

بسم الله الرحمن الرحيم

# جزوه آزمایشگاه سیستم عامل

## فهرست مطالب

۵	فصل اول
۵	دستورات DOS
۱۱	دستور کار عملی
۱۴	فصل دوم
۱۴	تعریف ماشین مجازی
۱۹	انواع نصب ماشین مجازی
۲۰	Virtual box چیست؟
۲۰	Virtual box چه کاربردی دارد؟
۲۱	Virtual box نصب
۲۵	چگونه می توان در virtual box یک سیستم عامل linux suse نصب کرد؟
۵۸	مزیت های اصلی استفاده از ماشین های مجازی در مقابل سیستم عامل ها
۶۳	دستور کار
۶۳	طریقه ی نصب و برپا سازی Wine
۶۴	تنظیمات اولیه
۶۴	طریقه ی نصب نرم افزار با wine
۶۵	افزودن کتابخانه
۶۵	اجرای چند برنامه ی ساده
۶۵	اشتراک گذاری فایل بین اوبونتو و ویندوز
۷۰	شبکه محلی دو ویندوز در VmWare
۸۷	فصل سوم (لینوکس)
۹۱	طلوع لینوکس روی میزی (Desktop Linux)
۹۳	کدام توزیع گنو/لینوکس را انتخاب کنیم؟
۹۷	ساختار سیستم فایل (File System) در یونیکس (UNIX)

۹۸	..... مفهوم سیستم فایل یونیکس (UNIX) و لینوکس (Linux)
۱۰۱	..... GRUB و LILO و فرآیند راه‌اندازی لینوکس
۱۰۱	..... محیط x windows چیست؟
۱۰۴	..... محیط کاری چندگانه
۱۰۵	..... محیط گرافیکی KDE
۱۰۶	..... تاریخچه لینوکس و ویندوز:
۱۰۷	..... Ubuntu 16.04 از طریق Boot from CD نصب
۱۲۰	..... رفع اشکال صفحه مشکی بعد از اولین بوت:
۱۲۲	..... نصب درایو گرافیک: (NVIDIA Driver)
۱۲۷	..... نصب پلاگین پخش MP3 و دیگر امکانات نظیر پخش ویدیو:
۱۲۹	..... استفاده از دستورات در خط فرمان
۱۳۴	..... دستور کار لینوکس
۱۳۴	..... فصل چهارم: برنامه‌نویسی bash script
۱۳۵	..... اسکریپت پشتیبان‌گیری خیلی ساده
۱۳۵	..... متغیرها
۱۳۵	..... مقایسه متغیرها در دو زبان برنامه‌نویسی
۱۳۶	..... متغیرهای محلی
۱۳۷	..... شروط
۱۳۸	..... حلقه‌های for و while و until
۱۴۱	..... فصل پنجم: اندروید چیست؟
۱۴۱	..... مقدمه‌ای از جاوا
۱۴۲	..... Eclipse و ADT توسعه اندروید و محیط موتور JDK
۱۴۶	..... ایجاد یک پروژه جدید
۱۴۷	..... اندروید استادیو
۱۵۳	..... رفتن از یک اکتیویتی به اکتیویتی دیگر
۱۶۰	..... ایجاد پروژه Hello world
۱۶۷	..... ایجاد یک صفحه login

۱۷۶.....	Splash	ایجاد یک صفحه‌ی
۱۷۹.....		نحوه‌ی افکت گذاری بر روی اکتیویتی ها
۱۸۰.....	ALARM	در ساعت مشخص
۱۸۸.....		طریقه‌ی ساختن یک پایگاه داده در اندروید
۱۹۴.....	SQLite	ادامه‌ی پروژه‌ی
۲۰۰.....	Insert	تراکنش
۲۰۷.....	Delete و Update	و Insert آموزش تراکنش
۲۰۸.....	Update	متد
۲۰۸.....	Delete	متد
۲۰۹.....	php,json	ارسال اطلاعات به سرور با استفاده از
۲۱۳.....		نکات مهم

برای اجرای تمام این دستورات از مسیر زیر استفاده می‌کنیم:

Start → run → cmd

دستورات	وظایف
نام درایو ←	برای تغییر درایو
mkdir ← اسم پوشه	برای ساختن پوشه
cd ← اسم پوشه	برای وارد شدن به پوشه
cd.. ←	برای یک شاخه به عقب برگشتن
cd\ ←	برای به اول درایو(ریشه)برگشتن
copy ← .txt اسم فایل متنی con	برای ساختن فایل متنی
-hi This is a test Ctrl + z	
type ← اسم فایل متنی	برای مشاهده فایل متنی
rd ← اسم پوشه	برای حذف پوشه
hostname	برای نشان دادن اسم کامپیوتر
ipconfig	برای نشان دادن ip address کامپیوتر
net stat	برای نشان دادن کامپیوترهای متصل در شبکه به کامپیوتر ما
del ← اسم فایل متنی	برای حذف فایل متنی
dir	برای لیست گیری دو مدل: - پوشه به صورت <dir> - فایل‌ها با am و pm (حجم) نمایش داده می‌شوند
copy ← آدرس مقصد آدرس مبدأ	برای انتقال(کپی) فایل متنی

← type .txt اسم فایل متنی	برای دیدن محتوی فایل متنی
.txt	پسوند اسم فایل متنی
*.dll	اسم هرچی ولی پسوند dll باشد
???.txt	به جای هر حرف از علامت?(۳حرفی) استفاده شود
← آدرس مقصد   آدرس مبدأ   move دستورات	این دستور معادل cut کردن فایل است نه برای پوشه
← آدرس مقصد   آدرس مبدأ   xcopy	برای کپی کردن پوشه
/s	برای سوئیچ
← اسم دستور   help	در اختیار گذاشتن راهنمایی از دستور به همراه سوئیچ‌های قابل استفاده با آن
attrib +h +r +s +a attrib h -r -s -a	برای مخفی کردن فایل +: برای اعمال خصوصیت روی فایل -: برای برداشتن خصوصیت از فایل
help   attrib	برای سوئیچ attrib دو مدل: s-1 /s (زیرپوشه) d-2 /d (پوشه)
← اسم درایو فلش attrib   -s -h /s /d اسم درایو فلش	برای برداشتن خاصیت سیستمی و hidden بودن از همه اول زیرپوشه‌ها و سپس پوشه‌ها (برای ویروس کشی فلشی که ویروسی شده)
← اسم پوشه   rd اسم درایو فلش	برای پوشه‌ای که با delet کردن از روی فلش پاک نمی‌شود
calc	برای نمایش calculator (ماشین حساب)
write	برای نمایش word pad
osk	برای نمایش کیبورد مجازی سیستم

magnify	برای فعال کردن ذره بین سیستم
ms tsc	برای نمایش پنجره remote

### خصوصیات فایل‌ها:

۱-خواندنی (read only): r

۲-پنهانی بودن (hidden): h

۳-بایگانی بودن (archive): a

۴-سیستمی بودن (system): s

- برای نمایش پوشه‌های مخفی شده :

از منوی

Tools → folder option → show hidden

برای پنهان کردن پوشه :

