

این کامند هم کامند ساکتی است برای به حرف درآوردن و اینکه یک پیغام نشان دهد که من حذف کردم، آپشن `-v` - که قبل از گفته شد، کنارش می‌آید:



```
sahar@dance-with-linux: ~
File Edit View Search Terminal Help
sahar@dance-with-linux:~$ rm na -v
rm: removing directory, 'na'
sahar@dance-with-linux:~$
```

اگر از دایرکتوری جاری، `ls` بگیریم، می‌بینیم که پاک شده است.

### تفاوت فایل با دایرکتوری

دایرکتوری چیزی نیست که روی هارد دیسک وجود داشته باشد و دارای `content` باشد؛ بنابراین پاک کردنش چندان هم معنا ندارد. پاک کردن دایرکتوری یعنی بیایم از درخت دایرکتوری‌های فایل سیستم حذف کنیم. از یک دیتابیسی حذفش کنیم. با این حال اگر یک دایرکتوری پُر باشد، نمی‌توان آن را پاک کرد.

از قبل می‌دانیم که برای دیدن دایرکتوری‌ها کافی از کامند `tree` استفاده کنیم.

اگر `tree` را تایپ کنیم:



در tree می‌بینیم که در دایرکتوری 1، دایرکتوری 2 وجود دارد، و در دایرکتوری 2، دایرکتوری 3 است و بعد هم دایرکتوری 4 در دل دایرکتوری 3 وجود دارد. پس اگر بزنم: rmdir 1 ارور می‌دهد چرا که دایرکتوری 1، پُر است.

زورش نمی‌رسه rmdir یعنی می‌خوای بگی  
دایرکتوری پاک کنه ولی فایل رو می‌تونه پاک کنه؟



همینه. rmdir زورش په دایرکتوری نمی‌رسه و فقط می‌تونه از لیست tree حذف کنه با این حال نمی‌شه یه دایرکتوری رو پاکش کرد.



اونوقت فیلسوف چان "پاک کردن" با "حذف کردن" چه تفاوتی داره؟

از لحاظ ادبیاتی نمی‌دونم ولی وقتی می‌گم پاک می‌کنه یعنی delete و یعنی دیگه نیست، دیگه وجود نداره. ولی وقتی می‌گم پاک کنه یعنی remove و یعنی فقط از لیست tree می‌شه دایرکتوری رو حذف کرد. ما فایل رو پاک می‌کنیم، ولی دایرکتوری رو تنها از لیست حذف می‌کنیم.





خوبی any way

این معلومه نفهمیده که انگلیسی گفت.



## ساخت فایل

برای ساختن یک فایل خالی خیلی راه وجود دارد.

"خ" رو دوست ندارم میشه زیاد استفاده نکنی؟



کجا دیدی؟

اینچا دیگه "خالی خیلی"





آخ<sup>ن</sup>چون که تو خوشت نیاد؛ راستی، خلیل توی خیاپون زید درخت خیار می خورد، خیلی هم خوشن می اومد. انقدر هم خندید، فکر کنم تا فیها خالدونت سوخت. اصلا خدا پاید اسم حوا رو می داشت خوا شاید اینجوری پشیریت به طریق پهتری می رفت ما هم راخت تر پودیم. ملاصه که خعلی خوش گذشت. می خواه از این به بعد په هوا هم بگم خوا. خالت خوش رفیخ. دور هم می خوایم پشینیم و یه تجدید نظر توی واژگان پکنیم و خیلی خیزای خدید پسازیم.

