

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

صرف ا جهت اطلاع!

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

خیلی مرسی!

حمایتون به پیشرفت این کار خیلی کمک می‌کنند؛ برای همین لینک حمایت مالی توی همین صفحه گذاشتیم. بدون حمایت مالی شما هم روند نوشتن این کتاب به آرومی پیش خواهد رفت و **رایگان** منتشر خواهد شد، و مطمئناً پول تنها راه حمایت نیست اینکه به دست افراد بیشتری برسوند، بزرگترین حمایته



حمایت مالی

<https://zarinp.al/@saharshaker>



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 سوالات شروع شد. آخه اگه کسی قبل
سیستم عاملی کار نکرده باشه که په فکر لینوکس نمی‌افته. فقط
خواهشنا با این سوالات چایی نگو فامیل منی، مرسی اه.

این کتاب تمدینم داره؟



از اینکه فقط بگوییم چیکار باید بکنید شاید چندان مفید نباشد. از این رو تصمیم گرفتیم تا تمریناتی هم قرار دهیم. (در فصل‌های ابتدایی تمرینی نیست!). والبته جواب این تمارین در pdf جداگانه در صفحه کتاب می‌توانید بپیدا کنید. با صرفا خواندن این pdf لینوکس یاد نخواهید گرفت! باید خودتان هم تمرین کنید.



چرا نمی‌ذاری بسیار من سوالمو پیرسم؟؟؟ مسئولین
چرا پیگیری نمی‌کنن؟؟؟

یه کلام از مادر عروس، خودش این شکلیه عروس چه شکلیه آخه.
مسئولین اگه رسیدگی بعوان پکنن اول دخته تو رو فیلترا
می‌کنن. بعدشم کی چلو تو گرفته؟؟؟ داد می‌زئی همیش. بگو یا
چوایپ 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 افتاده 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 فراگیرشدن 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
نذارین!

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



Hey guys, it's john

یه ربات هم گذاشتیم پرای آدمای محسنه تر از خودم، په
هر دلیل اگر حال نداشتین ټا سایت پرین و احیانا
دیدید چایی سوتی دادیم، قبل اینکه آپرمون در سطح
محلى، استاني، کشورى و پين المللی پرده په این ربات
تلگرام پفرستید، تو سایت هم نوشته که چه بهتر

@saharshakerbot





bug.saharshaker.com

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
41	نگاه کمی فنی‌تر به سیستم‌عامل
47	از Debian تا Red Hat
49	Fedora
52	نصب لینوکس
54	نصب لینوکس روی virtual box
65	نصب لینوکس روی سیستم
67	فصل دوم: قلب لینوکس
68	شروع کار با دسکتاب
75	قدمزدن در فضای دسکتاب
82	کرنل (kernel)

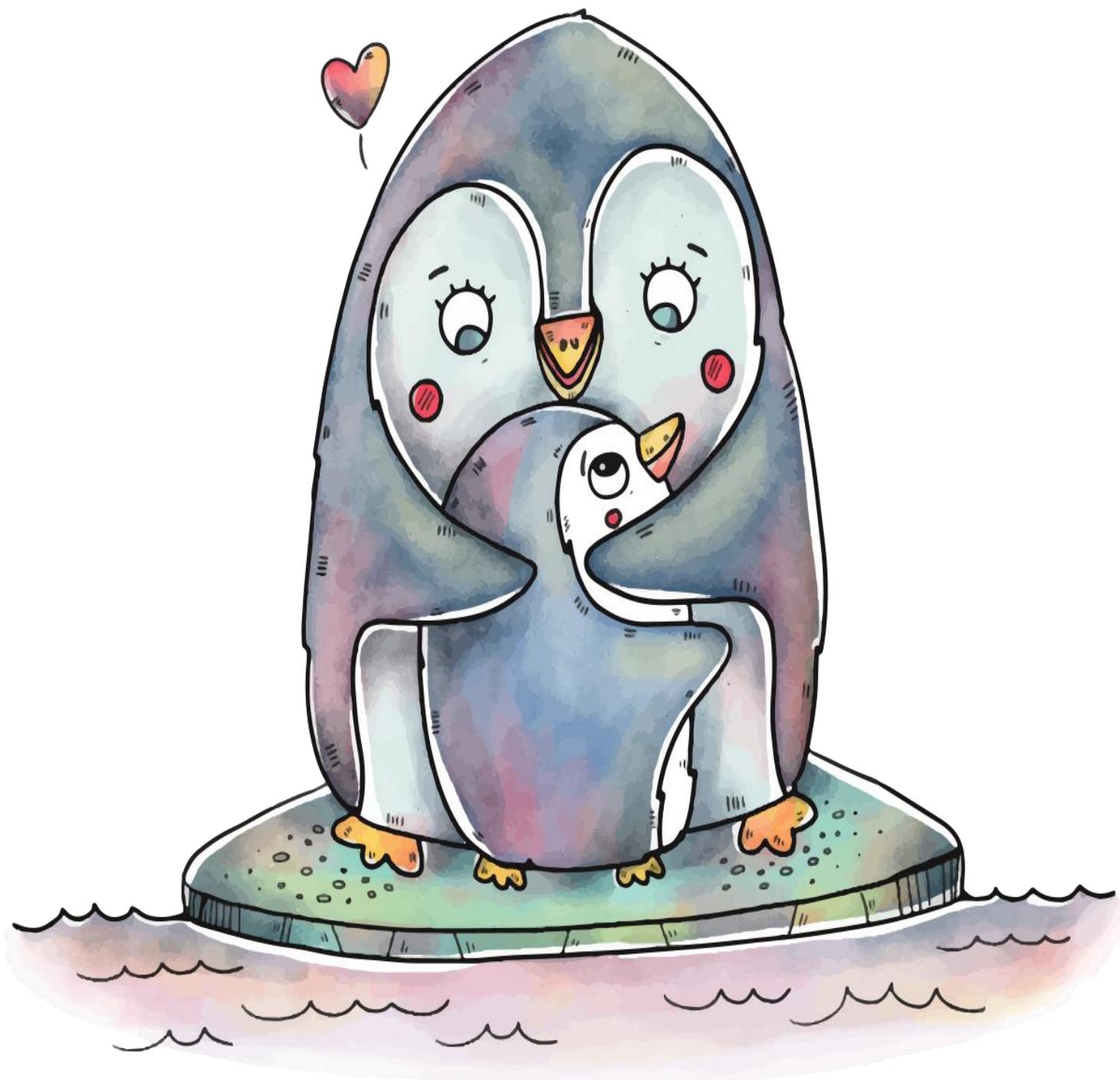
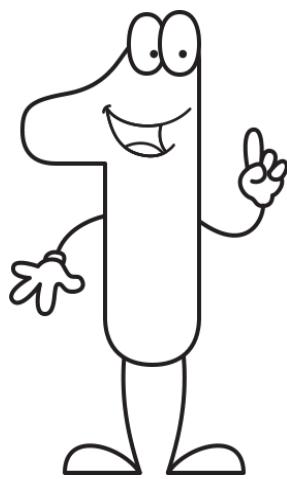
84	چگونه به یک shell دسترسی داشته باشیم؟
87	چگونه کار می کند؟ Bash
88	بررسی لاین خود
88	فهمیدن shell دیفالت
90	بررسی دایرکتوری ها
91	چجوری از یک shell خارج شویم؟
91	نحوه نوشتمن کامندها یا command syntax
92	آپشن ها (options)
97	آپشن هایی که مقدار می پذیرند
99	پیدا کردن اطلاعات در مورد کامندها
99	1- چک کردن PATH
101	2- استفاده از کامند help
102	3- می توانید help- را با دستوری که می خواهید به کار ببرید
105	4- دستور man
107	5- استفاده از info
108	6- استفاده از دستور gnome-help
108	آرگمان ها (arguments)
109	جایی که آرگمان ها با آپشن ها قاطی می شوند
111	متا کاراکترها
112	متا کاراکتر نقطه ویرگول

113	متاکاراکتر space
113	متاکاراکتر امپرسند &
114	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)

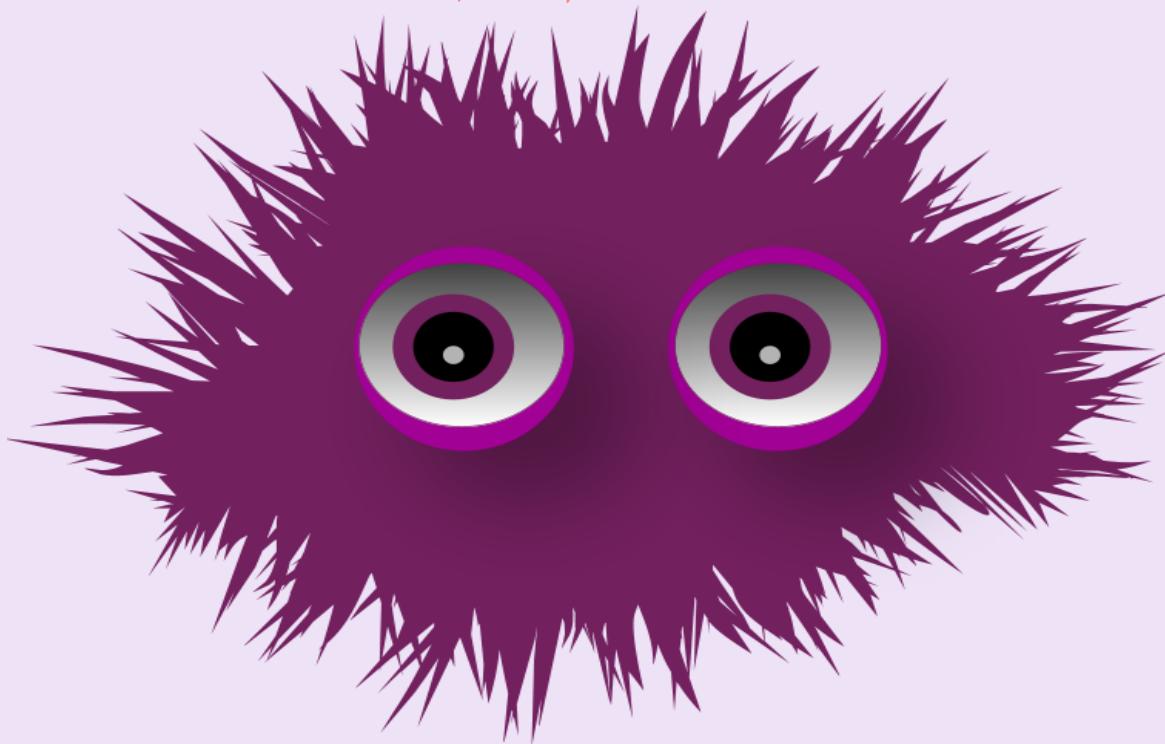
فصل اول

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 ذهن‌تان 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 ریموند

برنامه‌نویس، مهندس نرم‌افزار، نویسنده مقاله کلیسا و بازار، پیشتاز دنیای اپن‌سورس



Larry Augustin

لری آگوستین

مدیرعامل و بنیان‌گذار شرکت‌های

VA Research, SugarCRM, Geeknet



Linus Torvalds

لینوس توروالدز

مهندسان نرم‌افزار، دولوپر کرنل لینوکس



Bruce Perens

بروس پرنز

برنامه‌نویس، مدافع جنبش اپن‌سورس، او تعریف نرم‌افزار اپن‌سورس را ایجاد کرد.



داستان شکل‌گیری لینوکس^۱

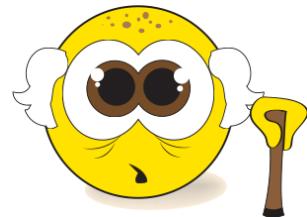


برای توضیح این که لینوکس چیه باید توضیح داد سیستم عامل چیه. و قضیه سیستم عامل اینه که آدم هیچ وقت قرار نیست اون رو ببینه چون هیچ‌کس در واقع از سیستم عامل استفاده نمی‌کنه. آدم‌ها از برنامه‌های کامپیوترشون استفاده می‌کنند. و تنها هدف زندگی سیستم عامل کمک به اجرای این برنامه‌های است. پس سیستم عامل هیچ وقت خودش کاری انجام نمی‌ده. فقط منتظر می‌شه برنامه‌ها درخواست منبع خاصی رو بکن. 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 آسون‌تر کنه.



^۱ این داستان برگرفته از دیالوگ‌های فیلم Revolution OS است.

اپن سورس راهیه که مردم بتونن روی نرم افزارها باهم همکاری کنن بدون این که درگیر اون همه مشکلات حقوق معنوی بشن. طوری که هر وقت کسی می خواد یه نرم افزار بخره مجبوره 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 بتونیم 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.



باید 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 همچ 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 نشونده هنده روحیه بازیگوشیه. که مشخصه هکرهاست. هکرها کسانی هستن که از هوشمندی بازیگوشانه لذت می برسن. خب اولین مشکلی که برآمدون پیش اومد وقتی بود که دنیای خارج بهمون فشار آورد که کلمه عبور داشته باشیم. ما روی کامپیوترهایمان کلمه عبور نداشتیم. و دلیلش این بود که هکرهاست که اون اول، سیستم



رو طراحی کرده بودن متوجه شده بودن که کلمه عبور راهی برای رئیس‌هاست که بتوان تمام کاربرها رو کنترل کنن. و نخواسته بودن قفل و کلیدی برای رئیس‌ها بسازن که بتوان اون‌ها رو کنترل کنن. بنابراین این کار رو نکردن و این مورد رو از قلم انداختن. فسلفه ما این بود که هرکس که پشت کامپیوتر نشسته باید بتونه هرکاری که دلش می‌خواهد 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 می‌گفتمن:

سلام، من یعنیم که کلمه عبور خلاص رو انتخاب کرده‌اید! چطور حموون هاریکو رو بکنید که من کردم؟ یعنی به جای کلمه عبور فقط ایسرو بزنید؟ خیلی کوتاه‌تره و تازه تایپ کردندم خیلی آسون‌تره

و البته با این پیغام من داشتم تلویحا بهشون می‌گفتمن که امنیت سیستمشون عملا در حد شوخيه ! (جاست فور فانه) در هر صورت با اين هک داشتم اون‌ها رو 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 اپن‌سورس می‌گیم، از کجا اومدن؟
این کار چطور شروع شد؟ چه کسی شروعش کرد؟



۵۹ ۴۹
سالی پین بین یعنی

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 اوایل دهه 70 یا اوایل دهه 80 بود که مردم شروع کردن به بستن نرم افزارهاشون. و گفتن: «نه اصلاً 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 مايكروسافت رو مقصراً قسمتی از اين قضيه دونست. اونها يكى از پيشگامان واقعی مدل نرم افزار خصوصی هستن.

خواهی خودمون؟
کنید خودمون؟



اواسط دهه 70 گروهی از هکرها و کامپیوترکاران سیلیکون ولی که تفندنی کار می کردند، باشگاه کامپیوترهای خانه ساز را تشکیل دادن. (انگلیسیش: Homebrew computer club). سال 76 بیل گیتس از شرکت تازه تاسیس مايكروسافت، 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 نوشه 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 تفتنی شما هدر رفته است. آیا کسی برنامه‌هایی با کیفیت برای بازار تفنن خواهد بود؟... بازخوری که ما از صدها نفر که می‌گویند از بیسیک استفاده می‌کنن گرفته‌ایم، بسیار مثبت بوده. هرچند دو نکته خارق‌العاده وجود دارد؛ یک: بیشتر این کاربران بیسیک را نخریده‌اند و دو: با مقدار حق‌التالیفی که ما از این تفتنی کاران دریافت کرده‌ایم، دستمزد کاری که صرف آلترا بیسیک شده است کمتر از ساعتی 2 دلار می‌شود. آخه چرا؟ همان‌طور که اکثریت تفتنی کاران مطلع هستند، بیشتر شما نرم‌افزارهای خود را می‌زدید 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 زدیدن نرم‌افزار نمی‌کنید جواب دادن به میتس برای مشکلاتی است که داشته‌اید. میتس از فروش نرم‌افزار خوب را می‌گیرید. چه کسی می‌تواند به رایگان کار حرفه‌ای انجام دهد؟ کدام تفتنی 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 تفتنی نکرده است. آن‌ها که آلترا بیسیک را دوباره می‌فروشند چه؟ آیا آن‌ها از نرم‌افزارهای تفتنی پول در نمی‌آورند؟ بله ولی کسانی که به ما تگزارش شده‌اند بالآخره خواهند باخت. آن‌ها کسانی هستند که تفتنی کارها را بدnam می‌کنند و باید از هر جلسه باشگاه که در آن پیدایشان شود بیرون‌نشان انداخت. من از نامه‌های هرکسی که بخواهد دینش را بدهد یا نظر و پیشنهادی داشته باشد استقبال. بیل گیتس. سهام دار اصلی مایکروسافت.



ریچارد استالمن:



62

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 (share 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 تناقض اخلاقی پیدا می‌کردند. پس این پروژه رو توی ژانویه ۸۴ شروع کردم. این وقتیه که از شغافتی 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.



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 ۸۰ که ریچارد استالمن داشت پروژه گنو رو می‌ساخت. دانشمندان کامپیوتر توی دانشگاه کالیفرنیا در برکلی در حال ساخت سیستم‌عامل آزاد خودشون



بودن. این سیستم عامل که به عنوان یونیکس برکلی یا BSD شناخته می‌شد برپایه هسته یونیکس و با اجازه AT&T ساخته می‌شد. به هر جهت، به دلیل مشکلات قانونی با AT&T و به دلیل چند پارچگی سورس کد، هکرهای دیگر کاربران غیرسازمانی به کندي BSD را می‌پذیرفتند.

ریچارد استالمن:



خب یونیکس از تعداد زیادی برنامهٔ مجزا تشکیل شده بود که با همديگه ارتباط برقرار می‌کردند. بنابراین فقط لازم بود اين برنامه‌ها رو يك‌يک جايگزين کنيم. پس شروع كردم به نوشتني. بعد كم كم افراد دیگه‌ای بهم ملحق شدن چون من اطلاع‌يه‌ای منتشر كرده بودم که از مردم دعوت می‌کرد به من ملحق بشن و توی نوشتني اين برنامه‌ها کمک کنن. تا حدود سال 91 عملاً همه برنامه‌ها رو جايگزين کرده بوديم.