ابتدا روی پوشه کلیک راست کرده و سپس

Properties → hidden

عکس‌های گرفته‌شده از اجرای

فرامین cmd

```
C:\Windows\system32\cmd.exe
D:\>mkdir os
D:\>cd os
D:\os>mkdir lab
D:\os>cd lab
D:\os\lab>mkdir lab1 lab2
D:\os\lab>cd..
D:\os>tree
Folder PATH listing
Volume serial number is 00000200 94D6:B5CA
D:
├── lab
│   ├── lab1
│   └── lab2
D:\os>
```

```
C:\Windows\system32\cmd.exe
C:\Users\p.ahmadi>d:
D:\>
```

```
C:\Windows\system32\cmd.exe
D:\os\lab>copy con sys.txt
this is a test
^Z
1 file(s) copied.
D:\os\lab>dir
Volume in drive D has no label.
Volume Serial Number is 94D6-B5CA

Directory of D:\os\lab

11/03/2011 09:04 AM <DIR>      .
11/03/2011 09:04 AM <DIR>      ..
11/03/2011 09:02 AM <DIR>      lab1
11/03/2011 09:02 AM <DIR>      lab2
11/03/2011 09:04 AM             16 sys.txt
1 File(s)                16 bytes
4 Dir(s) 94,489,948,160 bytes free

D:\os\lab>
```

```
C:\Windows\system32\cmd.exe
D:\os\lab>copy con sys.txt
this is a test
^Z
1 file(s) copied.
D:\os\lab>
```

```
D:\os\lab>type sys.txt
this is a test
```

```
D:\os\lab>copy sys.txt e:\os1
1 file(s) copied.
```



```
E:\>copy con ahmadi.txt
this is a test
^Z
        1 file(s) copied.

E:\>copy a*.txt d:
ahmadi.txt
        1 file(s) copied.
```

```
D:\os\lab>ren sys.txt s.txt

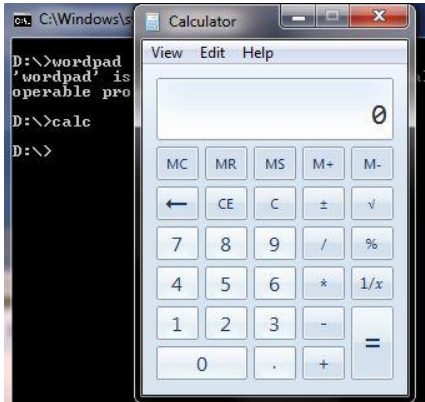
D:\os\lab>dir
Volume in drive D has no label.
Volume Serial Number is 94D6-B5CA

Directory of D:\os\lab

11/03/2011  09:08 AM    <DIR>          .
11/03/2011  09:08 AM    <DIR>          ..
11/03/2011  09:02 AM    <DIR>          lab1
11/03/2011  09:02 AM    <DIR>          lab2
11/03/2011  09:04 AM                16 s.txt
                                1 File(s)      16 bytes
                                4 Dir(s)  94,489,948,160 bytes free
```

```
E:\>move p.txt d:
        1 file(s) moved.
```

```
C:\Windows\system32\cmd.exe
E:\>copy d:\*.*
d:\ahmadi.txt
```



```
D:\>xcopy e:\os1 f:\1 /s/e
0 File(s) copied

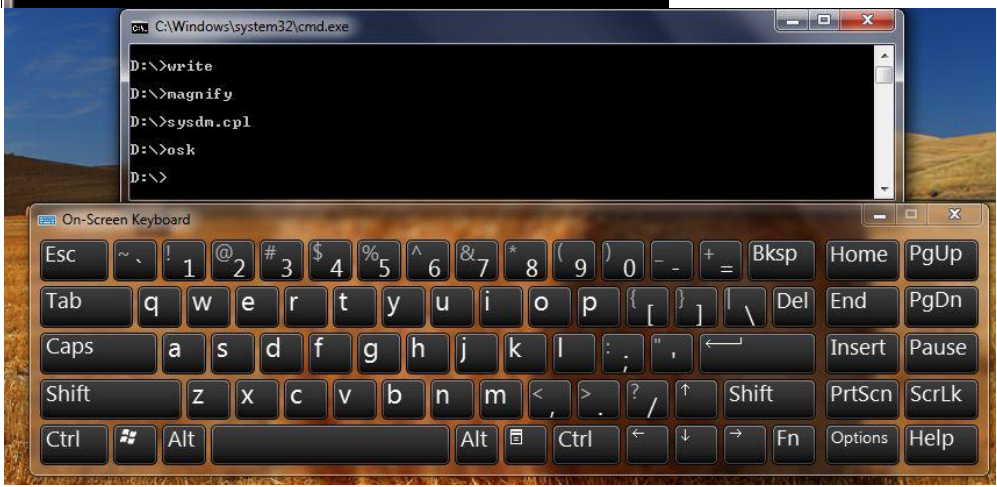
D:\>f:

F:\>cd 1

F:\1>dir
Volume in drive F has no label.
Volume Serial Number is 62EB-1D22

Directory of F:\1

11/03/2011  09:36 AM    <DIR>          .
11/03/2011  09:36 AM    <DIR>          ..
11/03/2011  09:36 AM    <DIR>          os2
                                0 File(s)      0 bytes
                                3 Dir(s)  37,620,932,608 bytes free
```





```
D:\>ipconfig

Windows IP Configuration

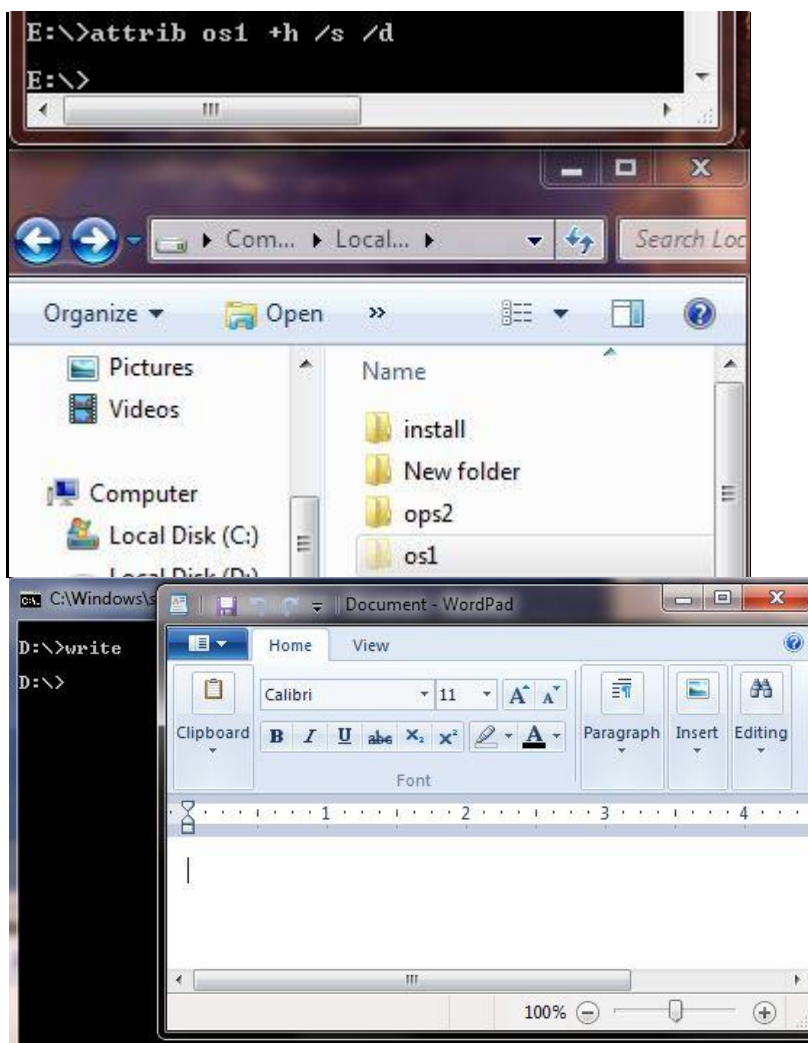
Ethernet adapter Local Area Connection:

    Connection-specific DNS Suffix  . : KHORASGAN.LOCAL
    Link-local IPv6 Address . . . . . : fe80::6c43:4d11:7c5:ef86%10
    IPv4 Address. . . . . : 192.168.0.93
    Subnet Mask . . . . . : 255.255.240.0
    Default Gateway . . . . . : 192.168.15.254
```

```
D:\>netstat

Active Connections

Proto Local Address           Foreign Address         State
TCP    192.168.0.93:49224       192.168.15.254:remote-winsoc ESTABLISHED
TCP    192.168.0.93:49438       192.168.15.254:remote-winsoc ESTABLISHED
TCP    192.168.0.93:49440       192.168.15.254:65421    CLOSE_WAIT
```