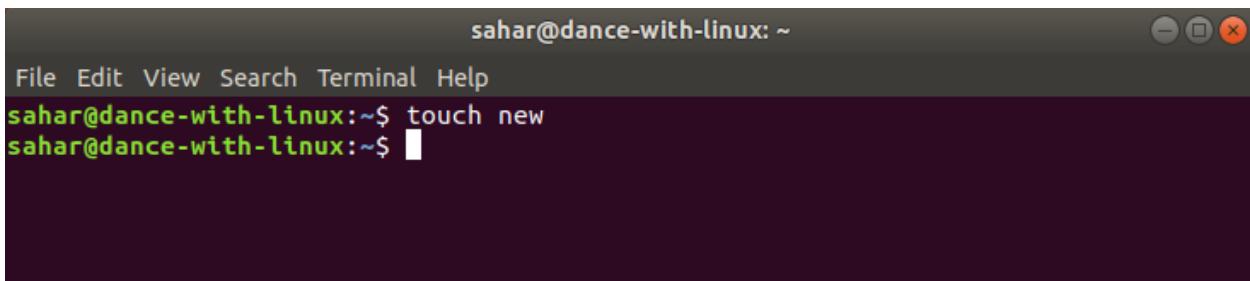
یکی از راهها برای ساختن فایل این است که بگیم "هیچی" رو ریدایرکت کن به فایلی مثل khali.txt و اینجوری یک فایل خالی به اسم khali.txt ایجاد شده است.

```
sahar@dance-with-linux:~$ >khali.txt
```

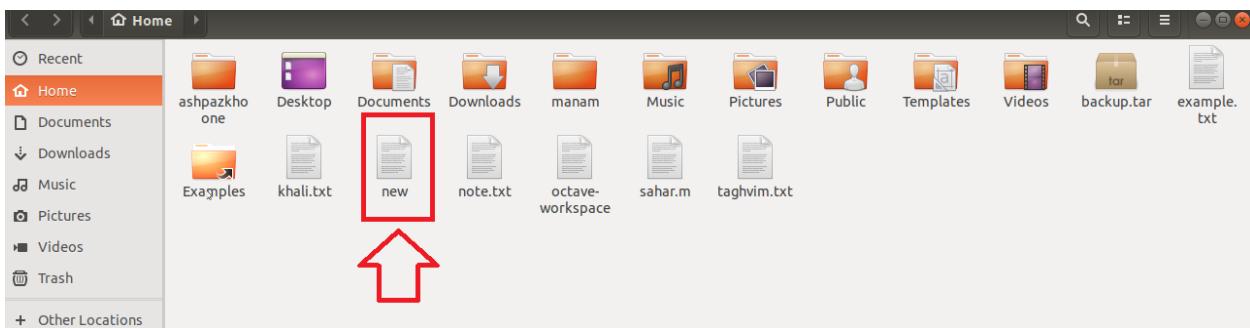
فایل ایجاد شد:



یک راه دیگر، استفاده از کامند touch است. مثلا اگر بزنم:



```
sahar@dance-with-linux: ~$ touch new
sahar@dance-with-linux: ~$
```



کار اصلی touch این است که آخرین زمانی که یک فایل modify شده را تغییر می‌دهد.

یعنی اگر پدم در یخچال رو باز کنم، و کامند touch رو پذنم، زمان آخرین پاری که در یخچال رو باز کردم، عوض می‌شود؟



آره. کامند گنج یا همون touch، به ازای هر فایلی که شما ذخیره می‌کنید، یک عالمه اطلاعات دیگر هم، مثل زمان ایجاد اون فایل، آخرین زمان تغییر اون فایل و... ایجاد می‌شود.



صاحب فایل رو هم نشون می‌ده. به اینا می‌گن meta data

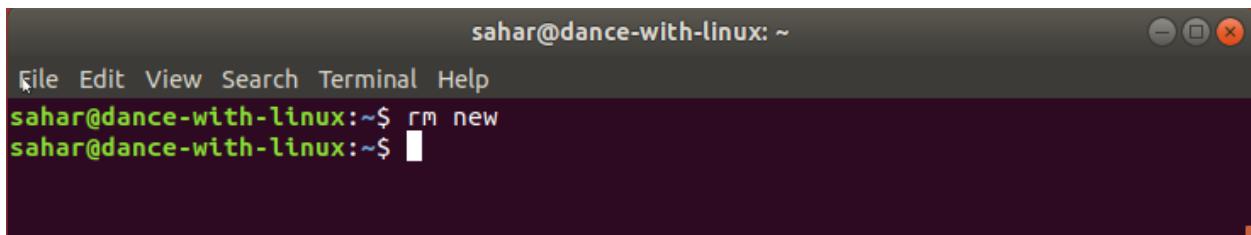


ولی چرا همچین واکنشی نشان می‌دهد؟ یعنی چرا می‌توان برای ایجاد فایل از آن استفاده کرد؟ پاسخ ساده است. در رودروایسی گیر می‌کند! مثلاً می‌گوییم `data meta` فایل `sahar` را اصلاح کن، می‌گردد، فایل `sahar` را پیدا نمی‌کند. یعنی فایل `sahar` کلا وجود ندارد. پس آن را می‌سازد! نجابت یعنی همین.