این برنامه‌هایی که جایگزین کرده بودید چه برنامه‌هایی بودند؟



خب ما باید يه سیستم عامل کامل می‌داشتیم. يك کرنل لازمه، که برنامه‌ای که منابع رو به همه برنامه‌های دیگه تخصیص می‌ده، يه کامپایلر لازمه تا برنامه هارو از سورس کدی که برنامه‌نویس‌ها می‌فهمن به يك سری عدد ترجمه کنه . برای این کار برنامه‌های دیگری هم در کنار مترجم لازم‌اند. يه دیباگر، يه تکست ادیتور، تکست فورم، ایمیل و... چیزهای خیلی خیلی زیادی لازمه. سیستم عامل‌های شبیه یونیکس صدها برنامه مثل این دارند.

مایکل تیمن:



اطلاع‌يه ریچارد استالمن رو دیدم و فوريه 87 خودشو ملاقات کردم. اون برای يه دوره آموزشی پنج روزه برای برنامه Emacs به شرکت ما اوهد. در طول روز اون درباره روش‌های جدید نگاه به Emacs توضیح می‌داد. و روش‌های گسترش دادن و بهترکردن



سورس کدهای Emacs حرف می‌زد. 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 می‌بینه. ولی من خیلی مشتاق بودم و تا ماه ژوئن اون سال که اعلامش کرد و بلافضلله از اینترنت دانلودش کردم با برنامه ور رفتم و چندتا سوال ازش پرسیدم وقتی سورس کد رو براش پس فرستادم، اون با ناباوری دید که چقدر سریع تونسته بودم تکنولوژی اون رو ارتقا بدم.

وقتی از آزادی حرف می‌زنیم دقیقاً از چی حرف می‌زنیم؟!



ریچارد استالمن:



مهم ترین خاصیت گنو اینه که نرم‌افزار آزاده و آزاد بودن نرم‌افزار نه به قیمتش که به آزادی ربط داره. پس به آزادی بیان فکر کنید. آزادی ای که دارم صحبتش رو می‌کنم اینه که اگر خواستین تغییراتی در نرم‌افزار بدین یا کسی رو استخدام کنید که برآتون تغییرات رو انجام بده. اگر برای کسب وکارتون از یه نرم‌افزار استفاده می‌کنید که بتونین مجدداً نسخه‌های 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 بهره‌مند بشن. و این آزادی هاست که نرم‌افزار آزاد رو از نرم‌افزارهای غیرآزاد متمایز می‌کنه. این آزادی هاست که مردم رو قادر می‌کنه یه اجتماع (یه کامیونیتی) تشکیل بدن. شما اگر تمام این آزادی‌ها رو نداشته باشید بینتون تفرقه می‌افته و یکی بهتون مسلط می‌شه.

نرم‌افزار آزاد که می‌گین همه چیش ولنگه و بازه؟!! هیچ لاینس یا مجوزی نداره؟



اگه نرم افزارهارو در مالکیت عمومی بگذاریم یکی می تونه بیاد و یک خرده تغییر توی اون بدھ 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 می کنیم. مفهوم کپی لفت یه "کپی رایت" سر و تھه. کاری که ما می کنیم اینه که می گیم این نرم افزار کپی رایت داره و ما نویسنده اون به شما اجازه می دیم که دوباره اون رو توزیع کنید، تغییر بدید یا چیزی بهش اضافه کنید. ولی وقتی که دوباره اون رو توزیع می کنید باید دوباره تحت همین شرایط باشه. نه بیشتر نه کمتر.



تا هرکسی که اون رو از شما می گیره آزادی همکاری با بقیه مردم رو، اگه بخواهد، داشته باشه. و به این روش، نرم افزار هر جایی که بره آزادی همراهش می ره. و همکاری با بقیه مردم. و تشکیل اجتماعات به یک حق لاینفک تبدیل می شه.

این چی بود؟ این چه لاینسی بود؟



خب کپی لفت ایده کلی بود. برای استفاده از اون باید یک نمونه مشخص داشت. نمونه مشخصی که ما برای اکثر بسته های نرم افزاری گنو به کار می بردیم این است:

GNU GENERAL PUBLIC LICENSE (GPL)

(مجوز عمومی همگانی گنو)

یک سند به زبان حقوقی که این کار را انجام می‌ده. خیلی‌ها همین مجوز را به کار برند، مثلاً : 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 استالمان ناشی شده.

خیلی از شرکت‌های تجاری شروع به استفاده از این نرم‌افزارها و سیستم‌عامل گنو و بیوس 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 آدم هایی رو پیدا کنه که این کار رو برآش 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 اونها 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 کمکرسانی رو که یک مهندس داخل سازمان داره در اختیار بذاره. و بتونیم این کار رو با نصف تا یک چهارم همون هزینه انجام بدیم. خیالمون می تونه جمع باشه که مردم واقعاً اون رو میخرن. 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 شاخص‌های 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. اون‌ها پولشون رو با ارائهٔ پشتیبانی در می‌آوردن.



ریچارد استالمن:



کرنل آخرین چیزی بود که نوشتنش رو شروع کردیم و هنوز خیلی از شروعش نگذشته بود که سر و کلهٔ 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 اولیهٔ 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 اصولا به این معنیه که سیستم‌عامل یک نهاده و تقسیم ناپذیر. در حالی که در microkernel (زیرهسته، زیرکرنل) سیستم‌عامل در حقیقت چیزی نیست جز مجموعه‌ای از سرویس‌دهنده‌هایی که کارهای مختلفی انجام می‌دان و پروتکل مشترکی برای ارتباط با همیگه دارن.



خب چطور شد که پروژه گنو که این همه جلوتر بود و از قبل داشت این کار رو می‌کرد 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 اتفاقا ما طراحی‌ای رو انتخاب کردیم که طراحی بسیار پیشرفته‌ایه. از نظر قدرتی که به آدم می‌ده ولی از اون طرف معلوم شد که دیباگ‌کردنش 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 پیام‌های ناهمzman به هم می‌فرستادن. مسئله اینه که این روش برنامه‌نویسی زمینه زیادی برای باگ‌ها داشتن. باگ‌هایی که پیداکردنشون اغلب خیلی سخته چون به این بستگی داره که آیا فلان برنامه، فلان پیغام رو قبل از این می‌فرسته که اون یکی برنامه بهمون پیغام رو می‌فرسته یا بعدش؟!! و نتیجه این شد که سال‌ها طول کشید که تونستیم برنامه رو به کار بندازیم.

رابطه لینوکس با پروژه گنو چیه؟



لینوس توروالدز:



خب می‌شه گفت در چندین سطح با گنو رابطه داره. یکیش سطح فلسفیه که آدم فکر کنه بازکردن سورس ایده خوبیه.



ریچارد استالمن:



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 می‌توانن پیدا کنن تا کنار کرنل شون 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 آوردیم! ولی در واقع شانسی در کار نبود. اونها تمام تکه‌های سیستم گنو رو پیدا کرده بودن که فقط کرنل رو کم داشت. به این ترتیب وقتی همه چیز رو سر هم کردن در واقع داشتن لینوکس رو توی شکاف سیستم گنو جا می‌دادن. ولی او نا اینو نمی‌دونستن!

لینوس تورووالدز:



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 پیشرفتهای 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.

لری 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 اپن سورس و لینوکس.



ایده اسم لینوکس چی بود؟

لینوس توروالدز:



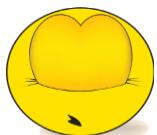
لینوکس اسمی بود که پیش خودم روش گذاشتم فقط به خاطر اسمم لینوس و × هم که حتماً باید باشه بخاطر یونیکس. اولش فکر کردم نمی‌تونم اسمش رو رسماً بذارم لینوکس چون خیلی خودخواهانه می‌شه.

ریچارد استالمن:



اونها فکر کردن یه عالمه برنامه پیدا کردن و دارن اون رو دور لینوکس می‌ذارن اینجوری شد که اسم قضیه رو گذاشتند سیستم عامل لینوکس. و این اصطلاح جا افتاد و نتیجه این شد که میلیون‌ها نفر حالا دارن از این گونه سیستم گنو به اسم سیستم عامل 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 Red Hat Linux مشکلی نداره یا لینوکس سوزه SuSe Linux یا لینوکس دبین Debian 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

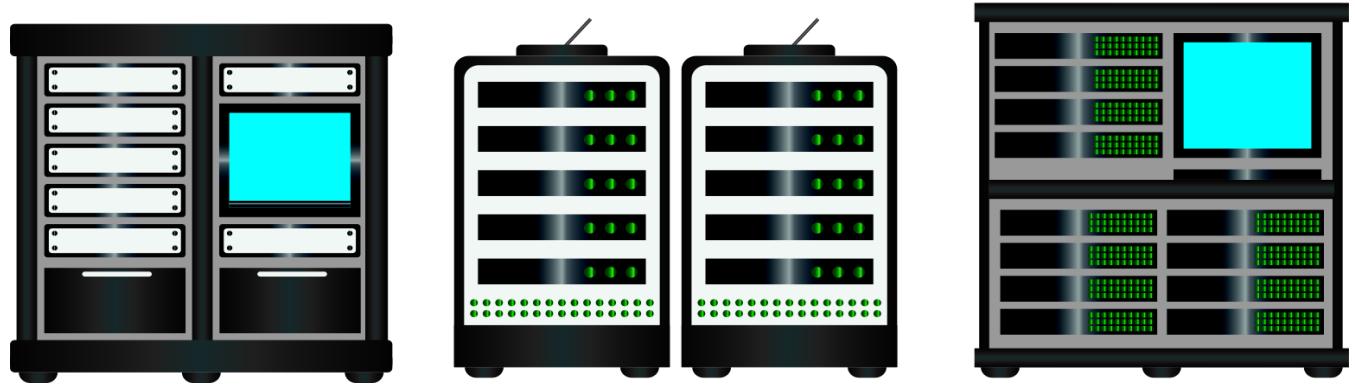
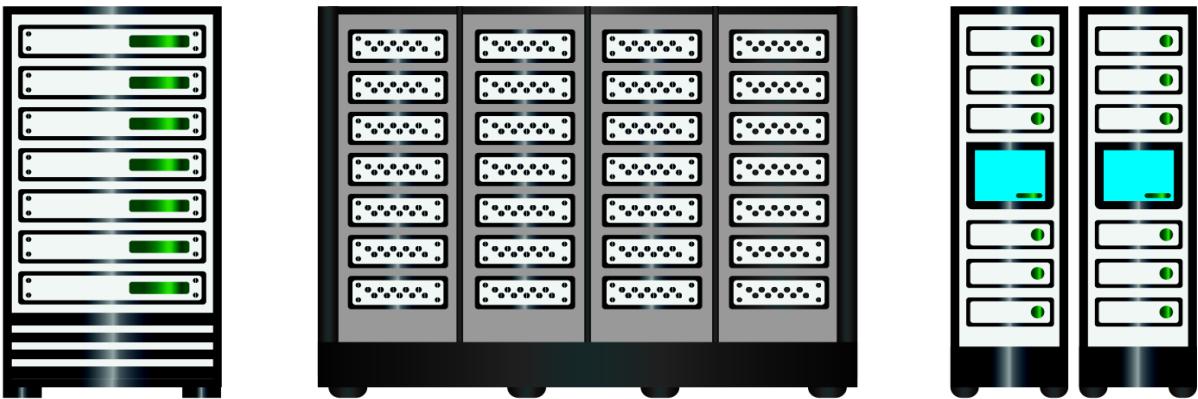


برای رشد لینوکس به ماورای دنیای برنامه‌نویسان کامپیوتر، استفاده و کاربردی نیاز بود که اون رو به یه فناوری غیرقابل اغماض بدل کنه چیزی که باعث عبور از این آستانه شد به وجود آمدن برنامه‌ای بود که ساختن وبسایتهای پیچیده رو امکان پذیر می‌کرد. این برنامه آپاچی وب‌서ور بود.



اریک ریموند:

گل سرسبد برنامه‌های لینوکس بدون شک آپاچی وب سرور بود. اگر به تاریخچه لینوکس دقیت 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 می کنم آپاچی یکی از اولین برنامه هایی بود که باعث شد مردم بگن : خب اگر لینوکس نصب کنم یه فایده مشخصی برایم داره. البته اون موقع کلی برنامه جالب روی لینوکس وجود داشت. ولی می دونید واقعا انگیزه ای وجود نداشت. می شه گفت فایده تجاری نداشت که کسی بجای NT از لینوکس استفاده کنه. تا وقتی که آپاچی اوهد و کلی چیزهای دیگه که به آپاچی وصل می شدن. می خواست بگم وقتی آدم می خواست بره و یسری سرور راه بندازه خیلی مقرون به صرفه تر بود. مقرون به صرفه تر 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 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 بدبست 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 کامیونیتی‌های مربوط به این سیستم‌عامل‌ها هنوز بیشترین مشارکت رو در تولید آپاچی داشتن. و این‌ها سیستم‌عامل‌هایی بودن که تامین کنندگان خدمات اینترنت شدیداً شروع به استفاده از اون کردن. و تامین کنندگان خدمات اینترنت خیلی از آپاچی خوششون اومد چون بهشون اجازه می‌داد کلی کارهای متفاوت بکن 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 همسنون 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 Red Hat از همه معروف‌تر است.

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



:DONNIE BRNES

Red Hat به عنوان یکی از محصول‌های مارک یووینیگ شروع شد. Marc Ewing وقتی که داشت توی IBM کار می‌کرد و توزیع لینوکسی می‌خواست که کمی بهتر باشه پس شروع کرد با لینوکس ور 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 تو ش زندگی می‌کرد. بعد یه 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 نوشتم به اسم کلیسا و بازار The Cathedral & the Bazaar و به این پرداخته بودم که اپن سورس چجوری کار می کنه. حتی او نمودع هنوز اسم "اپن سورس" رو نداشتیم و می گفتیم نرم افزار های آزاد. و گفته بودم که چرا ما تونسته بودیم نرم افزارهایی با کیفیت بی نهایت بالا رو تولید کنیم با اینکه مدام داشتیم قواعد مهندسی نرم افزار رو زیر پا می ذاشتیم. فرق بین دو روش متفاوت برنامه سازی یا develope کردن برنامه رو توضیح می دادم. یکی روش سنتی بود یا همون closed development style که من اسمش رو گذاشته بودم روش کلیسا یی. توی این روش آدم باید تک تک اهداف رو توضیح بده و گروه های پروژه کوچکی داشته باشه که با داشتن اختیارات سلسله مراتبی باهم کار می کنن و باید دوره انتشار محصولش هم طولانی باشه. از طرف دیگه اتفاقی که دیدم داره برای دنیای لینوکس می افته، روش خیلی فرد به فرد و نامت مرکز و شبیه یه بازار واقعی بود، که فواصل انتشار خیلی کوتاه بود و دائم از نظرات آدمایی که خارج از پروژه بودن بهره می برد.



بروس پرنز:

دلیل اهیمت نت اسکیپ Netscape's اینه که اونها اولین شرکت بزرگی بودن که به دنیای اپن سورس او مدن. البته ما سیگنوس رو برای ارائه پشتیبانی داشتیم ولی واقعا چندان تجاری در کار نبود. نت اسکیپ اصلا برای این اپن سورس شد که یجوری با مایکروسافت بجنگه. که داشت اینترنت اکسپلورر رو بیرون می داد. ولی اجازه نمی داد دیگران متن اون رو داشته باشن. اجازه نمی داد شرکت های دیگه همکاری کنن.





FRANK HECKER



(مهندس سابق نت اسکیپ)

من در بخش فروش کار می‌کردم. تصور خوبی داشتم که چرا مردم نرم‌افزارهای مارو 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 مروگر رو به دست بگیره و از اون برای منحرف کردن استانداردهای HTTP و HTML که وب وابسته به اون هاست 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 سرورها بیرون کنن. بازار سرورها هم جایی بود که نت اسکیپ ازش پول اصلیش رو در میاورد.



FRANK HECKER

آدم های کافی برای اینکه بتونیم نرم افزارهایمون رو 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 تو ش ارجاع دادم به مقاله کلیسا و بازار. قصدم 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 لایسننس رو چی بذاریم؟ چطور باید محصولاتمون رو 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 سرمایه گذاران رو به ما داد



FRANK HECKER

یه نسخه از مقاله رو دادم به مارک اندریسن Mark Andreessen یکی از بنیان گذاران نت اسکیپ اون زمان بود. اونم 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 کنن! 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 می گفتیم free مردم فکر می کردن نمی تونن ازش پول در بیان یا اونو بفروشن که تصور اشتباهیه. می خواستیم 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 کامیونیتی ها آزاد باشن و این برای کیفیت زندگی همه مهمه. و اومدن گفتن که اپن سورس 9تا حق مشخص رو به شما می ده:

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. می شه 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 Mac استفاده کنن باید بتونن نرم افزار رو تبدیل کنن.



3- کار اشتقاچی 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 زمینه‌های 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 دیگه‌ای 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 ردhet توزیع کرده باشم لاینس نمی تونه بگه که نمی شه این رو روی سیستم‌های Debian یا SuSE توزیع کرد.

9- لاینس نمی تونه نرم افزارهای دیگه رو محدود کنه



اگر من این رو روی یک CD با نرم افزار دیگه‌ای منتشر کنم، لایسنس نمی‌تونه بگه اون برنامه دیگه هم باید آزاد باشه و گرنه نمی‌شه برنامه من رو توزیع کنین

اریک ریموند:



یه اتفاق دیگه این بود که پایگاه‌های داده به این جمع اضافه شدن! اواخر ژوئیه و اوایل سپتامبر ۱۹۹۵ او را کل اعلام کرد برنامه‌هاشو به لینوکس چمل می‌کنه. 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 می‌شدند. بالاخره توجه سرمایه‌گذارها جلب شد. عکس لینوکس روی مجله فورچون چاپ شد و این یعنی توی اپن سورس داشت اتفاقاً می‌افتد. دیگه سرمایه‌گذارا نمی‌تونستن نادیده بگیرن و از دست من که مدام بهشون سرزده بودم خسته شده بودن. آخرش به این نتیجه رسیدن که سرمایه‌گذاری کنن.