## آز سیستم عامل - دستور کار عملی



تمام مسائل زیر را به ترتیب در پنجره cmd انجام دهید.

گزارشی از پاسخ مسائل زیر تهیه نمایید و فرمانها و دستوراتی که برای انجام هر قسمت وارد و اجرا کرده‌اید را بنویسید:

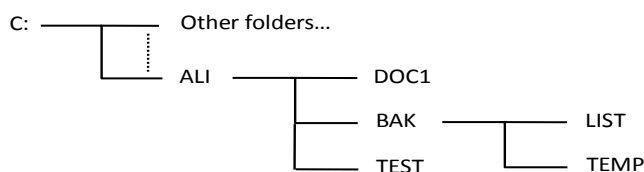
نکته: اگر با اجرای فرمانی پیغام خطا مشاهده گردید با استفاده از فرمان `HELP [command]` می‌توان فرمت و راهنمای دستور فوق را مشاهده نمود).

۱- به ریشه درایو C: رفته و پوشه‌ای به نام (و یا فامیل) خود ایجاد نمایید و تست نمایید پوشه موردنظر حتماً بر روی ریشه درایو C ایجاد شده باشد (پوشه ایجاد شده حتماً بایستی به اسم دانشجو ایجاد شود. البته در قسمت‌های بعدی فرض شده نام پوشه‌ای که به اسم دانشجو ایجاد شده است `yourname` باشد ولی شما در فرمان‌هایی که استفاده می‌کنید به جای `yourname` از اسم خودتان استفاده نمایید).

۲- کنترل نمایید آیا سیستم‌عامل ویندوز بر روی درایو C: و پوشه `WINDOWS` نصب شده باشد. (دقت شود ممکن است پوشه اصلی ویندوز با نام‌های `WINDOWS.0` یا `WINXP` یا ... باشد در صورتی که سیستم‌عامل ویندوز بر روی پوشه دیگری یا درایو دیگری نصب شده است در مسائل بعدی برای اجرای دستورات از نام جدید پوشه ویندوز استفاده شود).

۳- به پوشه ایجاد شده در مسئله ۱ که همانم با اسم خودتان ساخته‌اید رفته و سه پوشه به نام‌های `BAK`، `DOC1` و `TEST` بر روی آن ایجاد کنید؟ با استفاده از فرمان `DIR` و یا `TREE C:\yourname` اطمینان از ایجاد سه پوشه بر روی پوشه `yourname` حاصل شود).

۴- روی پوشه `BAK` رفته و دو پوشه دیگر به نام‌های `TEMP`، `LIST` ایجاد نمایید نتیجه اجرای فرمان `TREE C:\yourname` چیست؟ مثلاً اگر نام من `ALI` باشد درختی مشابه زیر مشاهده می‌گردد.



۵- بر روی پوشه `LIST` رفته و فرمان `COPY C:\*.*` اجرا شود چه تعداد فایل و با چه نام‌هایی در پوشه `LIST` کپی شده است؟ آیا فایل‌های مخفی نیز کپی شده است؟ با چه فرمانی می‌توان وجود فایل‌های مخفی که بر روی این پوشه کپی شده است را مشاهده کرد؟

۶- یک پوشه به عقب برگشته و بر روی پوشه `TEMP` بروید سپس کلیه فایل‌هایی که دارای پسوند `DLL` هستند و در پوشه `C:\WINDOWS\SYSTEM` موجود است را بر روی `TEMP` کپی نمایید. چه تعداد فایل کپی گردید؟ با استفاده از فرمان `TREE C:\yourname /F` از درخت ایجاد شده و فایل‌های موجود در هر پوشه مطمئن شوید؟

۷- نیمی از فایل‌های کپی شده در مسئله ۶ را به دلخواه خود به پوشه `TEST` انتقال (`MOVE`) دهید (از فیلترهایی مانند فایل‌هایی که با حرف `A` شروع می‌شود (`A*.*`) یا کلیه فایل‌های حداکثر ۵ حرفی (`*?????`) استفاده شود). نتیجه اجرای فرمان `TREE c:\yourname /F` چیست؟ چگونه درخت

موردنظر را در گزارش خود کپی نمودید؟ چگونه می‌توان درخت را در یک فایل متنی یا تصویری ذخیره نمود؟

۸- با استفاده از فرمان **XCOPY** محتویات پوشه **BAK** را به‌طور کامل در پوشه **DOC1** کپی نمایید (از سوئیچ **/S** برای کپی پوشه‌ها و فایل‌های داخلی استفاده شود)؟ چگونه می‌توان اطمینان پیدا کرد که عمل فوق به‌طور کامل انجام و در پوشه **BAK** کپی شده باشد؟

۹- آیا فایلی با پسوند **BAT** در بین فایل‌های کپی شده در مسئله ۵ وجود دارد در صورت مثبت بودن جواب محتوای فایل موردنظر را با فرمان **TYPE** مشاهده نمایید سپس نام فایل فوق را به **TEST.BAK** تغییر دهید. اگر فایل **BAT** وجود ندارد و در هر صورت فرمان‌ها و عمل‌های فوق را بر روی یک فایل دلخواه دیگر انجام دهید؟ اگر فایل متنی نباشد نتیجه فرمان **TYPE** چیست؟

۱۰- سعی کنید پوشه **BAK** را با فرمان **RD** حذف کنید؟ آیا پوشه موردنظر حذف شد؟ چرا؟ سعی کنید محتویات پوشه **BAK** را با فرمان‌های **DEL** و **RD** حذف کنید و در نهایت پوشه **BAK** را حذف کنید چه تعداد دستور برای انجام این مرحله اجرا شد؟

(از فرمان **TREE C:\yourname [/F]** را برای صحت اطمینان از انجام کار استفاده نمایید).