### کامند rm

جز `rm` کامندی داریم به اسم `rmdir`

این کامند فایل پاک می‌کند اما در مقابل حذف یک دایرکتوری عاجز است. یعنی می‌توان نوشت:

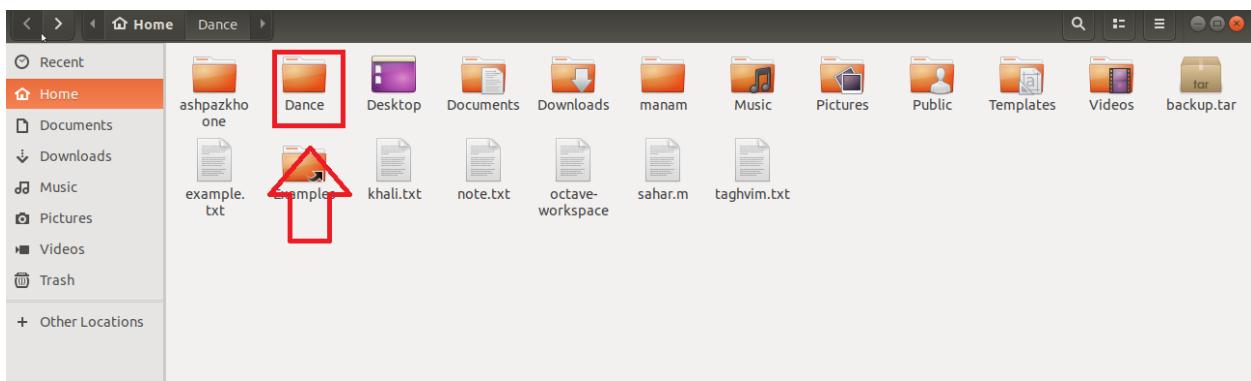


```
sahar@dance-with-linux: ~
File Edit View Search Terminal Help
sahar@dance-with-linux:~$ rm new
sahar@dance-with-linux:~$
```

فایل را پاک می‌کند.

این کامند نمی‌تواند "دایرکتوری" پاک کند؛ منظور از اینکه دایرکتوری را پاک کنیم، یعنی تمام محتویات داخل آن پاک شود، و بعد هم خود فolder پاک شود.

مثلاً فرض کنید در دایرکتوری جاری، یک فolder ساختیم به اسم `Dance`، اگر بخواهیم با دستور `rm` این دایرکتوری را پاک کنیم:



```
sahar@dance-with-linux: ~
File Edit View Search Terminal Help
sahar@dance-with-linux:~$ rm Dance
rm: cannot remove 'Dance': Is a directory
sahar@dance-with-linux:~$
```

ارور می‌دهد که Dance یک دایرکتوری است و نمی‌تواند آن را پاک کند.  
اما برای پاک‌کردن دایرکتوری Dance با استفاده از دستور rm می‌توان از آپشن این دستور،  
یعنی -r - کمک گرفت.

در واقع این آپشن در کنار کامند rm، اول محتویات داخل دایرکتوری Dance را پاک می‌کند  
و بعد هم دایرکتوری Dance را حذف می‌کند.