اریک ریموند:



مايكروسافت داره از لینوکس برای دفاع از خودش استفاده می‌کنه. اون‌ها از لینوکس برای اثبات این ادعا استفاده کردند که اون‌ها انحصار رو در دست ندارن چون اصولاً لینوکس می‌تونه هر لحظه اون‌ها رو از موضع برتری که دارن پایین بکشه. بحث خیلی جالبی بود. یه ادعای کاملاً غلط انداز، چون به هیچ وجه پاسخی به این اتهامات نداد که اون‌ها قبل مشغول قلدر بازی و دیگر کارهای غیرقابلی 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 می‌خواست 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 استالمن:



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، امدمون؟؟؟



چی شد که شما پنگوئن‌ها شدیدن نماد لینوکس؛ مثلاً این مینیون‌ها چشون بوده‌یا فیل‌شاه که کارتونشم داریم. اصلاً یه سوال کلی تدریجی هر چی جک و چونوره شده نماد ریان برنامه‌نویسی؛ چرا یه هیوالارو نمی‌ذاری؛ ما په این گوگولی مگولی!!!!



(یییییییییش. نخیدم. یه روز لینوس مارو توی
باغ وحش دید و عاشقون شد و یه نوع پنگوئن به اسم
رو که چاق و خوشحاله رو گذاشت روی لینوکسش



Oh my GOD

چه خفن، خیلی تعجب کردم تو قیافم معلومه؟

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

پسر امروز چند شنبه اس؟



هشت شنبه است دیگه. چطوره؟

عارضم خدمتتون که ما توی هشت شنبه ها مخ

نمی زنیم



او نوچت بر چه اساس؟

پر اساس اصل خودم. اصل خرها.

دلیل از این پالاتر؟



نگاه کمی فنی تر به سیستم عامل

همان‌طور که لینوکس توروالدز در ابتدای داستان اشاره کرد، لینوکس یک سیستم‌عامل کامپیوتری است. یک سیستم‌عامل شامل نرم‌افزارهایی است که کامپیوتر شما را مدیریت می‌کند و اجازه می‌دهد برنامه‌های خود را روی آن اجرا کنید. و چندین ویژگی دارد که به آن اشاره می‌شود:

شناسایی و تنظیم سخت‌افزارها: وقتی که لینوکس بوت می‌شود (یعنی وقتی کامپیوترون را روشن می‌کنیم)، به اجزاء سیستم شما نگاهی می‌اندازد (مثل CPU، کارت شبکه و...) و بعد نرم‌افزارها (دراایورها و ماژول‌ها) و مواردی که نیاز به دسترسی به این سخت‌افزارهای خاص را دارند لود می‌کند.

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

مدیریت پروسه‌ها یا فرآیندها: سیستم‌عامل 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 سیستم‌عامل باید راه‌های برای شروع، متوقف کردن، و تغییر وضعیت فرآیندها و پروسه‌ها را ارائه دهد.

مدیریت حافظه: رم و بخش swap (جلوتر می‌گیم چیه) باید به برنامه‌هایی که نیاز به حافظه دارند، داده شود. سیستم‌عامل این تصمیم را می‌گیرد که چگونه به حافظه یا همان مموری درخواست بدهد و آن را مدیریت کند.

ارائه‌دادن یک رابط کاربری: یک سیستم‌عامل باید راهی برای دسترسی به سیستم را برای یوزر ایجاد کند. سیستم‌هایی که از لینوکس استفاده می‌کنند، چیزی دارند به اسم shell . در واقع برای کار با سیستم لینوکس، ما درون shell برنامه یا دستورات را می‌نویسیم و اجرا می‌کنیم. اشتباه نکنید! برای اینکه از لینوکس استفاده کنید لزوماً به این معنی نیست که باید از shell استفاده کنید. این روزها با آمدن توزیع‌های مختلفی، مثل ubuntu، رابط



های گرافیکی دارند که می‌توان بدون داشتن دانش استفاده از shell و دستورات یا به اصطلاح کامنلاین‌ها، از لینوکس استفاده کرد. درست مثل چیزی که در ویندوز شاهد آنیم! 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 کنند. این 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 پروسه‌های خودشان را انجام دهند و کنترل کنند.

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 پروسه‌های 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 کامپیوترتان را فراهم کنند.

راه‌اندازی سرویس‌ها: برای استفاده از چاپگرهای رسانیدگی 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 لینوکسی 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 ترجمه کردن اصطلاحاتی که مال ما نیستند خوشم نمی‌آید!), specialized storage (ذخیره‌سازی تخصصی) بیشتر در شرکت‌ها استفاده می‌شود.



چه فرقی پین لینوکس و سیستم‌عامل‌های دیگه هست عایا؟

اگر تازه پا به دنیای لینوکس گذاشتید، حتماً 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 نرم افزارهای خودتان را به سیستم عامل وصل کنید تنها به این دلیل که سازندگان سیستم عامل شما می خواهد این انحصار را حفظ کند!

شاید پیش خودتان بگویید چه اهمیتی داره؟ من که نمی خوام دولوپر نرم افزار بشم. نمی خواهم ببینم که چطوری میشه سیستم عاملم رو تغییر بدم.

شاید حق با شما باشد. اما حقیقت این است که بسیاری از افراد و شرکت های دیگر که پیشگام خلاقیت و نوآوری هستند از این امکانات و نرم افزارهای اپن سورس و بعضا رایگان استفاده می کنند، 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 می کند به کلی از کارهایی که نباید بکنیم و می کنیم.

در بخشی از بندهای آن اشاره شده: لاینس این نرم افزار به شما فروخته شده است نه خود نرم افزار. یا مثلًا در جایی دیگر اشاره شده است که فقط می توانید خودتان از این نرم افزار استفاده کنید آن هم تنها بر روی یک کامپیوتر، نباید به کسی آن را بدهید و ...

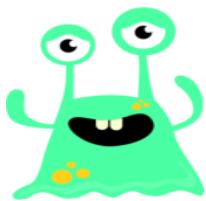
اگر جزء اقیلیتی هستید که لاینس را پرداخت کردید، مطمئنا هم در این دنیا و هم در آن دنیا رستگارید! (فقط دست مارم سر پل بگیرید!)



خوب که چی؟ می خواهی پگی 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.



توزیع لینوکس چیه؟

اگر داستان را خوانده باشید، فهمیدید که لینوکس مستقیماً اشاره به کرنل 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 (distro)



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

شرکت‌های مطرحی همچون Red Hat و Debian 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 چی؟ راهی نداره؛ ما می‌میریم
می‌میرم

چرا، بسیاری از شرکت‌ها همین distroهایی که وجود داشت، مثل Debian (دبیان)، کمی تغییر دادند و distro خودشان را ارائه کردند. (اینم یکی از مزیت‌های دنیای اپن سورس‌ه که می‌توانی هر چیزی را مطابق نیازت تغییرش بدی) در واقع با اینکار بعضی از distroهای محبوب، پایه شکل‌گیری و تولد distroهای دیگر شدند. مثلا 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 ملئ پول و درآمدشون رو از کجا در میارن؟!

این جاست که باید تفاوت بین دو واژه free مطرح شود. همان‌طور 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.

Debian تا Red Hat از



خوب این اسمای کچ و کوله چیه؟

من می‌دونم! اسم دوئا شد که یا پالاخره چاییه که از اپن‌سورس و دنیای لینوکس شروع کردن پول دراوردن و خدمات دادن



در اواخر سال 1990 به چند دلیل محبوب شد:

یکی استفاده از package management های RPM بود.

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

دلیل دیگر از 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 می توانست تاریخ و زمان، و کارهای پایه و اولیه خودش را انجام دهد.

سال های زیادی بود که Red Hat distro، محبوبی برای هر دو قشر شد. هم برای کسانی که حرفه ای کار می کردند و هم برای کسانی که علاقمند به دنیای لینوکس بودند. به تدریج که دنیای لینوکس گسترده می شد و افراد بیشتری وارد آن می شدند 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 سیستم عامل را توسعه داد: Fedora و Red Hat Enterprise.

سال 2012 این شرکت اولین شرکت اپن سورسی بود که به درآمد سالانه بیش از 1 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، distro های دیگر تمرکز خودشان را روی دستکتاب ها (کامپیوترهای شخصی) یا مشاغل کوچک گذاشته بودند. اما Red Hat بیشتر روی ویژگی ها و برنامه هایی کار می کرد که برای کسب و کارها و دولت نیاز بود.



Fedora

با اینکه Red Hat RHEL را عرضه کرده بود، اما Fedoradistro 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 کامپیونیتی دوستداران لینوکس نگه دارد.

Fedora بیش از 16 هزار پکیج نرم‌افزاری را دارد. بسیاری از این‌ها با آخرین تکنولوژی اپن‌سورس 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 نرم‌افزارها یا انجام یکسری کارها، باید کارهای اضافی‌تری هم برای به راه‌افتدان کارتون انجام بدید.

با این حال نباید به آسانی از این توزیع گذشت، Red Hat بسیاری از اپلیکیشن‌های جدید را در Fedora تست می‌کند. با کارکردن با این disro می‌توانید مهارت‌هایی که برای کار با بعضی ویژگی‌های Red Hat Enterprise Linux هست را بدست آورید.

با اینکه Fedora محبوب است اما در سال‌های اخیر Ubuntu توانسته بیش از Fedora محبوب شود.

Ubuntu بر پایه distro محبوب Debian ساخته شده است. و نسبت به Fedora از ثبات بیشتری برخوردار است. بسیاری از distroهای دیگر بر پایه Debian ساخته شده، که می‌توانید در سایت distrowatch.com پیداکردن کنید. بیش از 130 توزیع فعال وجود دارند که برپایه Debian هستند. بعضی از distroهای محبوب برپایه Debian این‌ها هستند: elementary OS، Linux Mint (من برای اولین بار که وارد دنیای لینوکس شدم از این توزیع استفاده کردم!), Zorin Os، Kali Linux، LXLE، Zorin Os (بیشتر برای امنیت بکار می‌رده) و بسیاری از توزیع‌های دیگر. ولی محبوبیت بیشتر Debian بخاطر پیداکش Ubuntu (ubuntu) بود.



نصب اين distro بسيار ساده است. از محيط گرافيكى خوبی استفاده می‌کند و برای نصب، برخی نرم‌افزارها را پیشنهاد می‌دهد. (يه چيزی مثل بازار برای اندورید يا play store داره که نرم‌افزارهایی رو برای دانلود گذاشته)

حتی می‌توانيد ubuntu را روی CD یا فلاش به صورت لایو load کنید و از آن استفاده کنید. لینوکس لایو به لینوکس‌هایی گفته می‌شود که کلا از روی سی‌دی می‌تواند بوت شود و همه کارهایتان را می‌توانید انجام دهید. و اگر هم بخواهید چیزی را save کنید به یک storage دیگه نیاز دارید. بیشتر به این درد می‌خورد که اگر یک سخت‌افزاری دارید که دچار مشکل شده است اینطوری تست کنید، یا می‌خواهید به هر دلیلی سیستم‌عامل خودتان را جابه‌جا کنید.



پرای شروع چی پذیریم؟



یعنی چی؟ مخانواده (ینچه) نشسته‌ها! الان قهر می‌کنه می‌رده دیگه
ادامه‌شو نمی‌گه. 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 (مایکروسافت) تولید می‌شود اما لینوکس اینگونه نیست. و مردم روی بخش‌های مختلف این سیستم‌عامل 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 (کامپیوتراي شخصی) استفاده کنیم

و گاهی روی سرور. مثلا ممکن است آن‌هایی که برای سرور هستند رابط گرافیکی نداشته باشند.

در واقع 70 درصد توزیع‌ها مشابه یکدیگرند و در بعضی بخش‌ها مثل مدیریت بسته‌ها پیکیج منیجمت فرق دارند. همان‌طور که ریچارد استالمن اشاره کرده، سیستم‌عامل‌های یونیکسی (مثل گنو/لینوکس) چیز جز بسته‌ها نیستند. حالا فرق بعضی از distro‌ها در مدیریت این بسته‌هاست. مثلا RPM از Red Hat استفاده می‌کند ولی distro‌هایی که مبتنی بر debian (دبيان) مثلا ubuntu، از مدیریت بسته‌ها 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 نخواهید داشت.

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



پیا فک و فامیلتتو چمع کن پید آبرومون رو پردن ای پاپا. پلد
نیست سوالشم درست پنویسه. مصلا آخه یا مثلا؟

شما به گندگی خودت پیختش. 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 مشخصاً و یکراست! می‌خواهید آزمون‌های Lpic را شرکت کنید، بسیاری از افراد در سطح اینترنت به رایگان در حال آموزش این آزمون هستند. مثلًا جادی‌عزیز. روند پیشرفت این کتاب به سمت دست و پنجه نرم‌کردن با لینوکس! و همینطور استفاده از آن در دنیا و ... خواهد رفت. با این حال برخی مطالب مربوط به آزمون‌ها گفته خواهد شد.

(این‌و گفتم برای کسایی که صاف میخوان برن فقط آزمون lpic را 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.)

حوالدم سر رفت. می‌گی اصل کاری رو یا نه؟



نصب لینوکس

چون distro‌های مختلف شرکت‌های مختلف ساخته شده‌اند، پس مراحل نصب هم ممکن است تفاوت داشته باشد. (مسئله شاید خیلی جدی نباشه! هر distro‌ای 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 پارتبیشن‌بندی و دیگری انتخاب فایل سیستمی است که می‌خواهید انتخاب کنید. والبته بحث نرم‌افزارهایی که می‌خواهید داشته باشید. (یا همون نصب پکیج‌ها)

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

2- می‌توانید از دو سیستم عامل روی یک کامپیوتر بهره ببرید: کلاً بعد از بوت‌کردن سیستم شروع به نصب‌اش 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 کامن‌لاین آن را بررسی کنید.



لطف می‌کنی پیش عذا رو پگی؟ موارد لازم چهه طرز تعییه‌اش، انگار
کلاس آموزش لینوکس پا مانع هستیم



اگر می خواهید روی virtual box یا هر ماشین مجازی دیگری نصب کنید، اول آن را دانلود کنید.

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



(ها من پیدا کردم یه فایله ISO اومد پرای دانلود



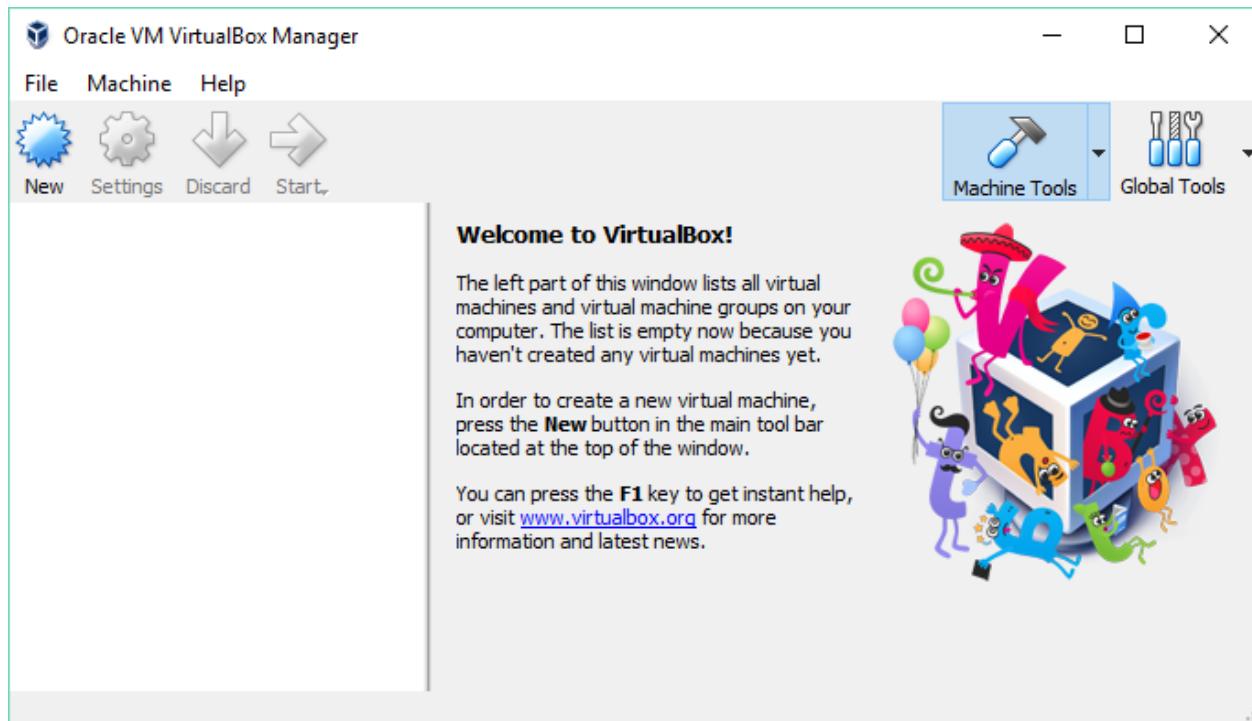
پاشه تو مخوبی عزیزم

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، تاچ سرم،

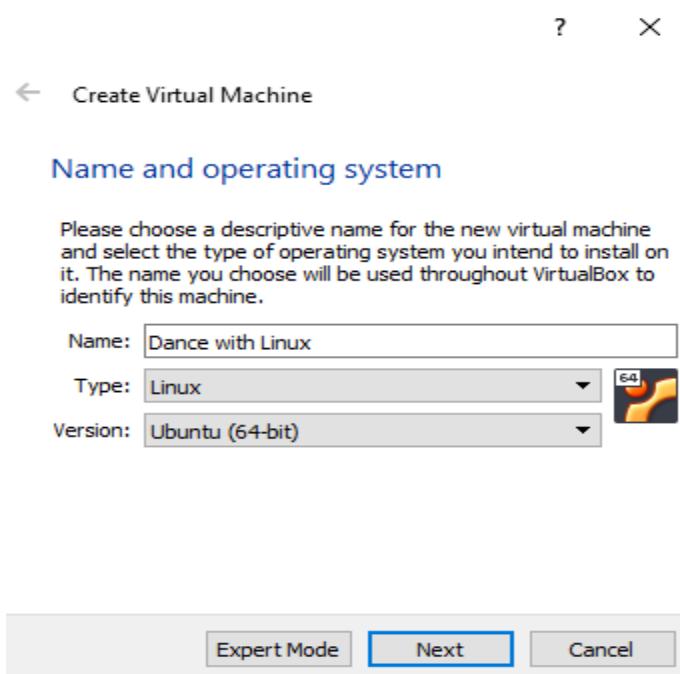
فکر ت نمیره از سرم....