۱۱- در پایان با استفاده از فرمان **DELTREE C:\yourname** پوشه‌ای که با نام خود در درایو **C:** ایجاد کرده‌اید و محتویات آن را به‌طور یکجا حذف نمایید اگر از این طریق نتوانستید پوشه فوق و محتویات آن را طبق روش مسئله ۱۰ حذف نمایید؟

۱۲- فرمان‌های **CLS**, **CHKDSK**, **TIME**, **DATE**, **ATTRIB** و کلیدهای **Alt-Enter**، **↑** و **↓** را تست و نتیجه اجرا را بنویسید؟

در علم کامپیوتر ماشین مجازی (Virtual machine) نرم‌افزاری است که بر روی یک کامپیوتر پیاده‌سازی می‌شود. این پیاده‌سازی به گونه‌ای است که تصور می‌شود یک کامپیوتر واقعی در حال اجرای برنامه‌های ماست.

یک ماشین مجازی، در ابتدا توسط Popok and Goldberg به صورت "یک نسخه کپی شده از روی یک ماشین واقعی، به صورت کارا و ایزوله شده" تعریف شد. استفاده‌های کنونی، ماشین‌های مجازی را شامل می‌شود که هیچ ارتباط با سخت‌افزار واقعی ندارند.

ماشین‌های مجازی، بر اساس استفاده و درجه ارتباط به ماشین واقعی، به دودسته اصلی تقسیم می‌شوند. یک ماشین مجازی سیستمی یک زیرساخت محاسباتی کامل را فراهم می‌کند که از اجرای یک سیستم‌عامل کامل پشتیبانی می‌کند. در مقابل، یک ماشین مجازی فرآیند، برای اجرای یک برنامه واحد طراحی شده، که این به این معناست که صرفاً یک از یک فرآیند خاص پشتیبانی می‌کند. یک ویژگی مهم یک ماشین مجازی، این است که نرم‌افزاری که درون آن در حال اجراست، با منابع و سطوح انتزاعی که توسط ماشین مجازی اعمال می‌شود، محدود شده است - یعنی نمی‌تواند از دنیای مجازی خود خارج شود.

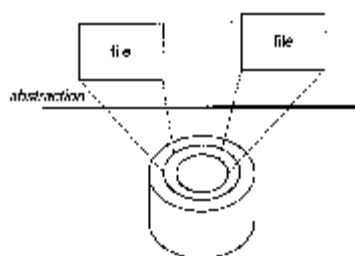
مثال: یک برنامه که به زبان جاوا نوشته شده است، با ارسال فرمان‌ها و دریافت نتایج موردنظرش، خدماتی از نرم‌افزار ماشین مجازی جاوا (JVM) می‌گیرد. با ارائه این خدمات به برنامه، نرم‌افزار جاوا، به عنوان یک ماشین مجازی عمل می‌کند.

کامپیوترهای مدرن امروزی جزء وسایل بسیار پیشرفته ساخته شده به دست بشر تلقی می‌گردند و دلیل اینکه ما توانستیم به چنین کاری نائل آییم، به احتمال قریب به یقین به خاطر دانش ما در مدیریت پیچیدگی‌ها می‌باشد. کامپیوترها از میلیون‌ها **چیپ** تشکیل شده‌اند که هر کدام از بیلیون‌ها ترانزیستور استفاده می‌کنند و تمام این قطعات به وسایل ورودی/خروجی و شبکه‌های مختلف متصل هستند تا در نهایت سیستم‌های نرم‌افزاری بتوانند بر روی تمام این‌ها سوار شده و کار نهایی را انجام دهند. نرم‌افزارهای متعدد گرافیکی، آموزشی و... نیز وجود دارند که کارهای ما را به انجام می‌رسانند.

کلید اصلی مدیریت پیچیدگی در سیستم‌های کامپیوتری همانا انجام کارها به صورت انتزاعی و قرار دادن ظواهری بین هر مرحله و مرحله بعد است تا کاربر اصلاً حس نکند که این میان چه اتفاقی در حال افتادن است و او در چه سطحی قرار گرفته است. این انتزاع باعث می‌شود که طراحی در مراحل بالاتر آسان شده و بدون توجه به مراحل پایین انجام شود. به عنوان مثال این حقیقت که هارددیسک به سکتورها و شیارهایی تقسیم می‌شود عملاً با وجود سیستم‌عامل برای یک نرم‌افزار بی‌معنا شده و نرم‌افزار تنها هارددیسک را به صورت یک محیط کاملاً قابل لمس و با یک سری فایل که هر کدام اندازه خاصی دارند می‌بیند. در این حالت یک

برنامه‌نویس به راحتی می‌تواند برنامه خود را بنویسد، بدون توجه به اینکه واقعاً در لایه‌های پایین‌تر چه دنیایی از پیچیدگی قرار دارد!

انتزاع خود دارای مراحل بی‌شماری است که از مراحل سطح پایین که عملاً با سخت‌افزار محض سروکار دارد شروع شده و به مراحل سطح بالا که مراحل انتزاعی نرم‌افزاری هستند ختم می‌گردد. در مراحل سطح پایین فقط سخت‌افزار و اجزاء فیزیکی وجود دارند. در مراحل سطح بالا اجزاء تشکیل‌دهنده همه نرم‌افزاری هستند و محدودیت‌های سخت‌افزاری را ندارند. در این کتاب ما بیشتر به سطوحی که در واقع جداساز بین مراحل سخت‌افزاری و نرم‌افزاری هستند می‌پردازیم، عملاً سطوح حد واسط بین سخت‌افزار و نرم‌افزار، از جایی که نرم‌افزار از سخت‌افزار جدا شده و عملاً ماشین فیزیکی که نرم‌افزار بر روی آن مشغول به کار است برای ما اهمیت پیدا می‌کند.



فایل‌ها در واقع انتزاعی از دیسک هستند. یک سطح از انتزاع در واقع ظاهر ساده‌تری را برای کاربران بالایی خود فراهم می‌آورد.

نرم‌افزار کامپیوتر توسط یک ماشین اجرا می‌شود (اصطلاحی که از اوان ایجاد کامپیوتر باب بوده است، امروزه معمولاً از کلمه پلتفرم بیشتر بجای ماشین استفاده می‌کنند). از دید سیستم‌عامل، یک ماشین عملاً از قطعات سخت‌افزاری همچون یک و یا بیشتر CPU و هم‌چنین مقداری RAM و وسایل ورودی/خروجی تشکیل شده است. منتها فراموش نکنید که کاربرد این واژه نسبی است، یعنی همان‌طور که گفته شد هرگاه سیستم‌عامل از کلمه ماشین استفاده کند، منظور آن اجزاء صرفاً سخت‌افزاری است، ولی هرگاه یک نرم‌افزار عادی از واژه ماشین استفاده کند، منظورش سیستم‌عامل مورد استفاده به همراه گوشه‌ای از جزئیات سخت‌افزاری است که توسط لایه سیستم‌عامل جداسازی نشده است و نرم‌افزار باید مستقیماً با آن‌ها درگیر باشد.