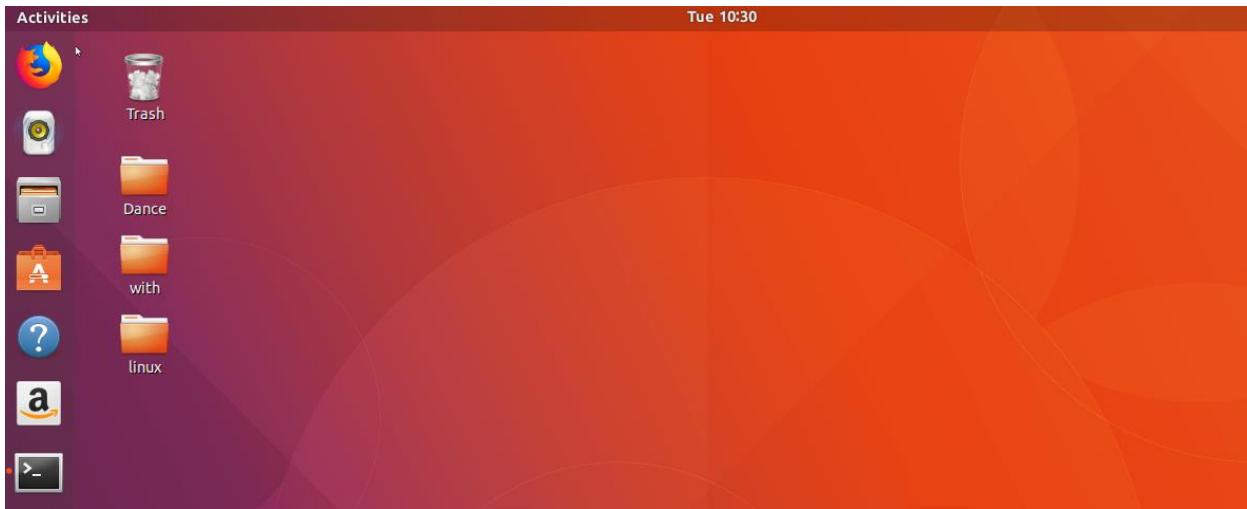
```
sahar@dance-with-linux: ~
File Edit View Search Terminal Help
sahar@dance-with-linux:~$ rm Dance
rm: cannot remove 'Dance': Is a directory
sahar@dance-with-linux:~$ rm -rv Dance
removed directory 'Dance'
sahar@dance-with-linux:~$
```



فان: rm هم پچه کم حرفیه پرای په حرف درآوردنس از -v - کمک  
گرفتیم، پرای همین نوشتم -۱۷



مثلا اگر سه تا فایل بسازم یکی به اسم dance و دیگری به اسم with و آخری به اسم linux و حالا بخواهم پاک کنم و یک پیغام هم نشان دهد می‌نویسم:



```
sahar@dance-with-linux: ~/Desktop
File Edit View Search Terminal Help
sahar@dance-with-linux:~/Desktop$ rm -rv *
removed directory 'Dance'
removed directory 'Linux'
removed directory 'with'
sahar@dance-with-linux:~/Desktop$
```

نکته: علامت \* همه فایل‌های داخل دایرکتوری جاری (دایرکتوری‌ای که هستید) را پاک می‌کند.



یه سوال پرسه؟ تو همیشه قیافت اینچوریه اگر  
خانواده ازت پرسن پیا پریم مهمونی می‌ری؟ چچوری  
واکنش نشون می‌دی کله؟