پسه گرمون کردی. نهون

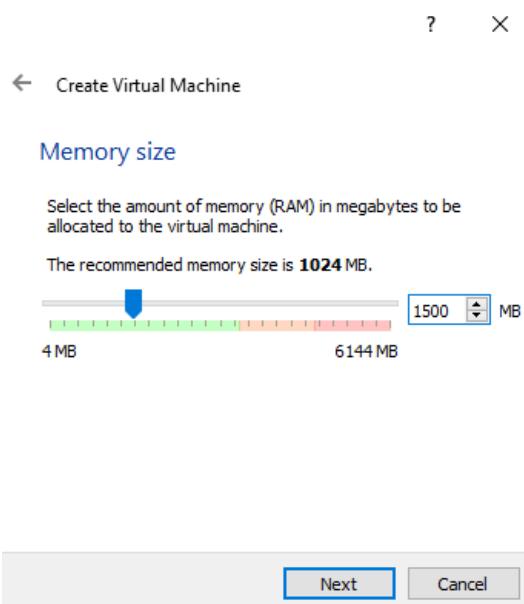


در قسمت linux، type، version مطابق چیزی که دارد

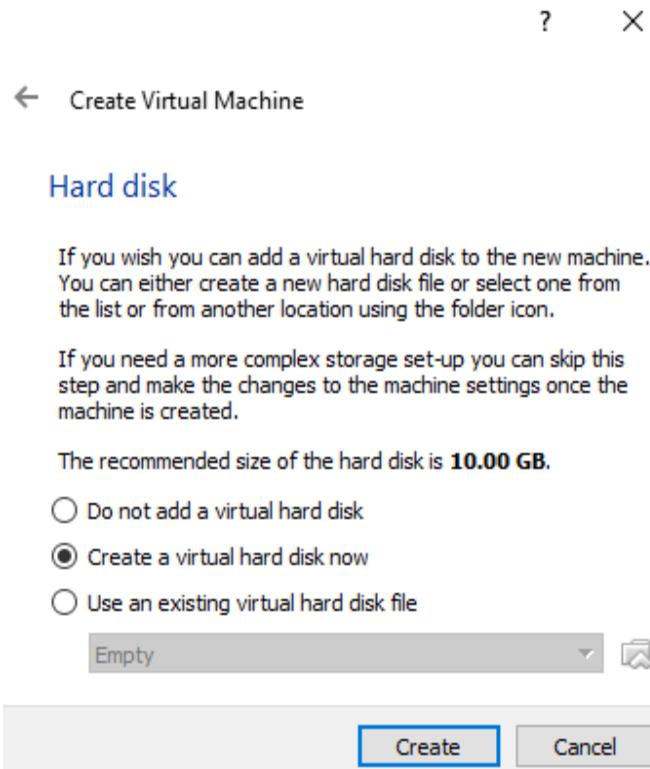


اگر هم distro ما در گزینه‌ها نبود، در گزینه‌های ابتدایی لیست، آخرین کرنل را انتخاب کنید.

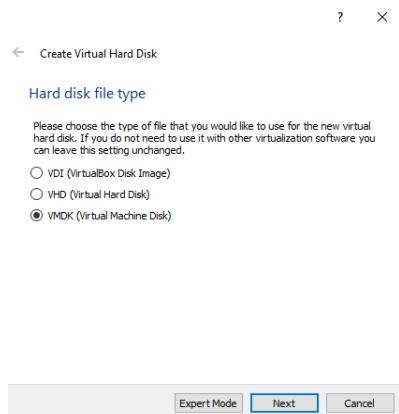
در قسمت Memory size میزان RAM که می‌خواهید را انتخاب کنید.



در قسمت Hard disk روی create a virtual hard disk now گزینه create 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 بگذارید.

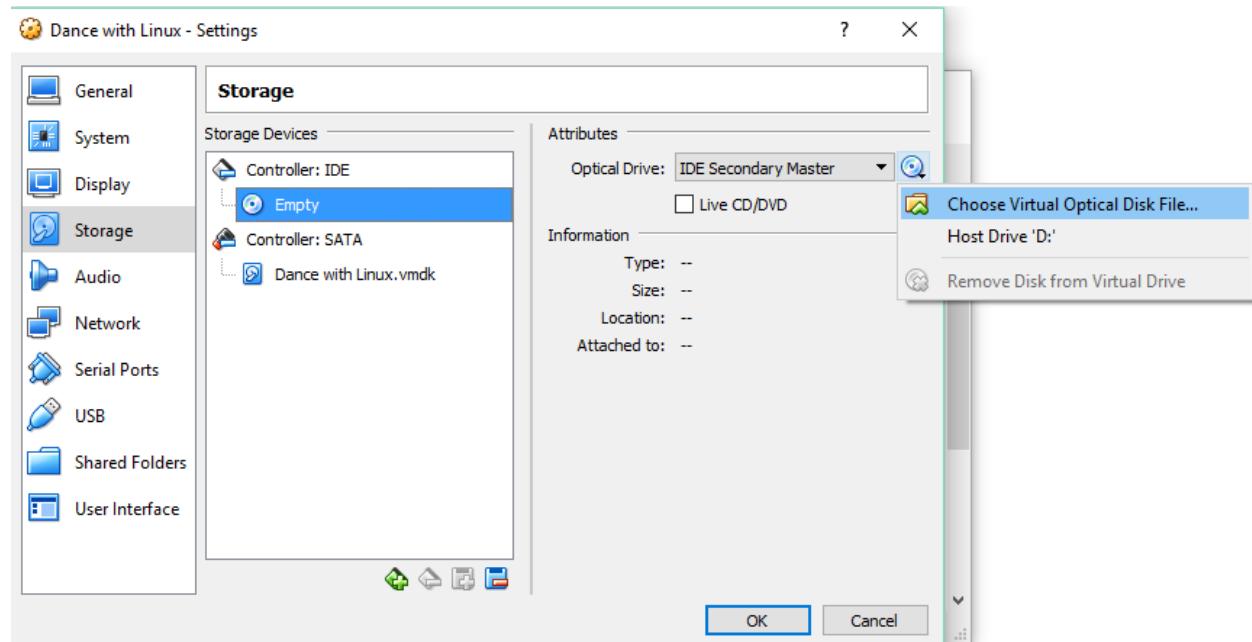
در قسمت بعدی یک اسم برای hard disk بگذارید. (یا همان اسم قبلی را بگذارید).



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

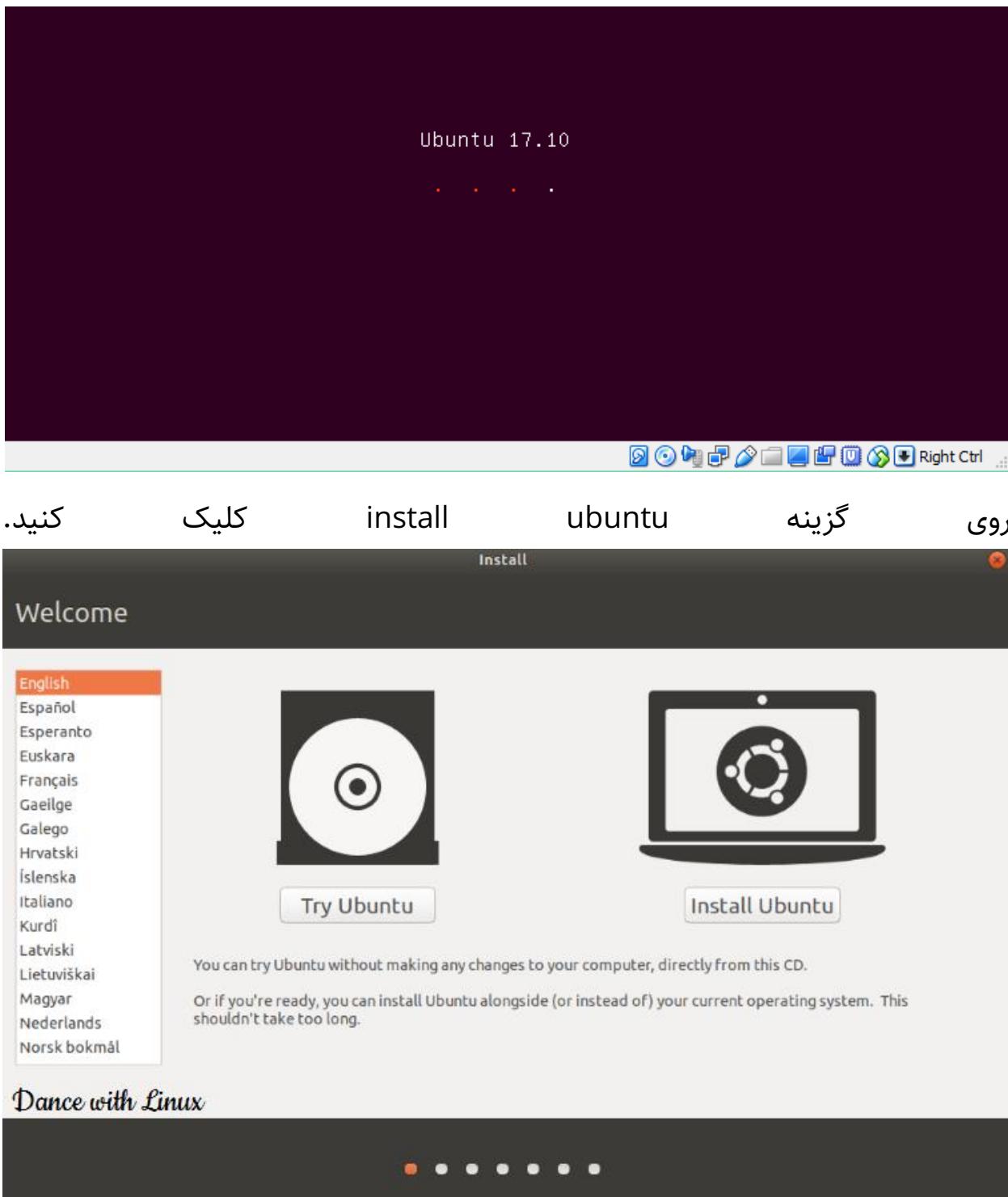
روی setting کلیک کنید. (بالا دومین آیکن)

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.



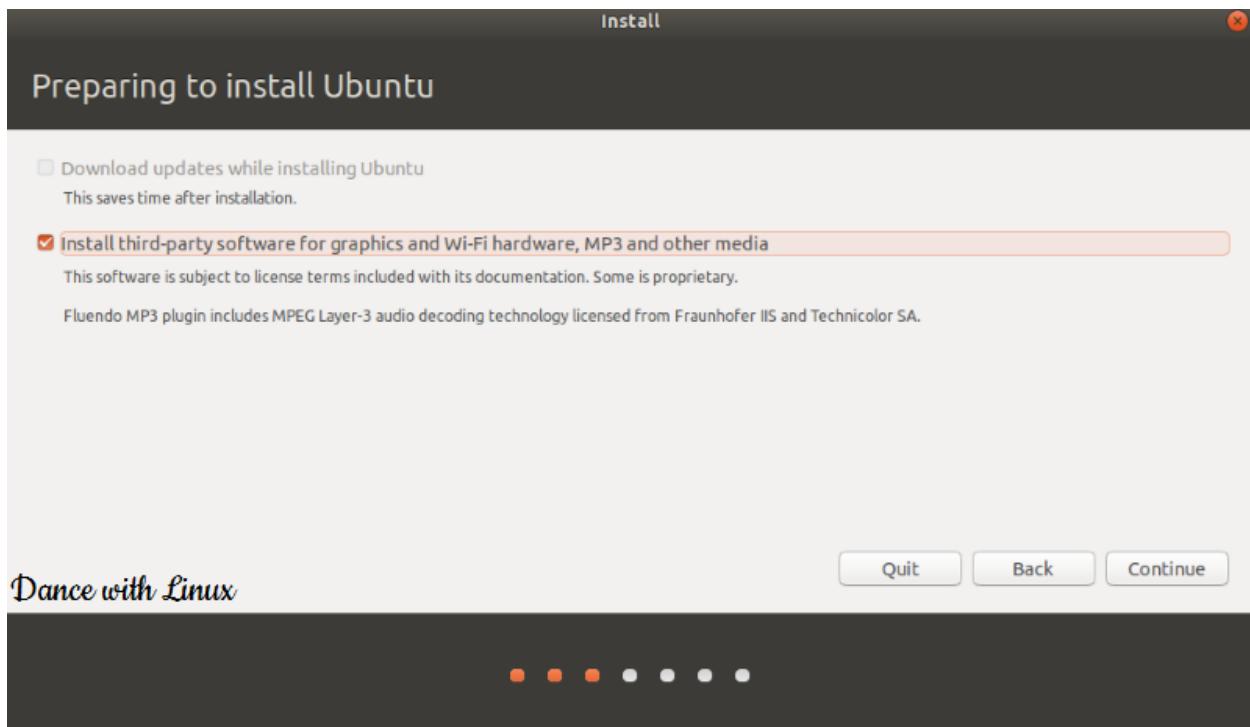
بعد روی continue و در آخر روی start بزنید.





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.



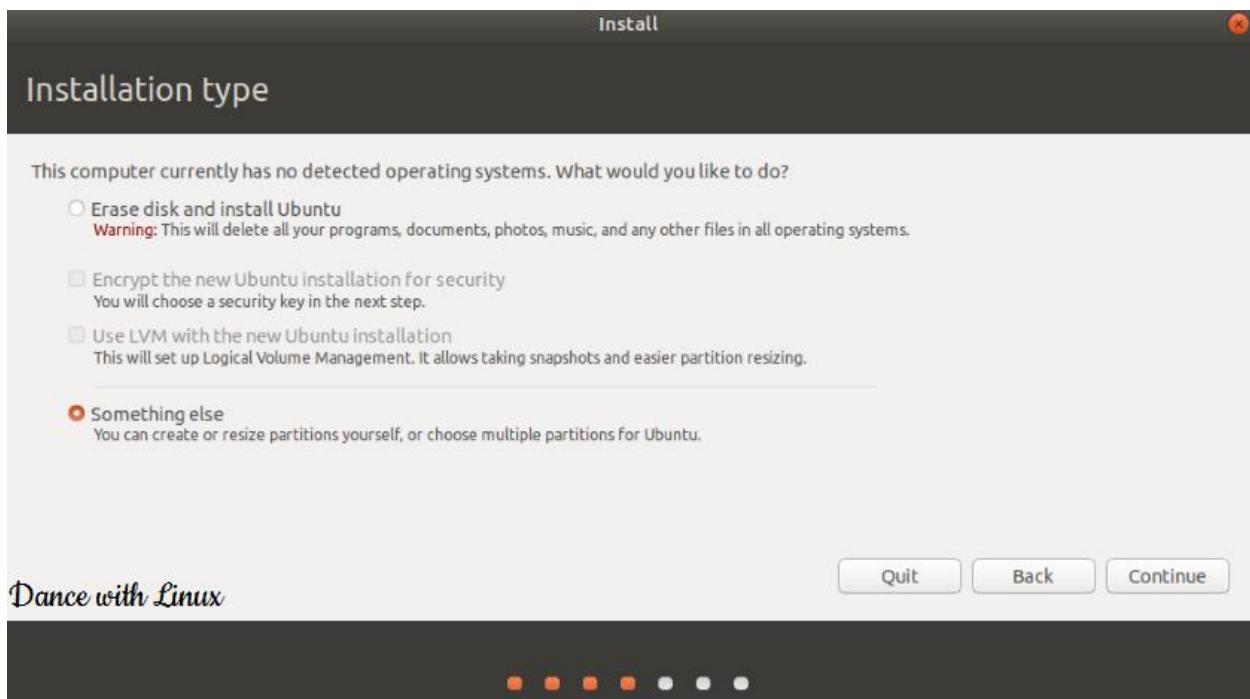


اینجا بخشی است که انتخاب می‌کنیم از چه پارتیشنبندی‌ای می‌خواهیم استفاده کنیم و چه فایل سیستمی مد نظر ماست.

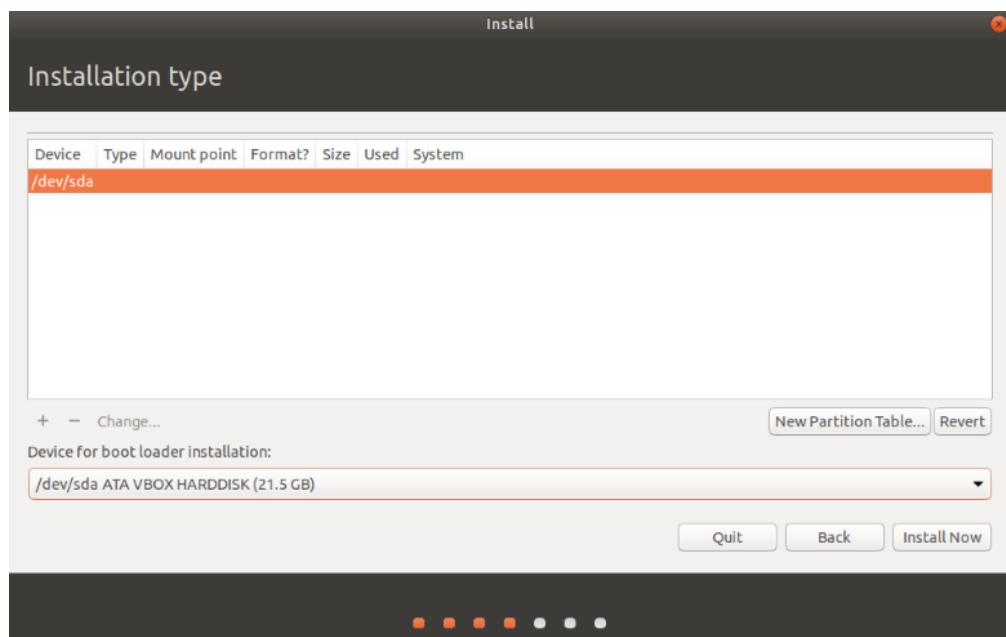
در گزینه اول، که دیفالت نصب هم است، به طور کامل هارد دیسک شما را پاک می‌کند و پارتیشن‌های مورد نیاز را نصب می‌کند. این گزینه بیشتر به درد کسانی می‌خورد که تازه وارد فضای لینوکس شده‌اند. با زدن این گزینه همه کار را خود نصب‌کننده انجام می‌دهد. منتهی برای توضیح برخی نکات ما گزینه `sth else` را زدیم.