به‌اندازه کافی در مورد انتزاع بحث کردیم، حال اجازه دهید در مورد آن فاکتور دوم که مدیریت پیچیدگی را آسان می‌کند صحبت کنیم: قرار دادن ظواهری آسان بین هر مرحله از انتزاع. وجود چنین ظواهری باعث می‌شود که بتوان عملاً مراحل طراحی یک کامپیوتر را توسط چندین تیم، مثلاً یک تیم نرم‌افزاری و یک تیم سخت‌افزاری انجام داد، بدون آنکه آن‌ها چندان با کار یکدیگر کاری داشته باشند. وجود جدول دستوری CPU یک نمونه از این ظاهرسازی است. به‌عنوان مثال طراحان AMD و یا Intel ریزپردازنده‌هایی را طراحی می‌کنند که از جدول دستوری استاندارد IA-32 استفاده می‌کند. از آن طرف کامپایلر نویسان RedHat و یا مایکروسافت نیز کامپایلرهایی می‌نویسند که دستورات را به این استاندارد دیکد کند. اگر هر دو این گروه‌ها کار

خود را به درستی انجام دهند، نرم افزار کامپایل شده بر روی هر ماشینی که از استاندارد IA-32 پشتیبانی کند به درستی و تمام و کمال اجرا می شود. ظاهر موجود در سیستم عامل نیز مثال خوب دیگری است که می توان به آن اشاره کرد. هر سیستم عامل با داشتن مجموعه دستورات خود عملاً ظاهری را برای برنامه نویسان برنامه های مختلف فراهم می کند که آن ها تنها می توانند با صدازدن آن توابع کار خود را به راحتی انجام دهند، بدون توجه به اینکه واقعاً چه اتفاقی در پایین در حال رخ دادن است. حتی ممکن است کل ساختار یک سیستم عامل در طی سالیان عوض شود و با سخت افزارهای متعددی کار کند، ولیکن او خود مسئول بروز نگاه داشتن توابع خود است و تا زمانی که این کار را انجام دهد، تمام برنامه هایی که حتی چند سال پیش نیز برای آن نوشته شده بودند، کماکان به کار خود ادامه خواهند داد. حال اگر چندین سیستم عامل از یک مجموعه دستورات پیروی کنند، عملاً نرم افزار نوشته شده برای یکی از آن ها بر روی بقیه نیز به راحتی کار خواهد کرد.

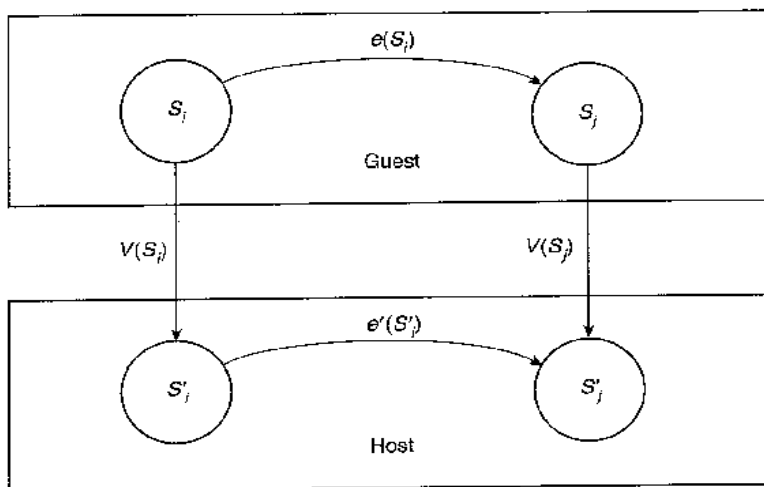
برخلاف تمام مزایایی که این ظواهر فراهم می کنند، آن ها خود عامل محدودیت هستند. برنامه هایی که برای یک مجموعه دستورات سیستم عاملی نوشته شده اند برای سیستم عامل دیگر کار نمی کنند. تنها استاندارد IA-32 وجود ندارد و CPU های دیگری نیز هستند که برای استانداردهای دیگر نوشته شده اند و این یعنی که سیستم عامل های طراحی شده برای IA-32 بر روی آن ها کار نخواهند کرد و این باعث به وجود آمدن سیستم عامل های مختلف (مثلاً ویندوز و لینوکس) می گردد. به عنوان یک قانون کلی باید گفت که وجود تنوع در مجموعه دستورالعمل ها و قوانین و کلاً ظواهر مختلف باعث به وجود آمدن سیستم عامل های مختلف و طبعاً برنامه های مختلف می گردد که این خود باعث ابداعات بسیار و جلوگیری از رکود فکری می شود؛ ولیکن در عمل این خود بزرگ ترین عامل محدود کننده ارتباط بین سیستم های مختلف است، مخصوصاً در دنیای شبکه شده امروزی که دوست داریم نرم افزارها را به همان آسانی که داده های خود را منتقل می کنیم، از سیستمی به سیستم دیگر منتقل کرده و با آن ها کار کنیم.

حتی پایین تر از لایه هایی که ارتباط بین سخت افزار و نرم افزار را فراهم می کنند، خود لایه های سخت افزاری مشکل ساز بوده و عملاً محدودیت هایی را حتی در مراحل بالاتر برای ما فراهم می کنند. امروزه با ظهور سیستم عامل های پیشرفته و زبان های سطح بالا، عملاً کاربران و برنامه نویسان چندان با مشکلات محض سخت افزاری دست و پنجه نرم نمی کنند و اصلاً از آن پایین ها خبر ندارند، ولیکن در معدود مواردی بازهم این مشکلات چهره کریه خود را نمایش می دهند و کاربر/برنامه نویس سطح بالای ما را با مشکل روبرو می کنند. نباید فراموش کنیم که بسیاری از سیستم عامل ها صرفاً برای معماری خاصی از CPU ها طراحی شده اند، بدین معنا که تمام سخت افزار موجود در ماشین تنها از طریق سیستم عامل نصب شده قابل دسترسی است که این خود دو نوع مشکل را ایجاد می کند: اول اینکه باید در عمل سیستم عاملی را انتخاب کنیم که اکثر برنامه های مورد علاقه ما را پشتیبانی کند، در واقع بتواند ارتباط آن ها را با سخت افزار ماشین مورد بحث برقرار نماید، حال تکلیف برای آن سری دیگر از نرم افزارهای بسیار خوبی که ممکن است در سیستم عامل های دیگری که ماشین ما را پشتیبانی نمی کنند باشد و ما با کار کردن به آن ها خو گرفته ایم چیست؟ و مشکل دوم که از اولی نیز بزرگ تر است این است که ما تمام ماشین خود را وقف یک سیستم عامل کرده ایم و



در صورتی که این سیستم عامل به هر دلیلی خراب شود، حتی اگر سخت افزار ما نیز سالم باشد (که معمولاً هست)، دیگر اصلاً نمی توانیم از ماشین استفاده کنیم!