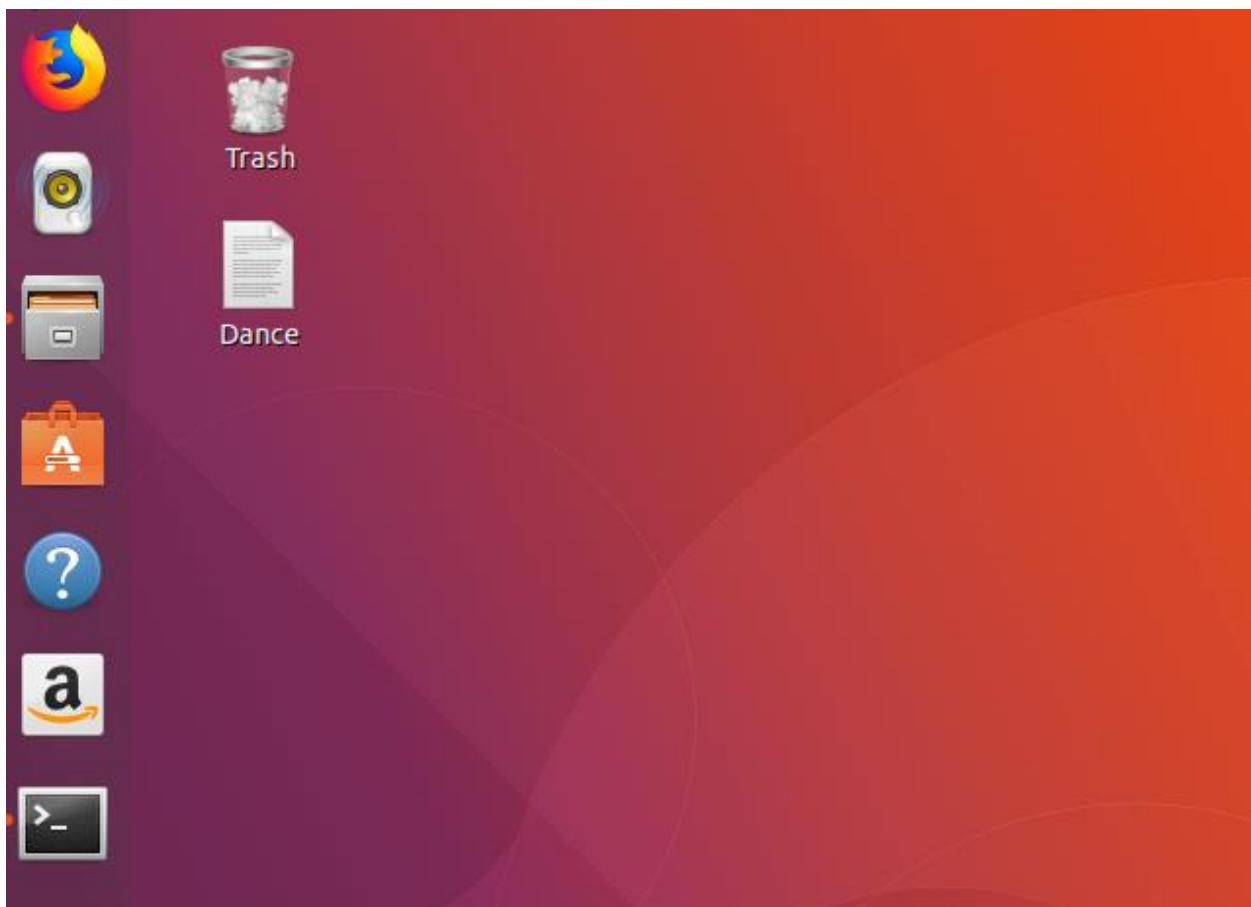
نه می گم شما پرید من تیپ خوردم.