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



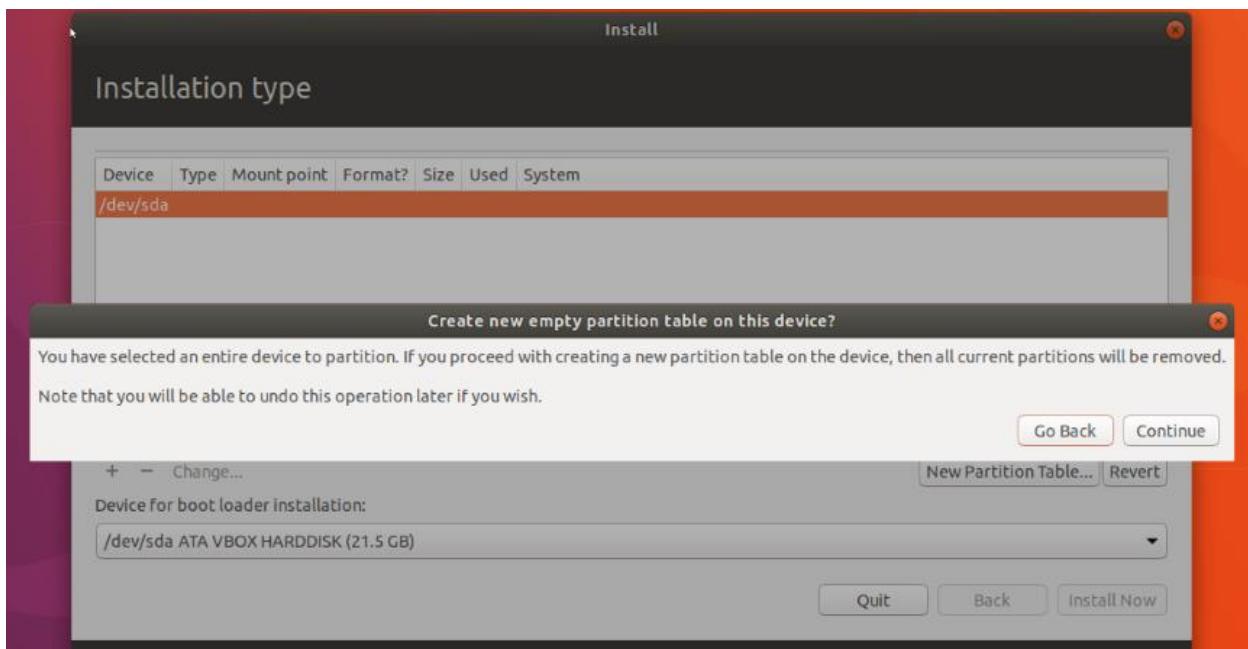


قسمت بعدی مربوط به لیست هارد دیسک‌های شماست. همه چیز در لینوکس به صورت فایل به شما ارائه می‌شود. dev در حقیقت فولدری است که سخت‌افزارها را به صورت فایل نشان می‌دهد و /sda هم مربوط به اسم هارد دیسک شماست.



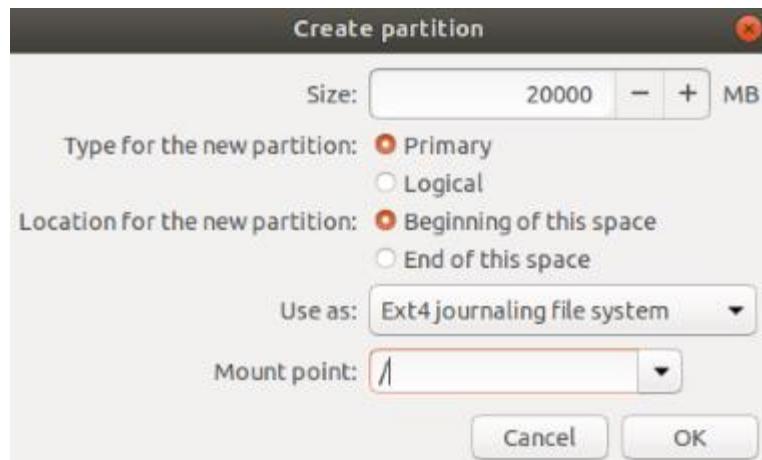
این هارد دیسک الان پارتیشن ندارد. روی new partition بزنید و بعد روی کانتینیو.





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 گزینه بگزینید. 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 مرحلهٔ بعد بروید.



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



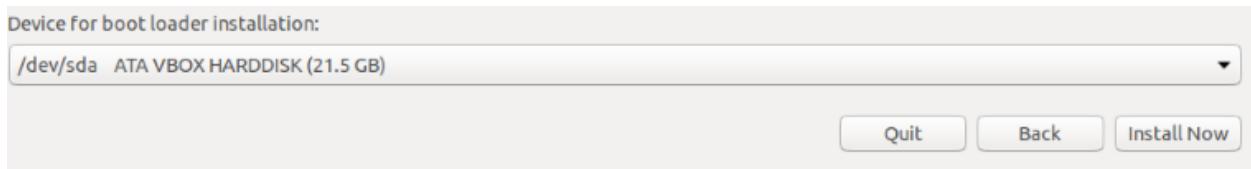
Swap پچیہ؟ این دو میں پارہ دارہ می کیا (اگر) فحشہ پکو صدا پر نہم ہیو لاہائی (پچھے) پلا پیاں (اگر)

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 بعده 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 area بزنید، و در قسمت use as گزینه swap را بزنید.

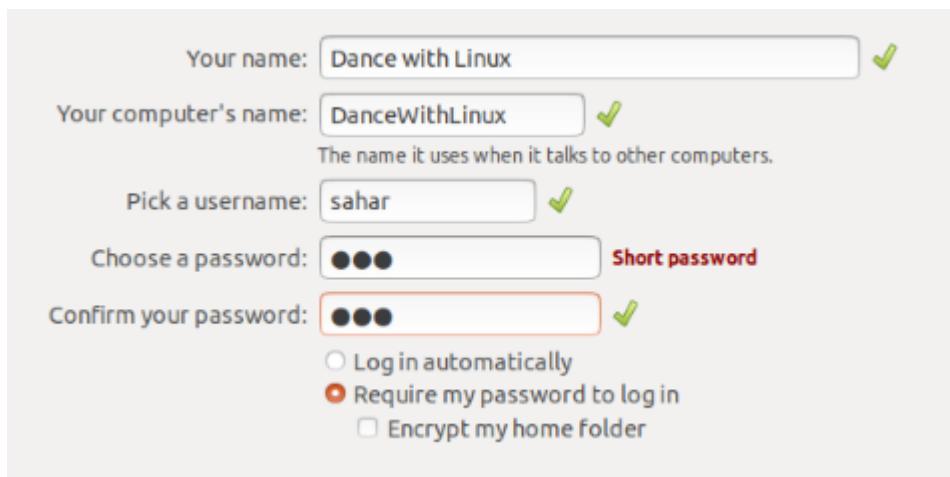


در قسمت بعدی باید انتخاب کنیم که boot loader روی کدام پارتیشن نصب شود. در واقع boot loader لینوکس را load می‌کند. گزینه را روی هارد دیسک بگذارید. . (دلیل فنی: نیاز دارید boot loader روی سکتور صفر قرار بگیرد.)

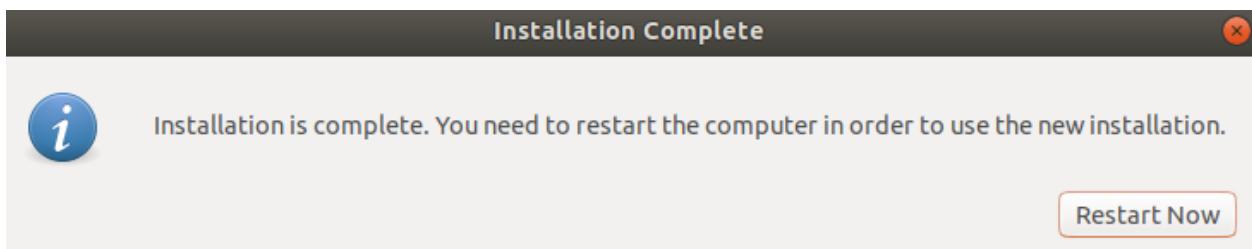


در آخر روی گزینه `install` و بعد `continue` بزنید.
کشور را انتخاب کنید.
توصیه: کیبور را روی انگلیسی بگذارید. بعدا می‌توانید کیبور فارسی را اضافه کنید.
حالا باید یک یوزر بسازید.
یک پسورد انتخاب کنید.

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 وارد کردن پسورد را می‌خواهد.



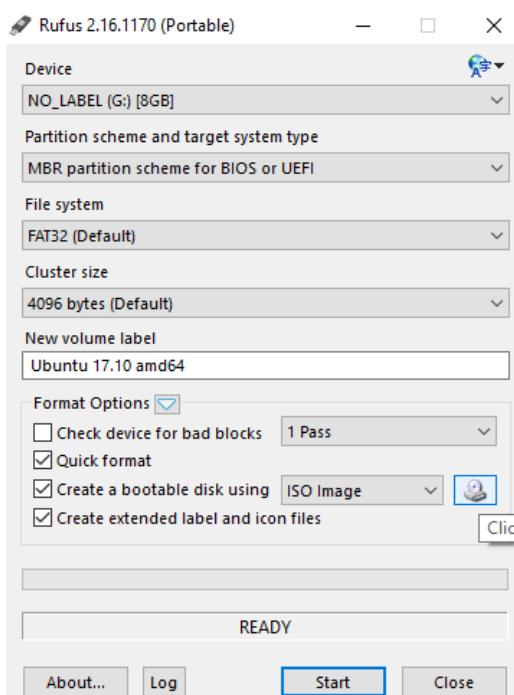
در انتهای هم پس از نصب پیغام restart سیستم را می بینید.



نصب لینوکس روی سیستم

برای نصب لینوکس روی سیستم، بیش از هر چیز، نیاز دارید یک فلش یا DVD یا IDVD را بوتیبل کنید. برای این کار بسیاری از نرم افزارها وجود دارند. مثلماً ما از rufus استفاده کردیم.

بعد از دانلود نرم افزار آن را باز کنید.



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



عین خیاپانی حرف می زنی! انتخاب رو انتخاب می کنن 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 Image start. در انتهای فایل داده هشدار می‌دهد که تمام فایل‌ها فلش شما پاک خواهد شد. دو تا اوکی بدین و تموم.

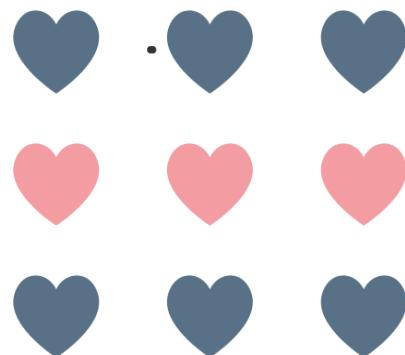
بعد سیستم را ریستارت کنید و فلش را بوت کنید. (یعنی موقعی که صفحه مشکی اومد جلوتون یکی از کلیدارو بزنید می‌رہ و فلشتون رو بوت می‌کنه. اگر نشد، یعنی در قسمت تنظیمات بوت شما، گزینه این که اول فلش را بوت کنه فعال نیست، اونو فعالش کنین). بعد از بوت کردن فلش، مراحل نصب به ترتیبی است که در تصاویر بالا، از قسمت install آمده است.



فصل دوم: قلب لینوکس



معرفی کرnel
کار با shell



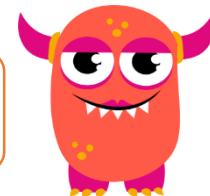
شروع کار با دسکتاپ

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.



شنبیدی که چی گفت؟ گفت دسکتاپ شخصی‌ترين
بخشن کامپیوترا.

مثل مسواج



دلپندم مسواك



پس من عکسای شخصی آپنده خودم و لینوکس چان و
پچه‌هایمان رو می‌دارم روی دسکتاپ. رسته‌ای این شکلی
هم می‌گیرم



خری دیگه خر



پس چی. من افتخار می‌کنم به خردمنم. اصلا هر کی
بعز من پکه خدم، پاهاش قانونی مطابق قانون خودم
بدخورد 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 خونتون می‌رسی دقیقاً په چې دسترسی پیدا می‌کنی؟



من پګم؛ پستېگی داره.



په چې؟



په اینکه خونه پاشم و په خونه لاګین کنم یا از پیرون پیام خونه



مګه می شه خونه پاشی و په خونه لاګین کنی؟
خونه‌ای لاګینی دیگه. مارم لنگه خودت مخ کردی؟



هدر چې با من بحث نکن. فرق داره.



چه فرقی؟



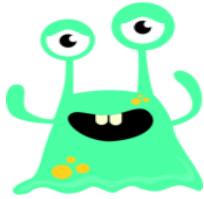
وقتی از پیدون میام خونه یا همین لایکین می کنم او لین
چیزی که پهش دسترسی دارم سرویس پهداشتیه.

ولی وقتی خونه پاشم و لایکین کنم په خونه، او لین چیز
میتونه حیلی چیزرا پاشه. بحث فلسفیه. از درک تو خارجه.
پاید توی موقعیتش پاشی تا بفهمی. اصلا پیا پدیم
خونمون پهنه پکم.



الان که فکر می کنم می بینم فلسفت حیلی سنگینه. همینجا
می مونم ولی همین که چای دستشویی گفتی سرویس
پهداشتی چای شکرش باقیه. این فلسفت نگه دار به لینوکس
پکو و پیرتش خونت نه من. شعور می خود که از تو کلا خارجه.





پهلوان پیچه رو بدہ

چشم تو فقط عصبی نباش. وقتی لاگین می کنی تقدیبا همین وضعیته که خرده گفت. یعنی وقتی لاگین می کنی می تونی په shell و دسکتاپ مختص په خودت و کلی اجازه 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 اکردن سه کار اصلی را انجام می دهد:

- شما را به عنوان یک کاربر خاص شناسایی کرده
- Shell یا پوسته و دسکتاپ مختص به خودتان را ارائه می دهد
- 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 دارند.

می‌خواهم دسترسی رو از سطح
دسترسی پیش خارج کنم یا فلسفة پیشنهادی





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



هرچی! اختیار خونه مُودمو که دارم



خونت نمیام

پس عروسیم نیا، چون اونجا دیگه حیلی چیز ارو از سطح
دسترسیت خارج می کنم. مثلًا اون دختره... آره همون، پاشدک
آشناش می کنم... یعوری از سطح دسترسی خارچت می کنم که
همش پوچ اشغال پذنی



چرا کلا بحث لاگین را مطرح کردیم؟ چون در لینوکس می توان 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 می شود).

کاربر عادی: شخصی است که از سیستم استفاده می کند. یک نام کاربری منحصر به فرد
و کلمه عبور مختص به خودش را دارد. یک سیستم می تواند 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 لاگین کنید باید خیلی مواظب باشید چون اشتباها در این ناحیه خدمات جدی دارد. به کاربر root اصطلاحاً 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 پاک کردن یوزرهای دیگر را نیز دارد.



چرا پايد پيئمون فرق وجود داشته باشه؟ اين فاصله طبقاتي توی لینوکس چيه که راه اندامختين؟ لا بد فردا حسن نژاد پرستي تون گل مي کنه و ما رو هم پيرون مي کنيد. اصلا اينو چواي پده؟ از يه کامپيوتر که فقط يه نفر استفاده مي کنه چرا اکانت‌ها مختلف پايد وجود داشته باشه؟

برای اینکه کاربر عادی که هر روز از لینوکس استفاده می‌کند اشتباها 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 احتمالاً بهترین دلیل برای بکارگیری یک سیستم لینوکسی می‌باشد. این داشتن نام کاربری یا اکانت مجزاً 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 یه
گشتی پذیریم؟

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



قدم زدن در فضای دسکتاب

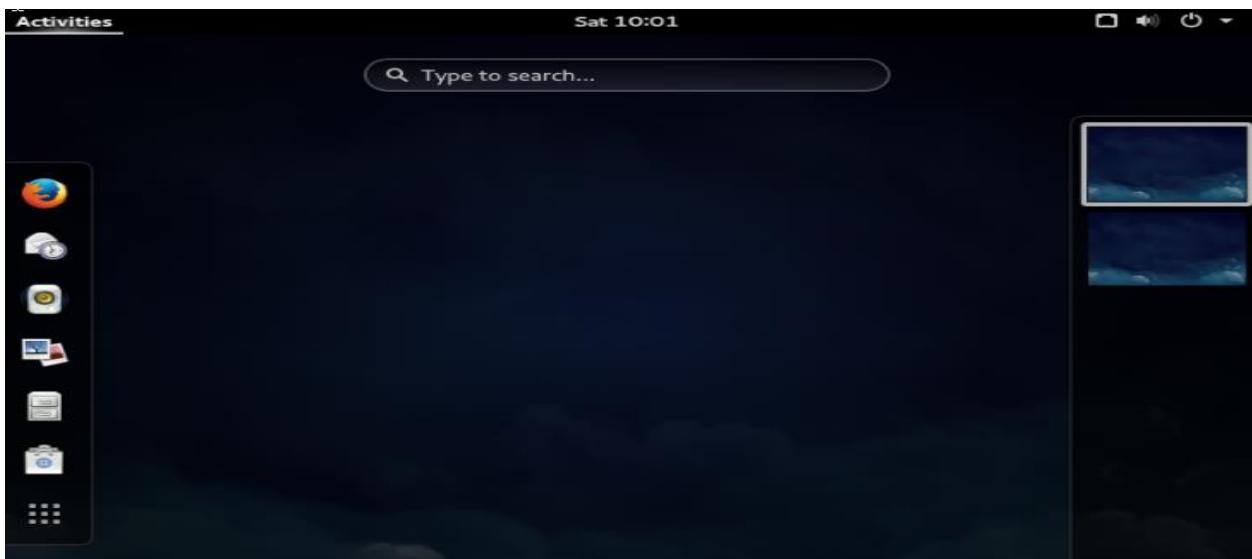
محیط گرافیکی یا GUI در واقع همان چیزی است که وقتی Desktop بالا می‌آید می‌بینید. وقتی بحث محیط‌های گرافیکی می‌شود، برخلاف ویندوز که یک محیط گرافیکی واحد دارد، لینوکس این چنین نیست. در لینوکس‌های مختلف، این محیط‌های گرافیکی متفاوت است اما در ویندوز این چنین نیست. مثلاً اگر از ویندوز 10 استفاده می‌کنید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

برای بیان بهتر این موضوع، چندین تصویر از محیط‌های دسکتابی لینوکس یا desktop enviroment در ادامه آورده شده است. اینکه کدام بهتر است بسته به سلایق افراد



تغییر می‌کند. شاید کسی از محیط خوش بیاید و همان محیط برای فرد دیگر مزخرف باشد.