مجازی سازی روشی را برای خلاص شدن از شر این قوانین که هر روز هم بیشتر می شوند فراهم می کند. هنگامی که یک سیستم و یا زیرسیستم، مثلاً یک CPU و یا هارددیسک مجازی سازی می شوند، ظواهر مربوط به آنها و تمامی منابعی که از طریق آنها ظواهر قابل دسترسی هستند به ظواهر و منابعی از سیستم واقعی که در حال کار است نگاشت می شوند. اگر بخواهیم مجازی سازی را به صورت رسمی تعریف کنیم، باید گفت که مجازی سازی عبارت است از یک هم ریختی [6] که یک سیستم مهمان [7] را به یک سیستم میزبان [8] نگاشت می کند (بر اساس تعریف Popok و Goldberk در سال 1974). این هم ریختی که در شکل زیر نشان داده شده است، حالت مهمان را به حالت میزبان نگاشت می دهد (تابع  $V$  در شکل) و به ازاء هر سری از عملیات،  $e$ ، که باعث تغییر حالت در مهمان شود (تابع  $e$  حالت  $S_i$  را به  $S_j$  تبدیل می کند) عملیات مشابهی،  $e'$ ، در میزبان انجام می شود که  $S'_i$  را به  $S'_j$  می برد. هر چند که این هم ریختی عملاً می تواند نشان دهنده انتزاع نیز باشد، ولی ما در اینجا بین انتزاع و ظواهر فرق می گذاریم، بدین صورت که مجازی سازی عملاً باعث بیشتر و یا کمتر انتزاعی تر شدن سیستم نشده و عملاً مقدار انتزاع بین ماشین مجازی و واقعی یکسان است.

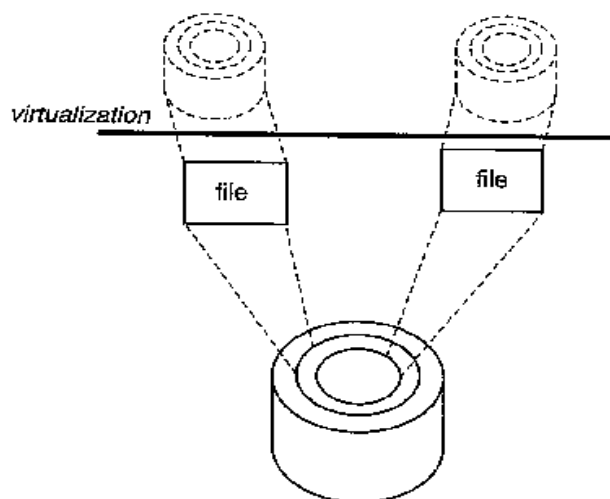


تعریف علمی مجازی سازی همانا ساخت یک هم ریختی بین سیستم مهمان و میزبان می باشد

اجازه دهید مثالی عملی از این مورد را بازگو کنیم. همان مثال هارددیسک معروف را در نظر بگیرید. اگر بخواهیم یک هارددیسک واقع در سیستم عامل را به چندین هارددیسک مجازی تقسیم کنیم، به ازای هر هارددیسک مجازی یک فایل بزرگ بر روی هارد واقعی تعریف می کنیم. هارددیسک های مجازی نیز برای خود سکتور و قطاع مشخصی را دارند، هر چند که اندازه آنها به اندازه هارد سیستم میزبان نمی باشد. برنامه مجازی سازی عمل نگاشت بین هارددیسک واقعی و هارددیسک های مجازی را فراهم می آورد (تابع  $V$  در هم ریختی). عمل نوشتن درون هارددیسک مهمان (تابع  $e$  در هم ریختی)، در عمل باعث ایجاد تقاضایی برای

نوشتن در هارددیسک میزبان می‌گردد (تابع *e* در هم‌ریختی) در این مثال عمل نوشتن بر روی هارددیسک‌های مجازی با تغییر سکتور و قطاع سروکار داشت و از این لحاظ انتزاعی در مقایسه با حالت واقعی شکل نگرفت.

مفهوم نه‌تنها می‌تواند در مورد اجزاء مختلف کامپیوتری عملی شود، بلکه می‌تواند یک کامپیوتر کامل را نیز شبیه‌سازی نماید. به‌عنوان مثال ماشین مجازی نصب‌شده بر روی یک کامپیوتر *Apple* می‌تواند یک ویندوز ۳۲ بیتی را که می‌شود نرم‌افزارهای ویندوزی را نیز درون آن نصب کرد، شبیه‌سازی کند. عملاً یک ماشین مجازی می‌تواند محدودیت‌های سخت‌افزاری و نرم‌افزاری یک ماشین واقعی را پشت سر گذاشته و درجه بیشتری از راحتی و جابجایی را برای نرم‌افزارها فراهم آورد.



پایه‌سازی دیسک‌های مجازی. مجازی‌سازی می‌تواند منابع و یا رابط‌های متفاوتی را در یک درجه از انتزاع فراهم آورد.

انواع مختلفی از ماشین‌های مجازی وجود دارد که می‌توان به‌صورت هم‌زمان آن‌ها را بر روی یک ماشین نصب کرد و این ماشین‌های مجازی برای هر کاربر محیط دلخواه او را شبیه‌سازی کنند. حتی می‌توان چندین ماشین مجازی را بر روی چندین ماشین واقعی در محیط کار نصب نمود که این کار درصد امنیت را نیز بالا می‌برد. می‌توان یک سرور قوی با چندین پردازشگر را عملاً به چندین ماشین مجازی تقسیم نمود تا هم سرعت خود سرور پایین نیاید و هم اینکه از پردازشگرهای دیگر نهایت استفاده انجام شود.

ماشین‌های مجازی همچنین می‌توانند از تکنیک‌های *emulation* برای اجرای یک برنامه بر روی سیستم‌های مختلف استفاده کنند. به‌عنوان مثال سیستمی که از دستوران *PowerPC* استفاده می‌کند می‌تواند عملاً به قسمی *emulate* شود که به نظر آید از دستورات *IA-32* استفاده می‌کند، بدین صورت می‌توان برنامه‌هایی که صرفاً بر روی *IA-32* اجرا می‌شوند را درون این سیستم نیز اجرا نمود. این کار می‌تواند هم در سطح سیستم (همان *simulation*) و با شبیه‌سازی کل سیستم عامل انجام شود (مثلاً نصب ویندوز درون *Macintosh*) و هم در سطح برنامه و یا پردازش در حال انجام (به‌عنوان مثال اجرای *Excel* درون

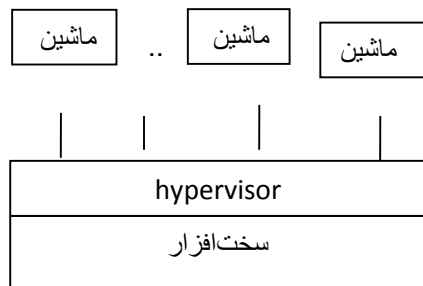
سیستم عامل های Sun Solaris و یا SPARC). علاوه بر emulation، ماشین های مجازی می توانند در هنگام اجرا به صورت آنی فایل های binary در حال کار را نیز بهینه کنند و سرعت پردازش ها را بالا ببرند.