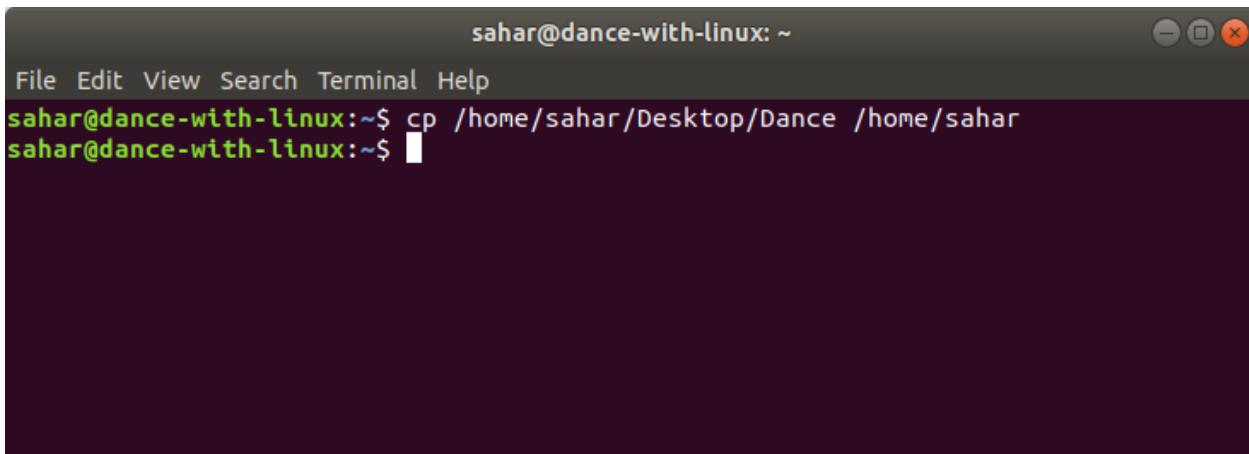


## cp کامند

cp کامندی است که برای کپی کردن فایل‌ها استفاده می‌شود. با آن می‌توان یک یا چندین فایل را در فولدری کپی کرد.

مثلا فرض کنید فایل Dance که بر روی دسکتاپ هست را می‌خواهیم در دایرکتوری home کپی کنیم:





```
sahar@dance-with-linux: ~
File Edit View Search Terminal Help
sahar@dance-with-linux:~$ cp /home/sahar/Desktop/Dance /home/sahar
sahar@dance-with-linux:~$
```

یعنی بعد از cp دایرکتوری جایی که فایل در آن است را می‌دهیم بعد یک space یا همان فاصله خالی و بعد جایی که می‌خواهیم فایل در آن ذخیره شود.



آدرس اتاقی که هست رو پده و پنه می‌خوای توی کدوم اتاق کپی شه

اگر من بودم از معشوقم ده تا کپی می‌گرفتم توی چاهای مختلف نگهش می‌داشم



اشتباه می‌کنی خب خدا، چیزی که بیوقتنه روی دور تکرار و همسش پیشیش که دیگه دوستش نداری



خوشحالم که شماهم از این چیزها حاليتونه چناب!

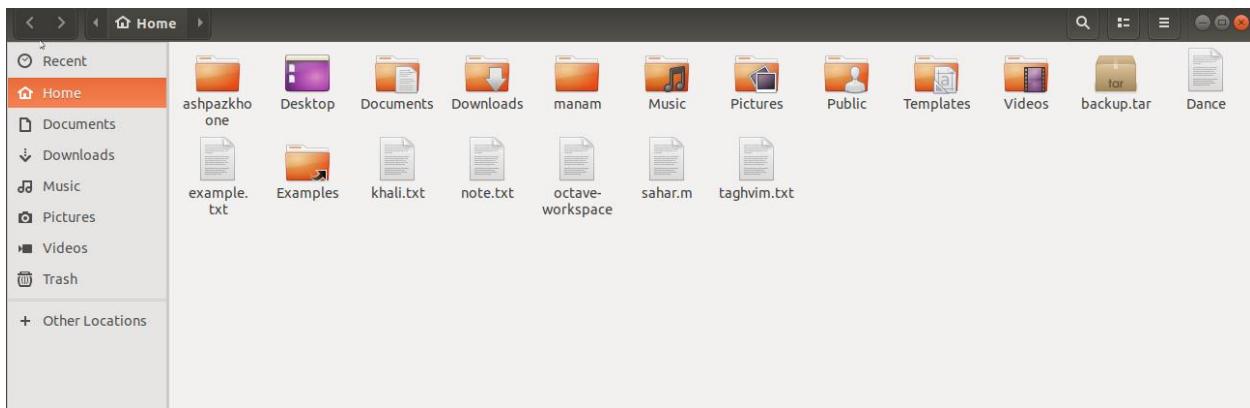




پا خودم عهد پسته پودم که دیگه تو رو شیر فهم نکنم



عمت شیر فهم شه پی شخصیت... من محرفه میشم (قیتاً)



مثلما فرض کنید در دایرکتوری home هستیم، حالا می‌خواهیم از روی دسکتاپ، فایل with را داخل home که هستیم، کپی کنیم:

برای اینکار اول باید آدرس جایی که with در آن هست را بدهیم، بعد یک فاصله، حالا بجای اینکه آدرس home را که دایرکتوری جاری است بدهیم، بجایش یک نقطه می‌گذاریم:

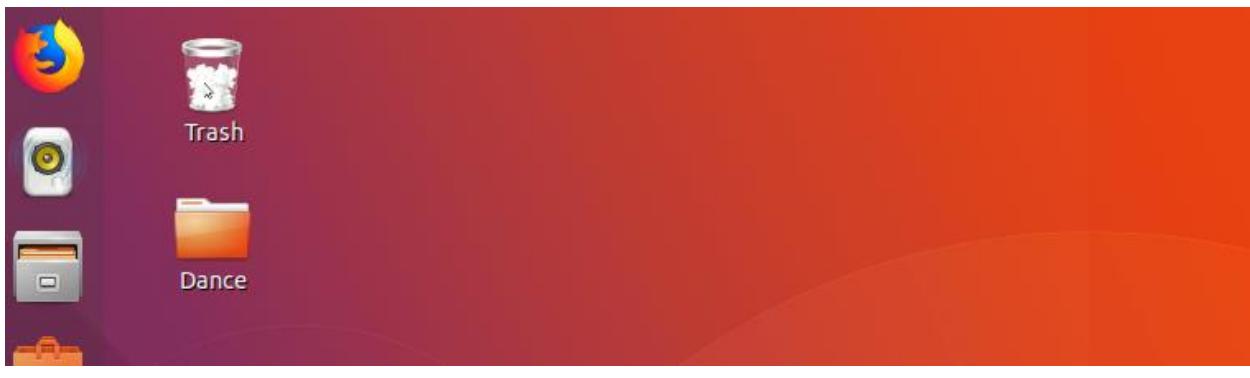
```
sahar@dance-with-linux: ~
File Edit View Search Terminal Help
sahar@dance-with-linux:~$ cp /home/sahar/Desktop/with .
sahar@dance-with-linux:~$
```

یعنی برای دستور cp نیازی نیست که آدرس دایرکتوری جاری را بدهید، کافی است یک نقطه بجایش بگذارید.





حالا مشکلی که با cp وجود دارد، این است که cp فایل جابجا می‌کند اما دایرکتوری نه.



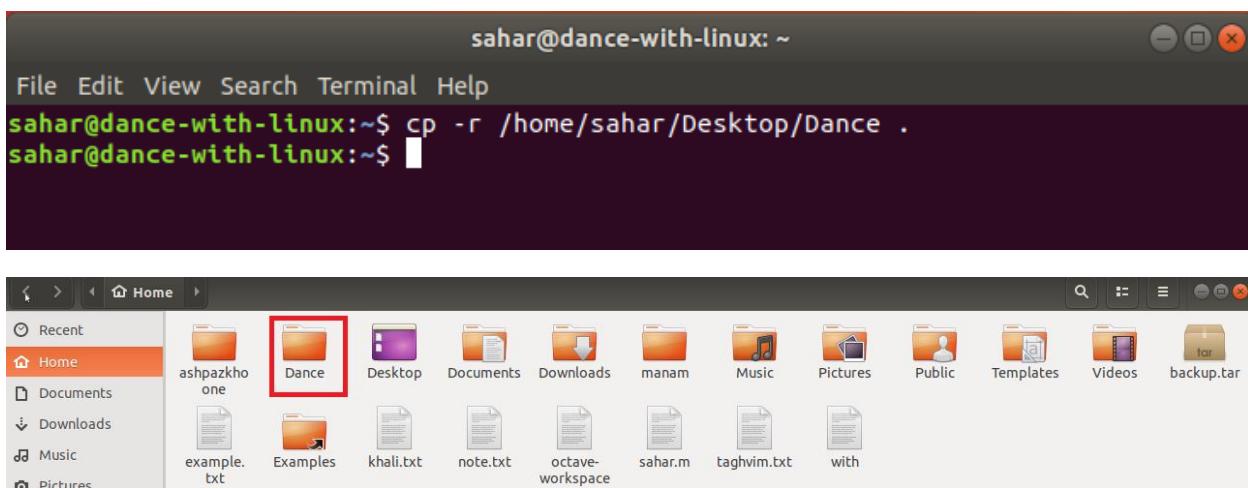
```
sahar@dance-with-linux: ~
File Edit View Search Terminal Help
sahar@dance-with-linux:~$ cp /home/sahar/Desktop/Dance .
cp: -r not specified; omitting directory '/home/sahar/Desktop/Dance'
sahar@dance-with-linux:~$
```

ارور می‌دهد.