GNOME 3

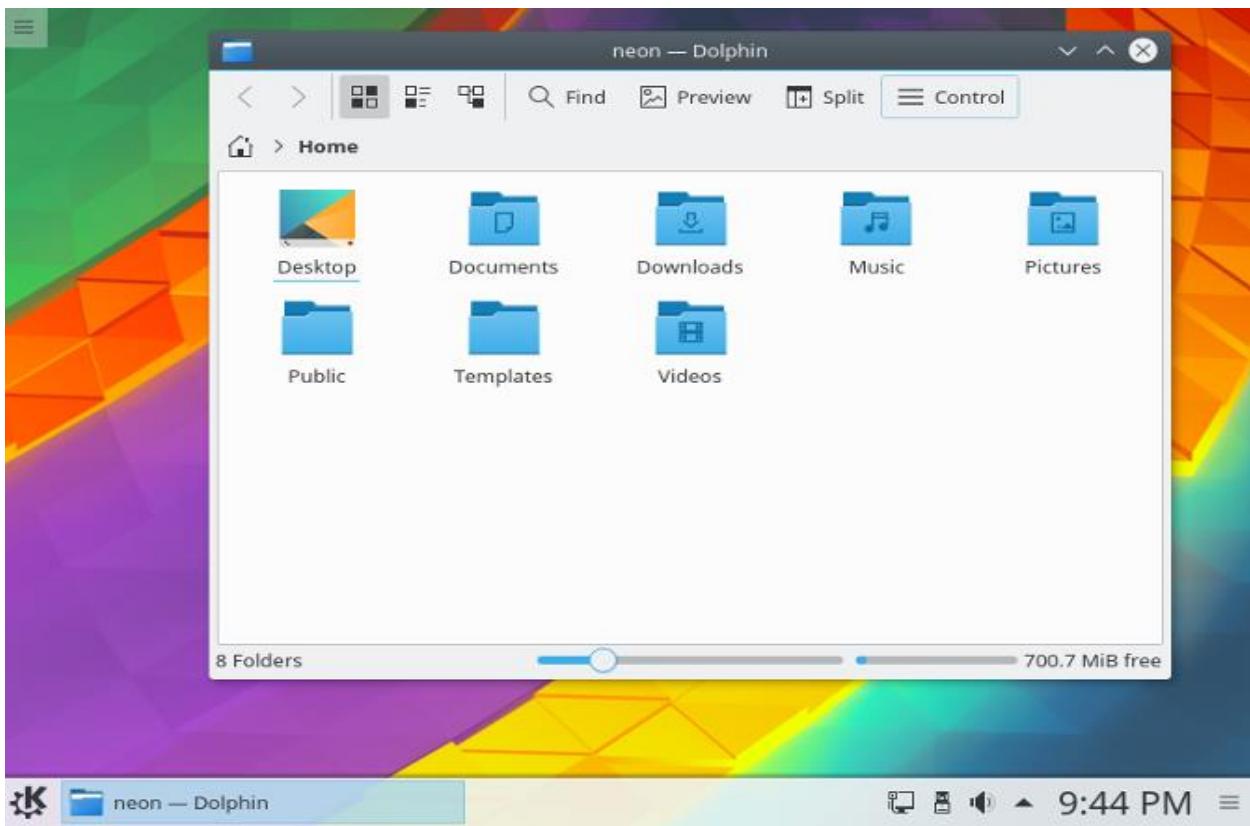


Ubuntu unity

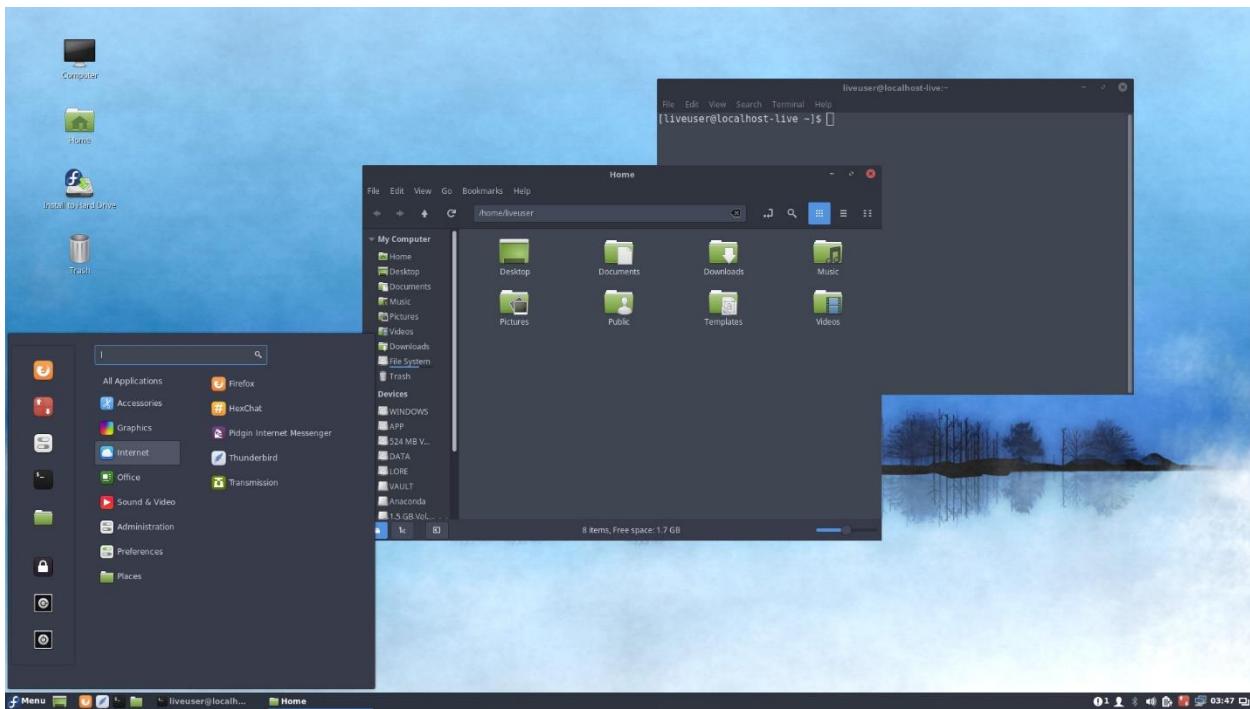


KDE



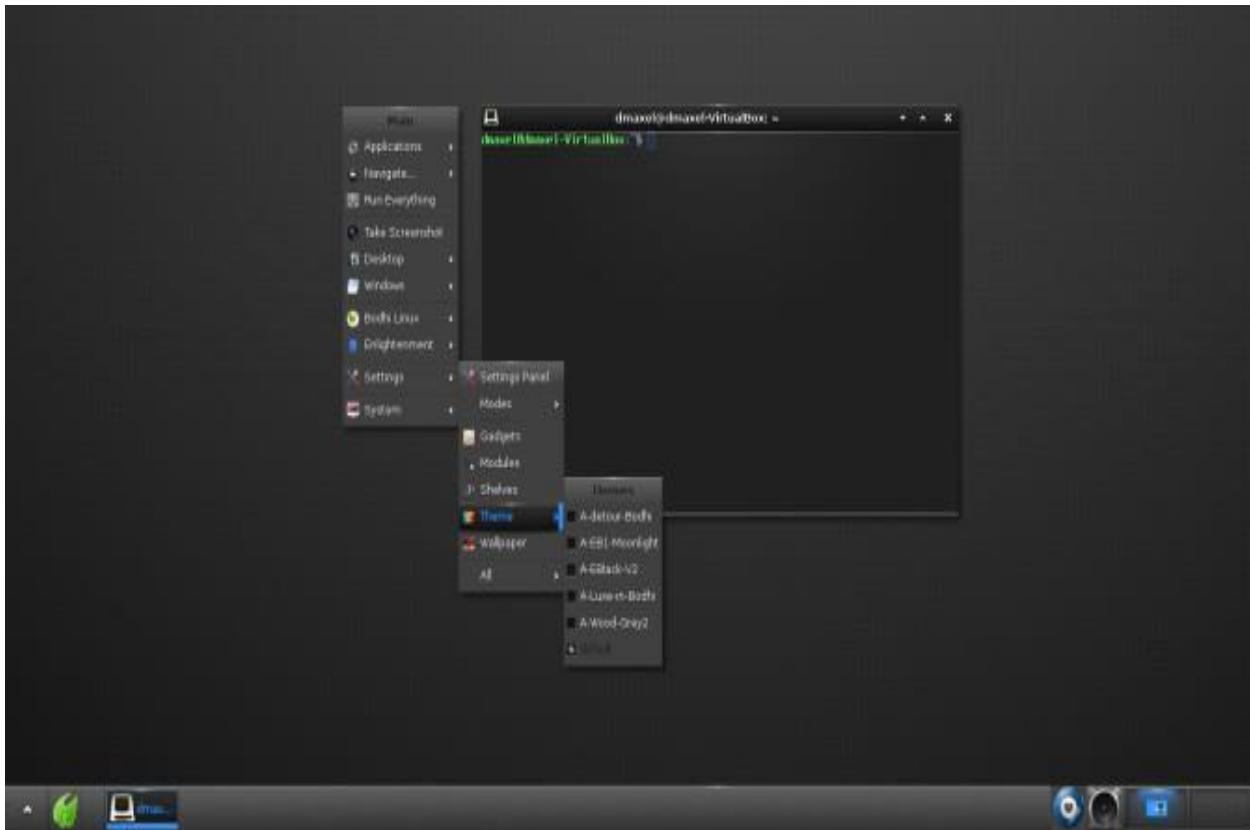


Cinnamon

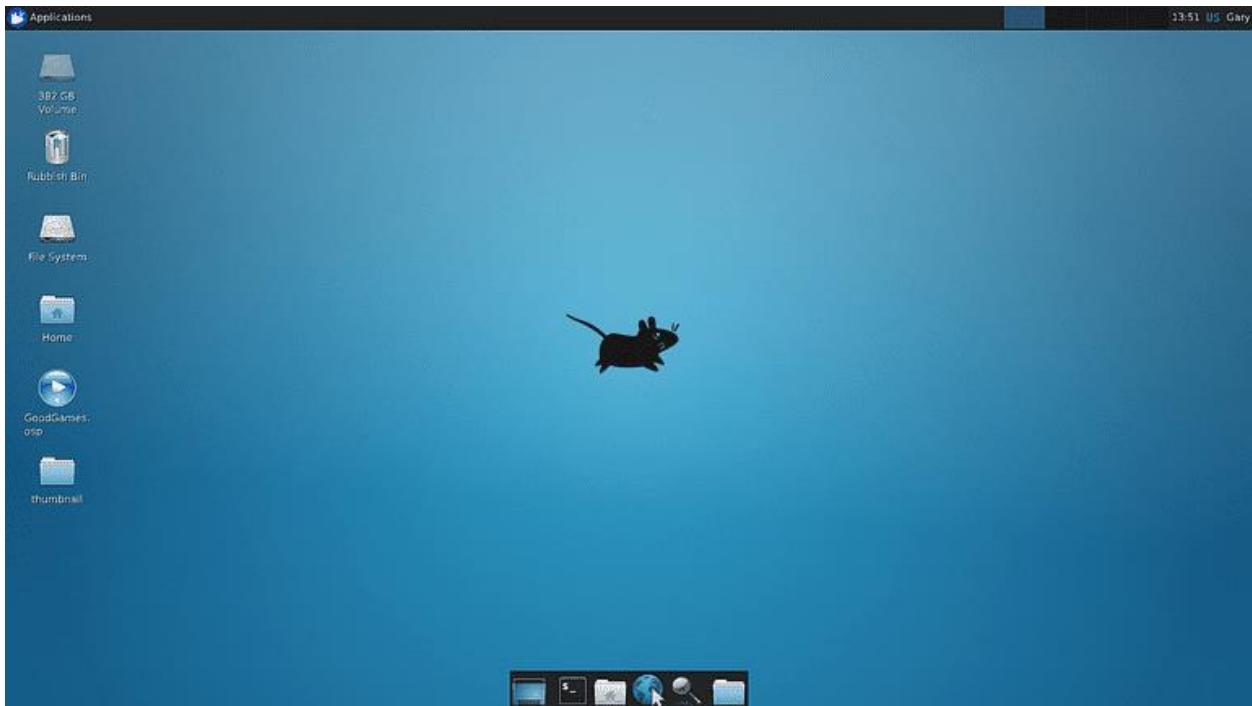


Enlightenment

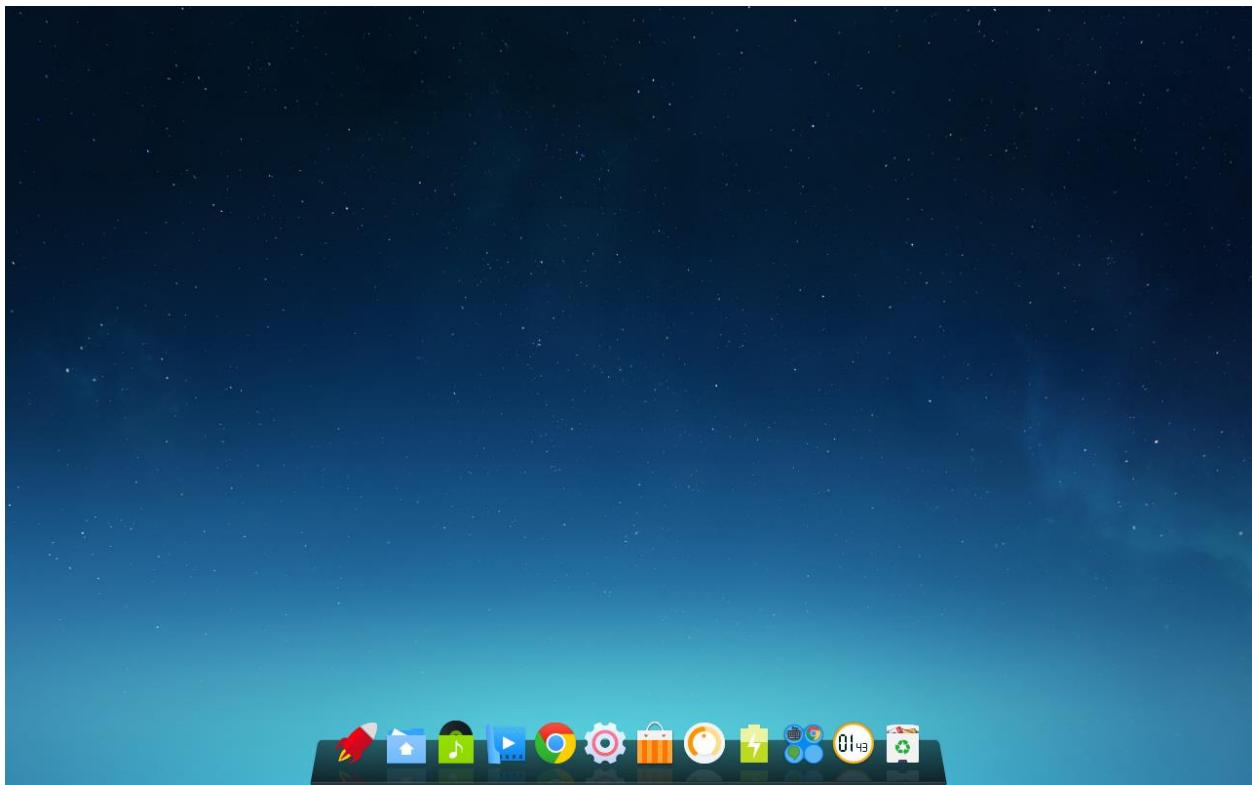




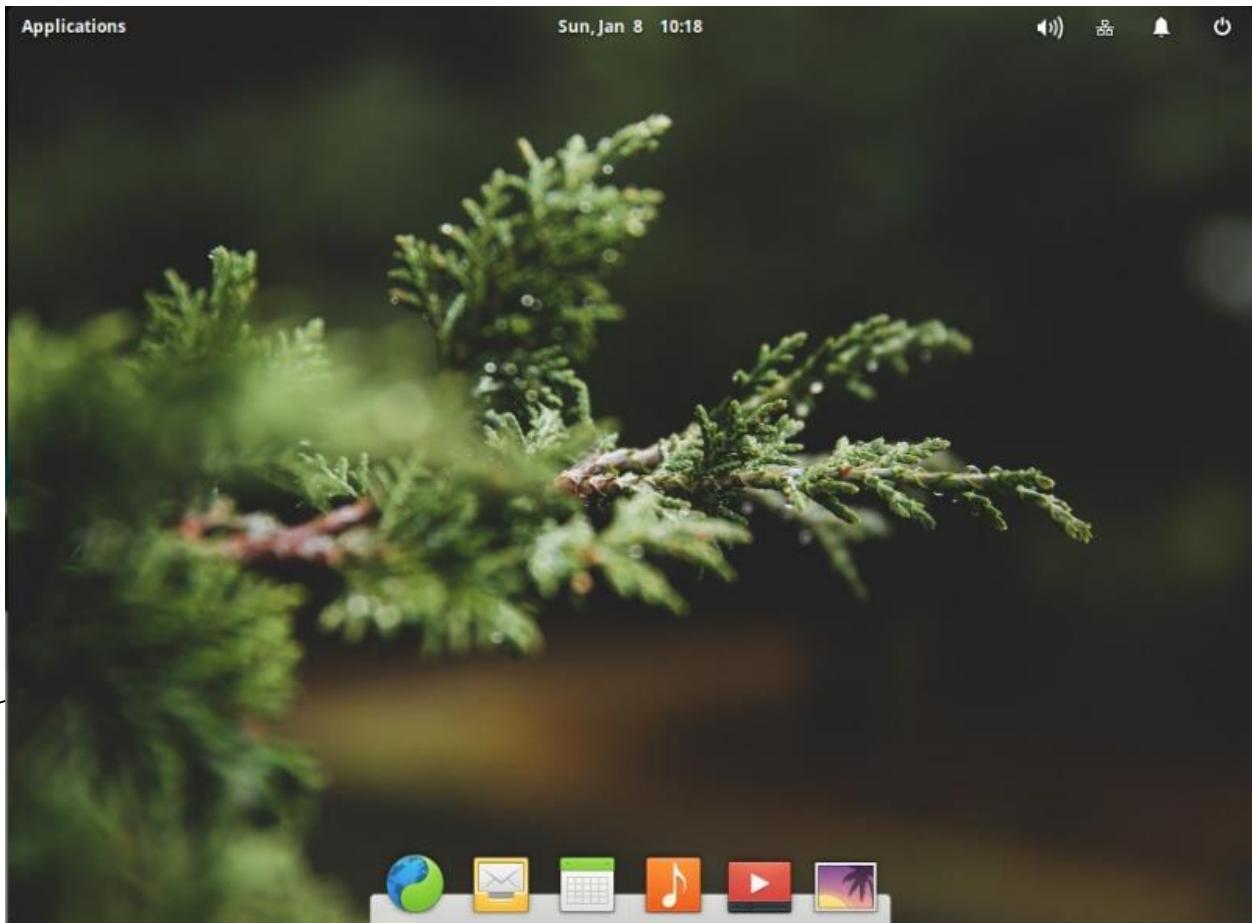
Xfce



Deepin

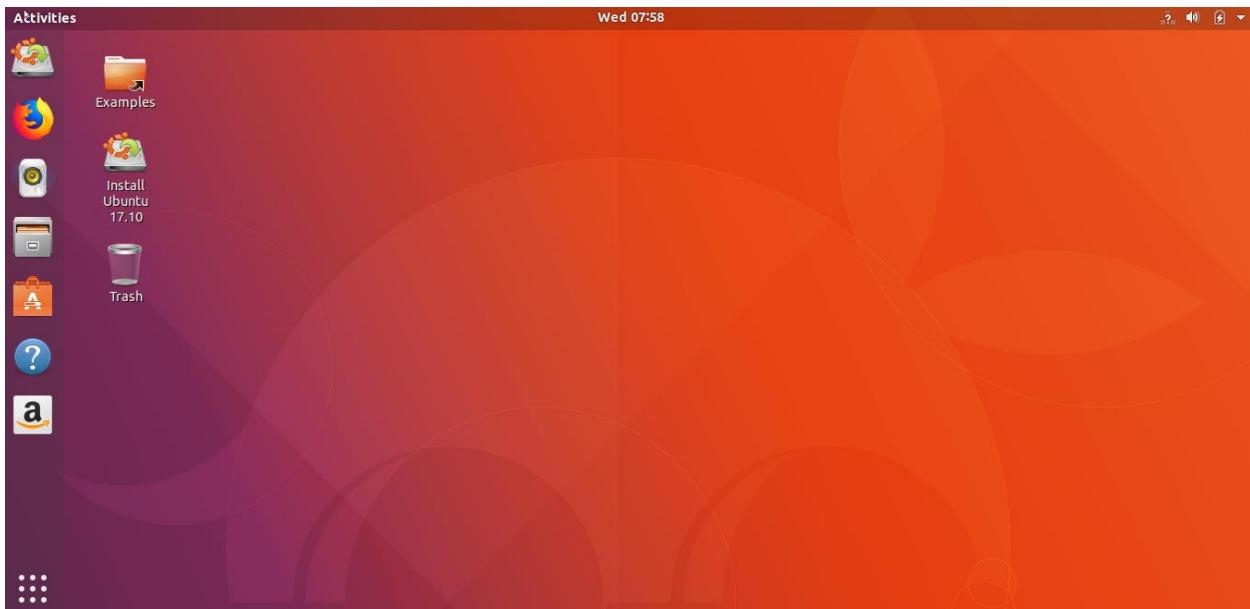


Pantheon



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 الان در حال اجرا هستند و این پردازش‌ها به کدام پردازندهٔ کامپیوتر می‌توانند دسترسی داشته باشند و این دسترسی چقدر طول بکشد.

3- باید سیستم فایل‌ها و اجزاء دسترسی را تشخیص دهد: کرنل مسئولیت‌های سنگینی را به عهده دارد و چون مسئول این است که به پردازش‌ها اجازه خواندن 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 مخاًنوم آیندم اجاًرتو تعیین
می‌کنه. دست از پا مخطا کنی میدم شیفت دیلیت کنه.

گیدری کردیما، آرزو پدر چوایان عیب نیست، چه حساب کردی
روی مخانوم 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 ڦول بعضی‌ها، ٿوی ٻلا د کفدر پهش می‌کنن shell
فارسیش چی می‌شه؟ ٿرکی پیلید سُن؟



شِل یا پوسٹه هم پهش می‌کنن



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 تو ش پنویسم I love
لَا و پفرستم پدای عیال شاید پیغام

روشی برای اجرای برنامه‌ها، کار با سیستم فایل، کامپایل کد کامپیوتر و مدیریت Shell کامپیوتر ارائه می‌نماید. اگرچه shell از الگوهای مرسوم محبوبیت کمتری دارد اما



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 گفته می‌شود. بسیاری از لینوکس‌ها هم گرافیکی دارند هم shell تکست.

قبل‌اهم اشاره شد که منظور از shell گرافیکی، همان دسکتاپی است که اغلب بعد لگین 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: No desktop کارتان با shell شروع می‌شود و نیازی به زحمت خاصی نیست.

With desktop: (مثل ubuntu که داریم استفاده می‌کنیم) باشد terminal را باز کنید. (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. Bash مخفف Bourne Again Shell است. این نام از این حقیقت سرچشمه گرفته است که bash با اولین shell یونیکس سازگاری دارد. (یعنی پوسته Bourne که به وسیله دستور Sh نمایش داده می‌شد).

حالا این حقیقت را به ما نمی‌کفتی
می‌تلکیدی؟ 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
بخونه عاشقونه غمارو رها کن په فردا نگاه کن عشقو صدا
کن... عشقو صدا کن



پا خدرا کشتی؟

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

Xterm -1

Uxterm -2

GNOME terminal -3

محیط‌های گرافیکی که base آن‌ها 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.



```
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 از هم 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 bash، return یا آنرا اجرا (run) می‌کند و خروجی را به شما نشان می‌دهد.

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 تعلق 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 وسیله عددی کاربر (uid 1000) نشان داده است. در اینجا گروه اصلی که sahar به آن تعلق دارد در groups نوشته شده. همچنین sahar به گروههای دیگری مثل adm(gid 4) و بقیه گروهها که آمده، تعلق دارد.

این اسمی و اعداد نشان‌دهنده permission‌ها 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 لایکن فعلی شما را نمایش می‌دهد و دستور grep تعریف اکانت شما را در فایل /etc/passwd نشان می‌دهد. در اینجا بجای user باید sahar 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 shells دیگری نیز کار کنید (با فرض اینکه نصب شده‌اند یا اگر نصب نشده‌اند می‌توانید نصب کنید). shells زیادی وجود دارد مثل dash، sh، csh، tcsh، ksh و...

برای کار با این shells 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 shells 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 دایرکتوری کنونی شناخته می‌شود مثلاً وقتی برای اولین بار به داخل لینوکس لاگین می‌کنید، کار با دایرکتوری home را آغاز می‌کنید.



این دایرکتوری په چه دردی می‌مُحوره؟

وقتی درخواست بازکردن یا ذخیره‌سازی یک فایل را می‌کنید، shell ما از دایرکتوری کنونی یا همان current directory به عنوان نقطهٔ رجوع استفاده می‌کند. مثلاً برای save کردن یک فایل، کافی است نام فایل را بدھید 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 محتويات 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 `ctrl+d` یا `exit` یا `ctrl+c` می‌توانید خارج شوید. این‌ها برای دست گرمی بود و گرنه صدھا دستور دیگر در دایرکتوری‌هایی مثل `/bin` یا `/usr/bin` 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 دنبالهٔ 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 پهم می گه پا این آیتم‌ها، کامندهامون شاسی پلند
می‌شه یا یه چیزی مثل اسقناچی که ملوان زپل می‌خورد البته
امروزه علم اثبات کرده اون اسقناچ نبوده که اون می‌خورده. هر
چیزی همون چیزی بوده که این خود هم می‌خوره

- ۱- آپشن‌ها (options)
- ۲- آرگمان‌ها (arguments)
- ۳- متغیرهای محیطی (enviroment variables)
- ۴- متاکاراکترها (metacharacters)



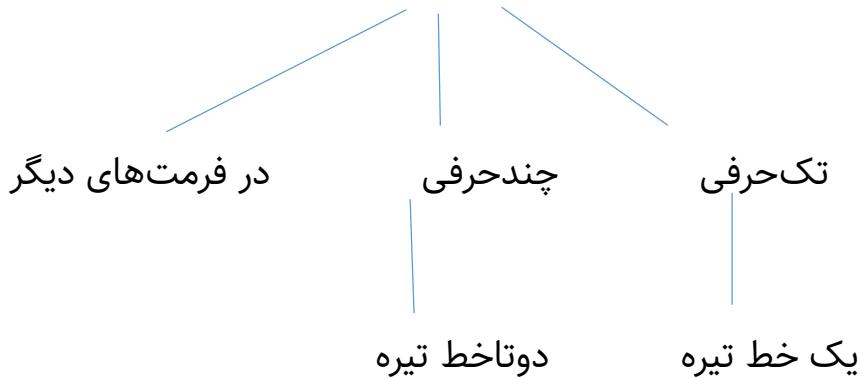
آپشن‌ها (options)

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

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

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

آپشن‌ها



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 نوشتیم. فکر کنم تا حالا هیشکی از ت چیزه
نگرفته. آخه خط تیره ای ؟؟؟ تمام پاورهامو ریختی پویم



```
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 بردہ می‌شوند).

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



پقیه اطلاعات لیست مربوط به اندکاره هر فایل پر حسب پایت، تاریخ و زمان آخرین پاری که دستتوون په فایل مخورده رو نشون می‌ده.
پاتریش نکشید اینارم پگ

- (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 "پرنامه‌ای دیگر" فکر کردم می‌خوای
مارو په خدای پذرگ پسپاری



خوب می‌خوای من پسپارم؟

تا پرنامه‌ای دیگر شما رو په خدای پذرگ می‌سپارم



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

-`la` : مثال

می‌خواود پکه می‌شه آپشن‌های تک حرفی رو توی یه مخلوط کن