ماشین های مجازی که در اینجا مثال زده شد تنها آنهایی هستند که برای کار با معماری ماشین های واقعی موجود طراحی شده اند. با این حال ماشین های مجازی وجود دارند که برای آن ها هیچ معماری فیزیکی واقعی وجود ندارد. امروزه دیگر برای طراحان زبان های برنامه نویسی امری بسیار عادی است که در ابتدا ماشین مجازی را طراحی کنند و زبان خود را به گونه ای بسازند که تمام دستورالعمل ها را به دستورالعمل هایی که برای آن ماشین مجازی قابل فهم است ترجمه کند. حال هر ماشین واقعی که این ماشین مجازی را درون خود نصب داشته باشد، به راحتی می تواند برنامه های نوشته شده برای آن زبان خاص را در خود اجرا نماید. قدرت این روش عملاً در زبان برنامه نویسی سطح بالای جاوا به همراه ماشین مجازی اش اثبات شده است.

گروه های مختلفی همچون برنامه نویسان، طراحان زبان و کامپایلر نویسان و طراحان سخت افزار در امر ماشین های مجازی سرمایه گذاری کرده و ماشین های مختلفی را می سازند. بدون توجه به نوع ماشین مجازی و کارهایی که انجام می دهد، یک سری تکنولوژی های مشترک بین تمامی این ماشین ها وجود دارد که برای ساخت یک ماشین مجازی، باید از آن فناوری ها استفاده کرد.

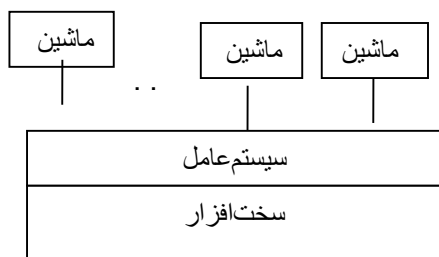
### انواع نصب ماشین مجازی

سه نوع نصب ماشین مجازی داریم:



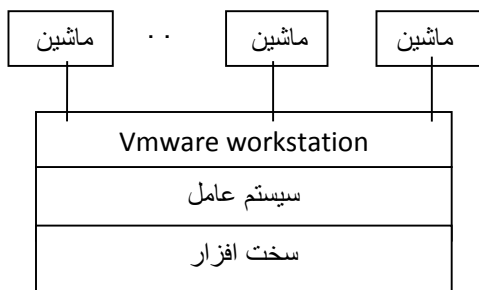
۱- hypervisor مستقیم روی سخت افزار نصب می شود و بدون هیچ واسطه ای در نتیجه فقط وقتی مشکلی برای سخت افزار ما پیش بیاید ماشین ما دچار مشکل می شویم پس یک ریسک داریم.

۲- در این روش روی سخت افزار یک سیستم عامل نصب است مثل winserver 2008 r2



یا vmserver که سرویس hyperv آن را فعال می کنیم و ماشین مجازی را میتوانیم نصب و کار کنیم پس ۲ تا ریسک داریم یکی سخت افزار خراب شود و دوم اینکه سیستم عامل ها مشکل پیدا کند ماشین های ما مشکل پیدا می کنند.

۳- در این روش چون ۳ لایه داریم ۳ ریسک هم داریم یکی سخت افزار ما خراب شود دومی



سیستم عامل host یا میزبان ما مشکل پیدا کند و سومی نرم افزار ماشین مجازی که ما نصب کردیم مشکل پیدا کند.

ما ۳ نوع انتقال اطلاعات بین ماشین‌های مجازی یا vm ها و host که سیستم‌عامل میزبان است داریم :

۱-ارتباط ماشین‌های مجازی باهم

۲- ارتباط ماشین‌ها باهم و با host

۳-ارتباط ماشین‌ها با دنیای بیرون

که می‌توانیم همه این ارتباط‌ها را خیلی راحت با دادن یک ip به ماشین و سپس از ماشین log out و بعد remot شدن از سیستم‌عامل اصلی یا host و تایپ ip انجام دهیم وصل می‌شویم

یا راست کلیک روی فایل سیستم‌عامل اصلی و انتخاب گزینه share و سپس در run ماشین تایپ [address\ip](#)

و یا اگر ویندوز ما سرور باشد از administrator tool گزینه srmvse و فعال کردن ftp انتقال‌ها به راحتی بین میزبان و ماشین حتی با drag & drop انجام می‌شود اگر لینوکس باشد سرویس sumba را add می‌کنیم ارتباط انجام می‌شود.