کپی کردن برای یک فایل معنا دارد. یعنی یک فایل داریم، که داخل این فایل محتوایی وجود دارد حالا می‌توانیم آن را جابجا کنیم. اما دایرکتوری خودش وجود ندارد، صرفاً در tree یک اسم دارد و کپی کردن مفهومی برایش ندارد.

اما وقتی به طور عادی حرف az کپی زدن می‌شود، یعنی توقع داریم یک فایل جدید ساخته شود و محتوای آن فایلی که می‌خواهیم درون آن هم کپی شود. برای اینکار یک آپشن -r- کنار کامند cp می‌آید:





اگر هم تایپ کنیم `cp -rv` ، آپشن `v` پیغام نشان می‌دهد که چه چیزهایی را در کجا تایپ کرده است.

نکته: اگر فایلی که داریم جابجا می‌کنیم، از قبیل در مقصد وجود داشته باشد، آن فایل بازنویسی می‌شود، یعنی پاک می‌شود و بعد فایل جدید جایگزین آن می‌شود؛ به عبارتی دیگر فایل قبلی پاک می‌شود! برای اینکه این اتفاق نیافتد از آپشن `b`- استفاده می‌کنند. یعنی با استفاده از این آپشن، حتی اگر قبلاً هم این فایل در مقصد وجود داشته باشد، اسم قبلی عوض می‌شود و فایل جدید کپی می‌شود. در واقع شما هر دو ورژن را خواهید داشت.



این فصل تمام شد ولی فکر کردیم که چقدر کاربردی تر می‌شه اگر در مورد shell scripting پیشتر بحث بکنیم و بعد بقیه‌ی سرفصل‌هایی که توی اکثر آموزش‌های لینوکس می‌گن رو بگیم. پرای همین فصل بعد پرای آپدیت بعدی به طور خاص در مورد shell scripting حرف خواهیم زد!





هی فیلسوف اعظم ما داریم می‌ریم، حالا خردپیار و  
باقالی پار کن



مگه باقالی‌ها رو با خودت نمی‌بری؟



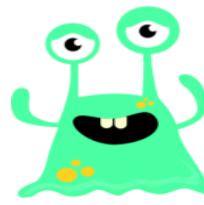
نه گذاشتم و اسه تو پوسته بکنی



قصده چیه؟



فیلسوف اعظم، عمنه! هیچی پاپا سحریه ساینت زده، دید آپدیت‌های اینجا داره گند پیش می‌رده پرای همین از این په بعد برای کتاب پسنه‌هاشو اونجا می‌داره، و هم اینکه از دنیای لینوکس و برنامه‌نویسی می‌خواهد پسنه‌های کوتاه پنوهیسه



چی؟!!!!



مدرسان شدیف!!

مگه ما مسخره ايم پا اين همه آدم جمع شيم پرييم اونجا؟؟



تازه كچاشو ديدى؟؟ همه فك و فاميلا توئنم اونجان،  
همون صفحه اول هم نشونت دارن

مي گم رقصارو دارن مي گيرن. ما رو بخاطر علاقه سحر  
نگيرن پندارن تو گونی. نکنه اسم سايت چديش  
دنس گذاشت؟



كچاست؟

مدرسان شريفا!!



نه اسم سايت "پاگ"، فقط اونجا يكم موقع حرف زدن رعایت  
کنین قیلند مومن نکنن. حچاپتون رو هم رعایت کنین گشت ارشاد  
نگيره. حرف از اودلار و گرونيش هم نزنيد. اين مهره رو هم  
نيديم آيدو پرها. يا تشكير، مدعييته يارك



خواسته  
از معاشر  
کار  
فکر  
خواسته  
از  
خواسته  
از  
خواسته  
از



میتوانم عرض کنم

@saharshaker





*Dance with Linux*

[Saharshaker.com](http://Saharshaker.com)

Update: 97/8/1