پریزید و قاطیش کنید و کنار هم پیارید مثل `-ls -l -a`



یعنی این سه تا هیچ فرقی با هم ندارند:

`ls -l -a`

`ls -a -l`

`ls -la`

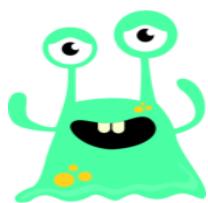


```
sahar@dance-with-linux:~$ ls -la
total 84
drwxr-xr-x 13 sahar sahar 4096 May  9 12:45 .
drwxr-xr-x  3 root  root  4096 May  9 12:14 ..
-rw-----  1 sahar sahar   14 May  9 12:45 .bash_history
-rw-r--r--  1 sahar sahar  220 May  9 12:14 .bash_logout
-rw-r--r--  1 sahar sahar 3771 May  9 12:14 .bashrc
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 کامندی 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`, `p`, حساب می‌شود.



خدا لگدش کنه که همه رو ټیکه ټیکه
می کنه

مثلثا فرض کنید نمی‌دانیم دستور `ls` چه کاری انجام می‌دهد. می‌توانیم از آپشن `help` 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.

مثلًا 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.

مثلًا 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 بیانیم، ip addr show dev lo 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 پاپد حفظ کنیم؛ ۰_۵

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.

پیداکردن 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 ھم 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 بخواهد هیولا پاشه

از کجا بعومنیم اصلاً چه کامندهایی هست؟ چه آپشن‌هایی
می‌گیرن؟ یا چچوری پیش‌رفته تر ارشون (استفاده کنیم و
اقرودنی‌های معجازشون چیه؟



مگه کلاس آشپزی با مانع است؟ بقیه قصه برای
چواب سوالنه

- چک کردن PATH

در shell تایپ کنید:



echo \$PATH

نشد



پنگوک چیز کن PATH قبیلشم یه عالمت دلار پذاره.

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         ntfsfalllocate systemd-escape
bzegrep        hciconfig   ntfsfix     systemd-hwdb
bzexe          hostname    ntfsinfo    systemd-inhibit
bzfgrep        ip           ntfsls     systemd-machine-id-setup
bzgrep         journalctl  ntfsmove   systemd-notify
bzip2          kbd_mode    ntfsrecover  systemd-sysusers
bzip2recover   kill         ntfssecaudit  systemd-tmpfiles
bzless         kmod        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       ulockmgr_server
chvt          loadkeys   ping4      umount
```



-2- استفاده از کامند help

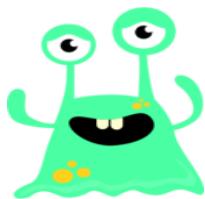
بعضی از کامندها هم در دایرکتوری‌ها یا همان اتاق‌ها دیده نمی‌شوند. در واقع این کامندها داخل خود shell از پیش ساخته شدند. برای دیدنش از کامند help می‌توانید استفاده کنید. منتهی خروجی این کامند چون خیلی زیاده، می‌توانید با نوشتن کامند زیر، به صورت صفحه به صفحه ببینید.

`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.

<code>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 ..> set [-abefhkmnptuvxBCHP] [-o option-></code>	<code>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-></code>
--	--



قبول نیست چیزی که ما نمی‌فهمیم رو پنویسی و ما فقط تایپ کنیم. اصلاً من بازی نمی‌کنم



قدار نیست اینچوری ٿموم شه، و گرنه ْغم انگيڏه. همشو می گه اڳ
خدا قبول کنه. تو فقط تحمل داشته پاڻ. اين را ياد پکير ڪو چائي
موندي هي گيچ نشي



يعني چي هي مي گه صفحه صفحه؟

يعني موستو اسکرول کنی مياد پاين پاڙ، مي ٽوني ٩ رو پڏندي ٿموم
کنی اين پاڙي گئيفو



اين دستور فقط با shell يا پوسته bash کار مي کند اگر با دیگري کار مي کنيد، اين
دستور اجرا نخواهد شد.

3- مي توانيد help- را با دستوري که مي خواهيد به کار ببريد

خيلي از کامندهايي که وجود دارند، آپشنی به اسم help هم گذاشته اند تا کاربر گيچ
نشود. مي توانيد کامندي را که مي خواهيد بغلش --help را استفاده کنيد.

مثلا فرض کنيد نمي دانيم ڪا چه آپشن هايي را قبول مي 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 خواپم می بینم. چون دا کامند دیگه ای پلد نیستی؛ یکم خوشحالون کن دیگه چیه همش تکراره تازه بالا هم اینو مثل زدی.



تو دیگه از تنوع حرف نتون. توی این هزار سالی که از مخدا عمر گرفتم فقط چیافت همین شکلی بود. ما هر شوختی ای کردیم درین از یه اونها روی صورتت



هر||||||| سالنه ؟؟؟ مچجالت پکش از هیکلت.

پاپام مخدایام ر عادت داشت ما هر خدیتی که می کردیم می گفت چهل سالنه از هیکلت مچجالت پکش. اگر تو رو می دید که هزاره ای، خیلی خبر کیف می شد.





بایا تم مُرد و تو رو پرای ما په ارث گذاشت، همهٔ خریت‌ها نسل په نسل
چهرخیده و اپیاشته شده توی تو. تو هم خراب شدی روی سر ما، یه ذره از
خریت‌تو پا مُودش په گور نپرد یا ارث می‌ذاشت پرای بقیه.

مثلًا با دستور `mkdir` می‌توان یک دایرکتوری ساخت، برای اینکه اطلاعات بیشتری از او
داشته باشیم:

این دستور آمار همه رو می‌ده؛ پنځن پینیم ځونه
لینوکس کچاست؟



```
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:~$
```



تُوی واقعیت پرای مخونه ساختن پاس عین این خرد پول چمع
کنی، صد نقدرو راضی کنی پیان پسازن بعد اینجا با یه mkdir
یه آنچه میزتی زندگیتو میکنی



مگه شما خرها هم مخونه میسازین؟



آره مگه مثل شما آواره سحر نامیم؛ ما مخونه زندگی
میسازیم روی پای خودمون و امیسیم. بیدون از اینجا
واقعیت محضنه

کم موئده بود تو یکی بهمون درس اخلاق پدی



man -4 دستور

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

نکنیم. خانواده نشسته. نشون می‌ده لباس چی پوشیده.





می خواستم پنهان برای اطلاعات بیشتر به روزنامه های کنیرالانتشار
مراجعه کنید. این همیش می خواهد مارو پیچونه. حالا ببین کی گفتم



یعنی چی هی کلمه man کنار کامند مردم بیاد. ملت
ناموس دارن. چه وضعیه،



تا پپ کردم این man رو.
what manal page do you want?
می گویی وقتشه پدم بیلمو بیارم؟



باید کامندی که می خوای رو هم بغلش بیاری.

برای مثال بزنید man mkdir



دوباره قفل کرد روی یه دستور چدید. پاپا ۹۹۹۹۹۹۹۹۹۹



```
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)
```

info - استفاده از

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.



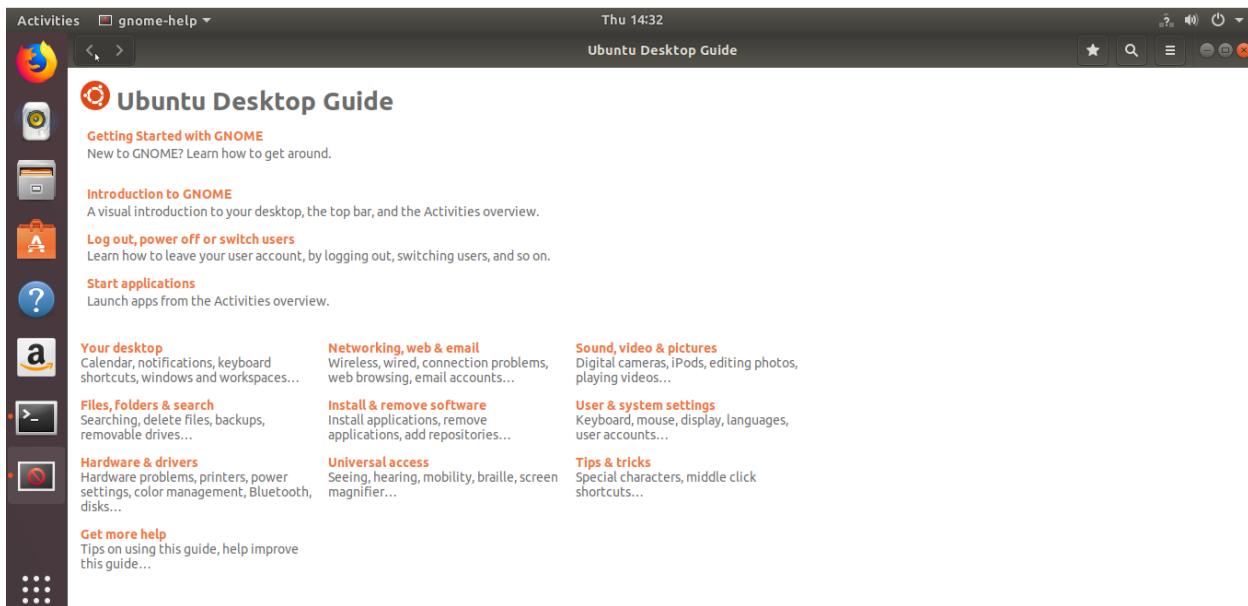
مُثلاً اینکه دیدوْر هم چی پوشیدن

تو نمی‌خوای 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.



آرگمان‌ها (arguments)

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



ملت چه توانایی‌ها بی دارن. نمی‌شه منم په فرزند
خوندگی قبول کنن؟

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



یعنی کلا اضافی‌ان؟



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

عین پلیسراقتار می‌کن، چاسوشه؟



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

```
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:/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
```

جايی که آرگمان‌ها با آپشن‌ها قاطی می‌شوند

آپشن‌ها در واقعی نوعی آرگمان‌اند.

گاهی اوقات یک آرگمان با یک آپشن شريک می‌شود (!) در اين حالت، آرگمان فوراً بعد از آپشن قرار می‌گيرد.

وقتی می‌خواهيم به دنبال آپشن‌های تک حرفی، آرگمانی بياوريم، بعد از یک space می‌آيد. اما اگر آپشن مد نظر، تک حرفی نبود، برای اينکه آرگمان را به دنبال آن بياوريم، از علامت تساوي = استفاده می‌کنيم.

مثلاً: دستور ls --hide=Desktop



```
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 خوبیست. مگه اون ٿو چھپدڙه؟

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



کشتی خود ٿو فهمیدیم اسپیس نداره دیگه

مثال زیر، یک آپشن تک‌حرفی است که به دنبال آن یک آرگمان آمنه 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.

اونچا که می که پساز منظورش `c` مخففش اومده توی آپشن. اونچایی هم که می که "فایل" منظورش `file` که توی آپشن مخففش `f` اومده. اون حرف `V` هم که وسط اومده، منظورش `verbos` (به معنی طولانی، دراز) که یه پیام متنی می ده که پنهان پک آپی که می خواستی ساختم. حالا اگر پرید توی دایرکتوری `/home/user` یوشه هه رو می بینید.



متاکاراکترها

یکسری کاراکترهای خاص وجود دارند که نمی توانید آنها را در کامندها استفاده کنید. به اینها متاکاراکتر گفته می شود.

هر کدام از این متاکاراکترها برای `bash` معنی خاصی دارد. انواع مختلفی هم دارند. یعنی از طریق این کاراکترها ما می تونیم با `bash` راحت تر حرف بزنیم و چیزهای بیشتری ازش بخواهیم. در ادامه به معرفی بعضی از این متاکاراکترها می پردازیم.

متاکاراکتر نقطه ویرگول

نقطه ویرگول یا semicolon یا ; کاراکتری است که دو دستور را از هم جدا می‌کند. اگر بنویسیم:

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 ویرگول نمی‌ذاره پقیه به عنوان آرگمان پاس داده پشتن،
هر کدام رو یه کامند جدا می‌دونه

چه فهمیده. کار ایشون رو دوست دارم. پیاد من و لپتوکس رو
linux ; khare پذرنه



اشتیاه فهمیدی، کار نقطه ویدگول جدا کردن دو تا کامند از
همدیگرس نه وصل کردنش. یعنی می‌شه توی یه خط چندین کامند
رو نوشته و یا نقطه ویدگول جدا کرد.





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

منم همینطور



متاکاراکتر space

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

متاکاراکتر امپرسند &

بعضی از کامندها برای اینکه کامل اجرا شوند، طول می‌کشد (ماهم نسل بی‌حواله) 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 معمولا در پایان یک برنامه می‌آید.

کامندهایی که مربوط به فرمت متن می‌باشند یا به اصطلاح text formatting مثل troff nroff یا commands که بعدا 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. مثلًا میزان پرشدن هارد دیسک شما یا کاربران خاصی که لاغین کردند.



شل اسکریپت چیه؟

یکسری برنامه 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

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



عادت داری حرف تکراری پذیری؛ 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 پیش‌اومنده

مثلاً 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، معمولا 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 این کار 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 قبلا پخته این 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` اسامی کاربرانی که در هر فایل `/etc/passwd` شروع می‌شوند را گرفته و آن‌ها را برحسب 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 متاکاراکتر اگر `date` درست اجرا شود یعنی آن صفر باشد، دومی دیگر اجرا نخواهد شد. به عبارتی این متاکاراکتر، هر کدام که صحیح 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 می‌گذاریم 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.

متاکاراکتر < یا خروجی 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 خروجی ارسال کند. مثل زمان که در برنامه پایتون با دستور print می‌گفتیم خروجی را ارسال کن و مثلًا روی صفحه یک سلامی به ما بده hello را نشان بده. مشابه همین، این خروجی به دست bash می‌رسد، به طور دیفالت این خروجی را روی صفحه نمایش، نشان می‌دهد، اما کاراکتری وجود دارد به اسم "بزرگتر از" یا > یا greater than به طور مخفف .gt.



پایام نمی‌مونه، من همیشه < یا > قاطی می‌کنم. از کجا بفهمم؟

پیشنهاد این ماهیه به کدام طرفه؛ هر جا که غذا پیشتر پاشه ماهی
به اون سمت دهنیش پاره.





ماهیه چرد خید پچی؟ اصلاح دهن ماهی های من ^ این شکلین.



عاشق اوئیم کە دەنگش پستىس. خەرەچان سختىش نکن دېگە. يە شەلۋ ياد بېگىر دېگە قاطى نكىنى.



اصلن مىم مىخواهم ياد بېگىرم.
مىم بازى

نە تو دېگە. هەر چى چك و چۈنۈرە اينچىجاست. بە اندارە كافى از خەرە
مىكشىيم. تو يىكى وارد نىشۇ پا اون حافظە سە ئۆزىيەايىت



این علامت، يك متاكاراكتر به شمار مىرود و ھر وقت bash اين کاراكتر را ببىند متوجه
مىشود کە باید خروجى را در يك فايل برىزد. چە فايلى؟ فايلى کە شما نام آنرا بعد از
علامت بزرگتر مىآوريد.

مثلا در ترمinal مىنويسىم: cal

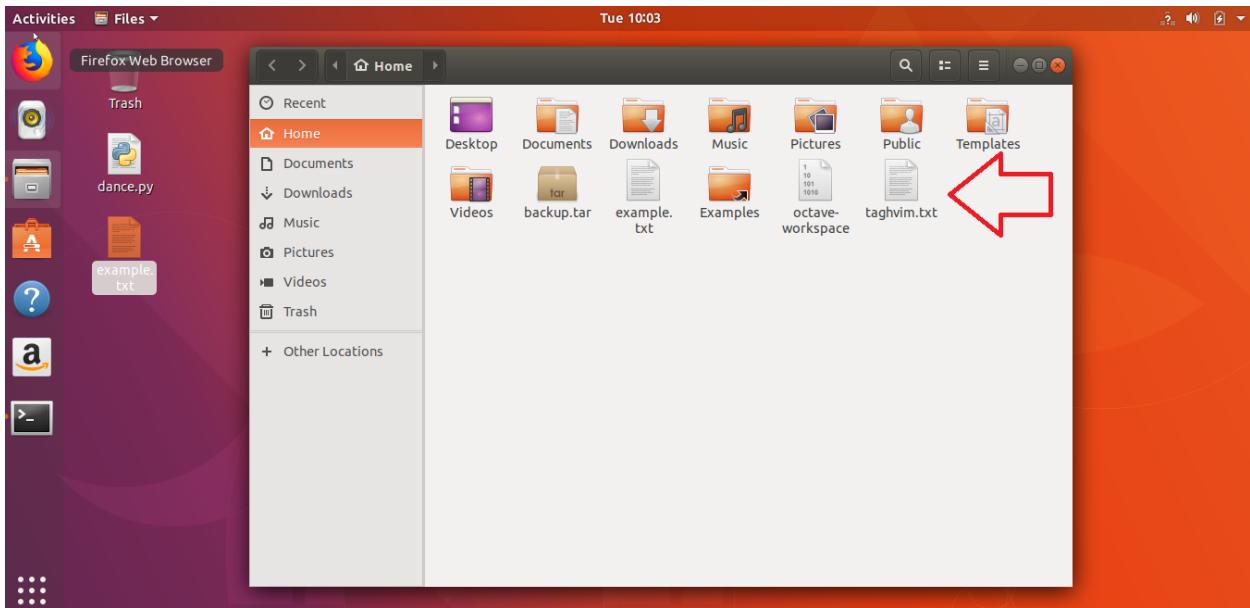
با اين کار تقويم را روی صفحه نمايش به من نشان مىدهد، اما اگر بنويسىم:

cal > taghvim.txt

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

ديگە خروجى اى روی صفحه نمىبىنيد، و خط prompt براي گرفتن دستور بعدي مدام
چشمك مىزند و اين يعنى دستور قبلى اجرا شده. اما خروجى كجا رفته؟ خروجى درون
فايل taghvim.txt درون دايirkتوري جاري حضور دارد. دايirkتوري جاري ما الان فolder
است. (يادمون نرە كە با pwd هم مىتونستىم ببىنىم توى چە دايirkتوري
ھستىم)





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.



مثل موقعی که په این یغچیال گنده‌ها پول می‌ندازی توش
ت پهت آپه معدنی پده، تا پولوندی 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 ٿوی یغچاله. 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 پاں



متلا پرنسمه یه چای محالی داده این شکلی: ----- * 2 = عددی که ٿوی چای محالی اول می‌خوای قرار پگیده رو می‌پرسه. این مسئله فرقی ٿوی نصوه خردی ٿو داره عایا؟

اره. اصلاً من دوست دارم پرنسمه چوری کار کنه که دو دو ٿا رو پڙنه 5.7



ھفت و نیم ٦٦٦٦٦٦ چطور ممکنه؟



پرنسامه خودمه دوست دارم اینچوری پاشه، خیلی دخالت می کنیا،
می گم از سناریو پندارنست پیدرون. حریم شخصی حالیت نیست؟



برنامه تا وقتی که از یوزر ورودی می خواهد، متوقف می شود. برای مثال، در مثال دو دوتا می شه هفت و نیم، برنامه منتظر می ماند آن 2 یا هر عددی دیگر را وارد کنید تا ضرب انجام گیرد و خروجی نشان داده شود.

حتی می توان 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.

مُوپه چدیدا پا bash هم حرف می زنی! چشمم روشن. فشار
امتحانیتنه 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 خرد تو خوبی شد



به ادب! در این حالت نمرات رو به عنوان ورودی به برنامه moadel می‌ده، پس از اینکه معدل رو حساب کرد، خروجی رو تولی 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





چه پچه مُوپیه، سر په راهه، دوسشن دارم.
کار مُخودشو می کنه.

با هم یه مورد عینی را امتحان کنیم. در bash برنامه‌ای داریم به اسم wc که مخفف word count 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 اجراش کن، باید آخرش **ctr+d** را بزنیم:

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

اینجا باید **ctr+d** را بزنم:

```
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 می گه من می خوام پشینم پشمایش.



دست شما درد نکنه دیگه شدیم دروغگو

ای چانم! کجا پودین شما تا الان؛ خودش مخالی پنده
پاپا. چلو خودم، پیستا کارت پستال "عذریم تو تنها
عشق منی" خرید. شما غم به دلته راه نده.



کامند `wc` هم مانند سایر کامندها، آپشن‌های زیادی دارد.

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

فرض کنید فایل `example.txt` محتویاتی این چنینی دارد:

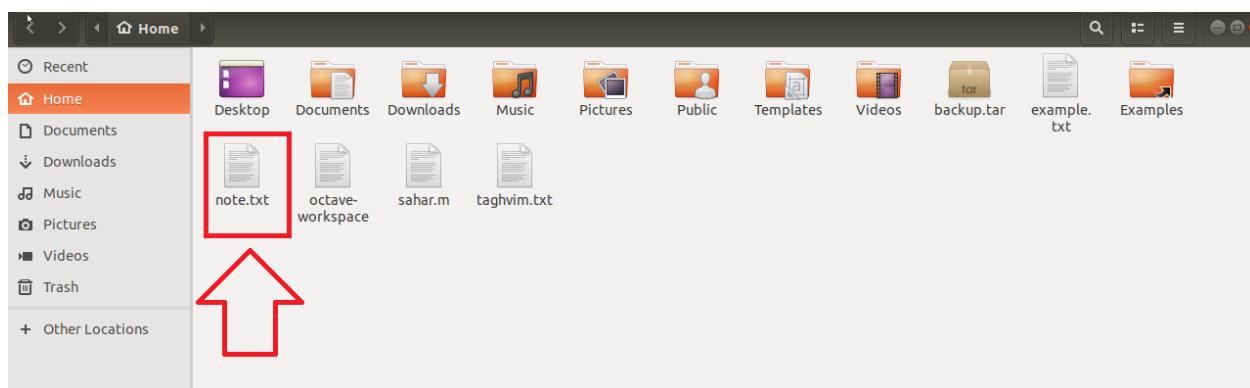


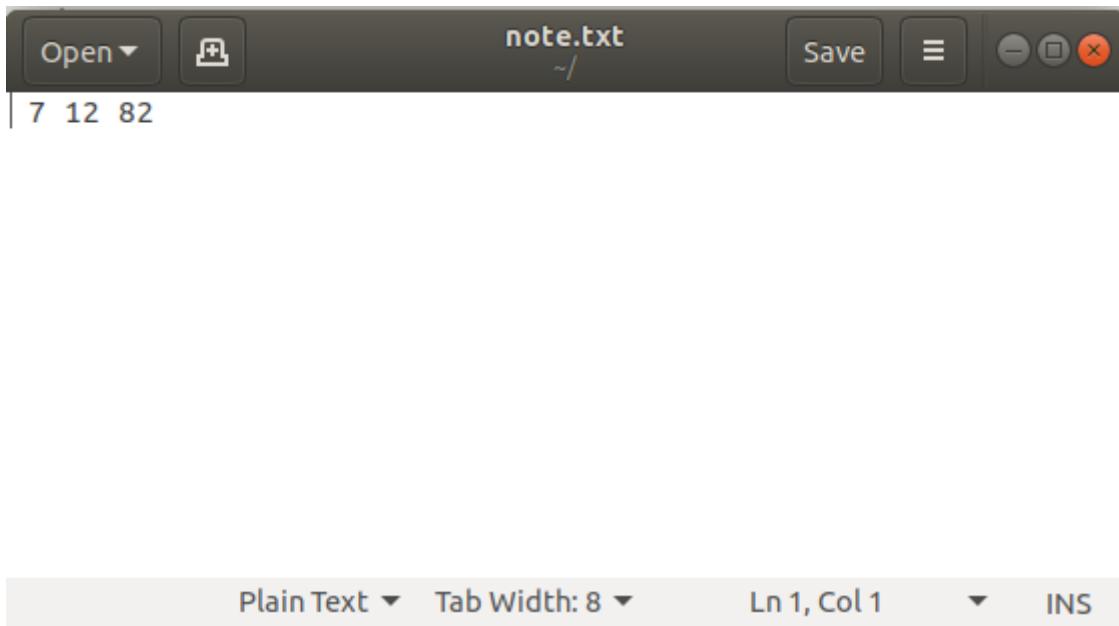


می خواهیم:

```
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 محتویات داخل دایرکتوری /dev را لیست کردیم، حالا لیستی که به دست آمده را به عنوان ورودی کامند wc نمایش دهد:



فان: گلاب په روتون می‌شے 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 قديما گفتن کار چوهر مرده

هد چي شما پيگد



اين چش شده پرا کمنگ؟

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



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

متغیرهای محیطی (enviroment variables)

کامندها به طور کلی به دو دسته تقسیم می‌شوند:

- کامندهای داخلی یا internal
- 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 اوْنُو ٿوی ڙپان مُثلاً ٿوی python نوشته‌یم
و اوْنُ چویده و فایلش رو می‌ده. دیگه shell می‌تونه اوْنُ پھونه

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 اينکه درون اتاقی (پوشه‌ای، فایلی) مناسب قرار دادیم، می‌توانیم توقع 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 اين فایل اجرایی باید با کلمهٔ 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 کلمهٔ اول کامند شما یکی بود پیدا کرد آن را اجرا می‌کند و بقیهٔ کامند شما را که با `space` از هم جدا شده‌اند را به صورت آرگمان اجرا می‌کند.



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



پو گو پینیم کی گل می‌شه؟



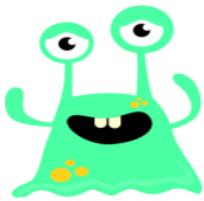
گلش پستگی په مخودت داره و راویه شوتت! تو هم که کلا شوتی
مطمئن پاش روی تو یکی چواب می‌ده



مثلا کامند ping یک کامند اکسترناال 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 نوشتمن متغیرهای محیطی و تنظیم آنها در فرصتی جدا بحث خواهیم کرد.



پنهانی دستی دستی پیچوندی مارو سحر؟



نه نوشتمن متغیرهای محیطی خودش په پرنامه نویسی نیاز داره و چندتا کامند دیگه.

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





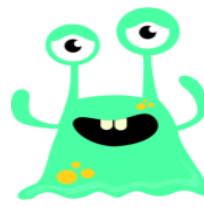
هی فیلسوف اعظم ما داریم می‌ریم، حالا خردپیار و
باقالی پار کن



مگه باقالی‌ها رو با خودت نمی‌بری؟



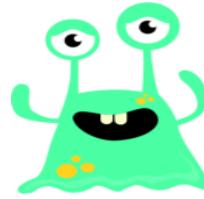
نه گذاشتم و اسه تو پوسته بکنی



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





Dance with Linux

Saharshaker.com

Update: 97/4/20