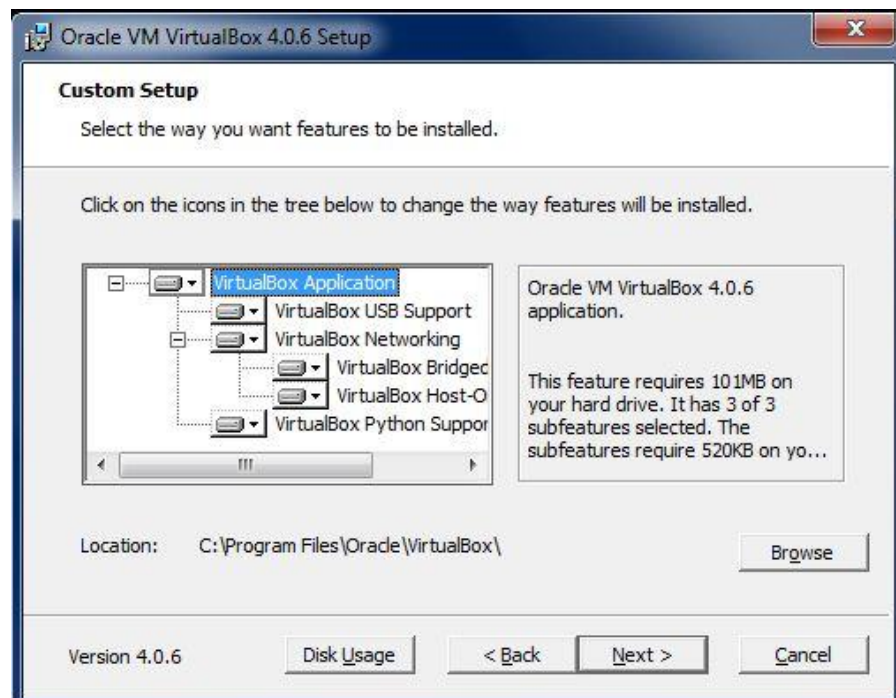
### Virtual box چیست؟

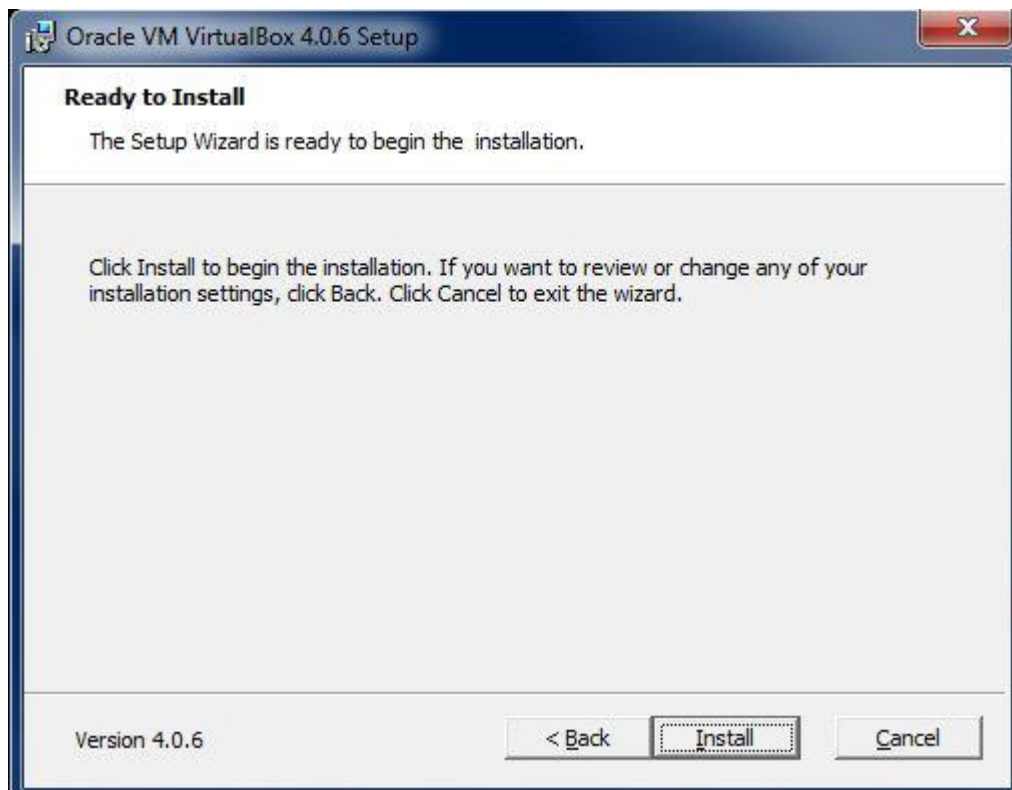
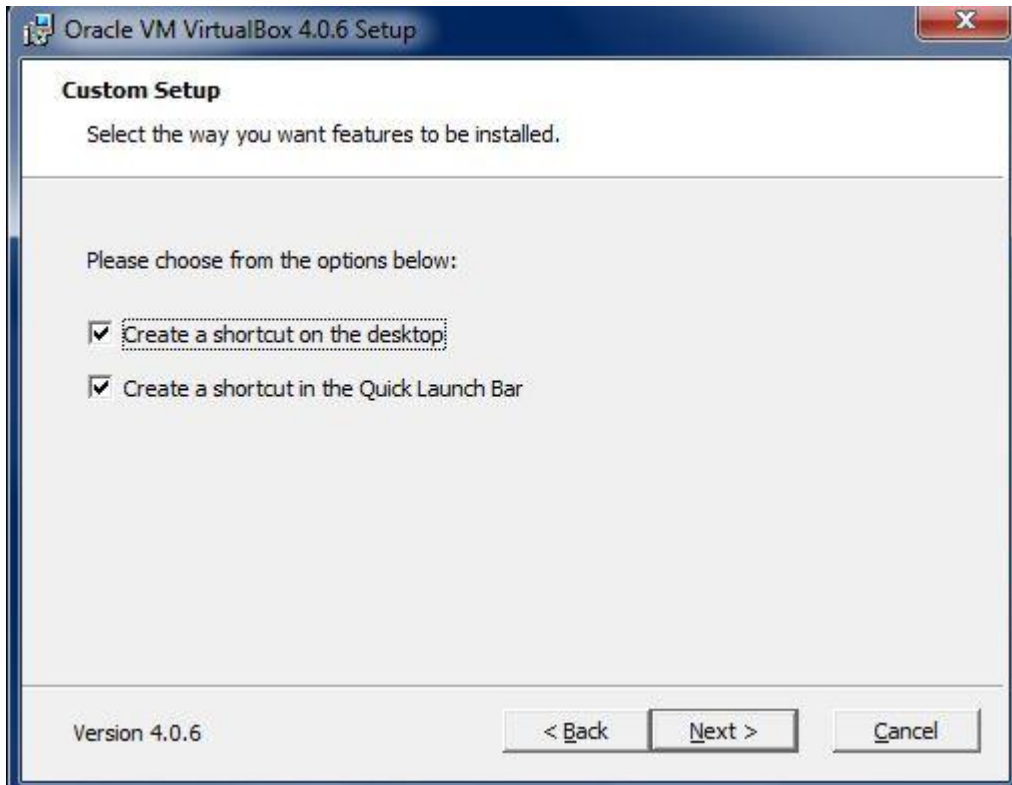
نرم‌افزاری است رایگان برای شبیه‌سازی یک سیستم کامل . در این برنامه بخش‌هایی از سیستم مانند کارت صدا و درگاه‌های USB سیستم با گزینش شما به صورت اشتراکی میان سیستم میهمان و میزبان (سیستم فعلی شما) استفاده می‌شوند . بخش‌هایی مانند دیسک سخت و کارت شبکه شبیه‌سازی می‌شوند، و در پایان بخش‌هایی مانند cd drive توانایی شبیه‌سازی و کاربرد اشتراکی را هم‌زمان دارا می‌باشند.

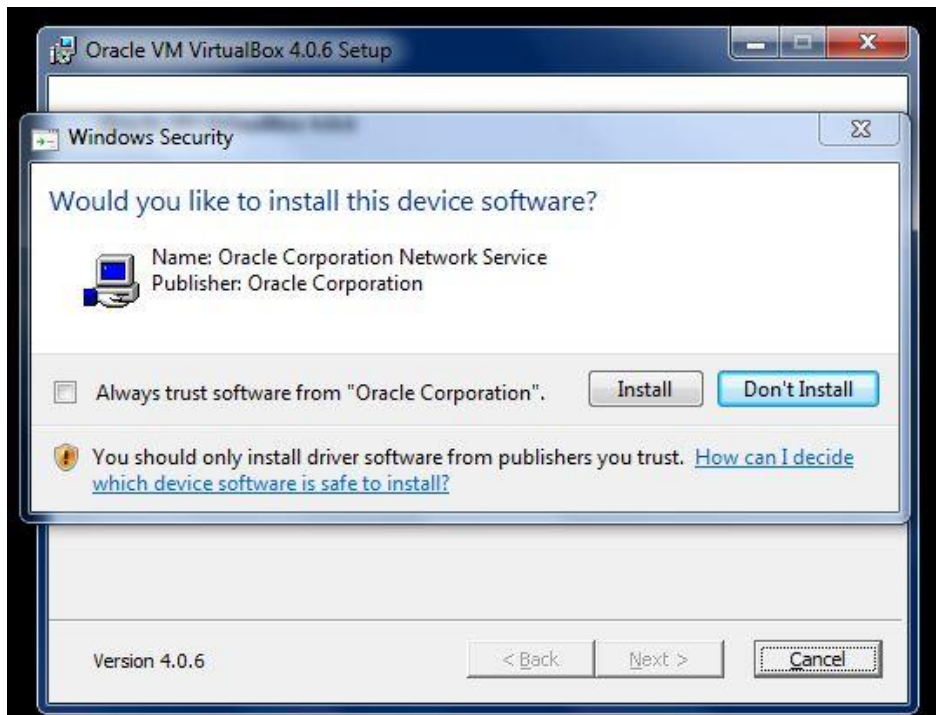
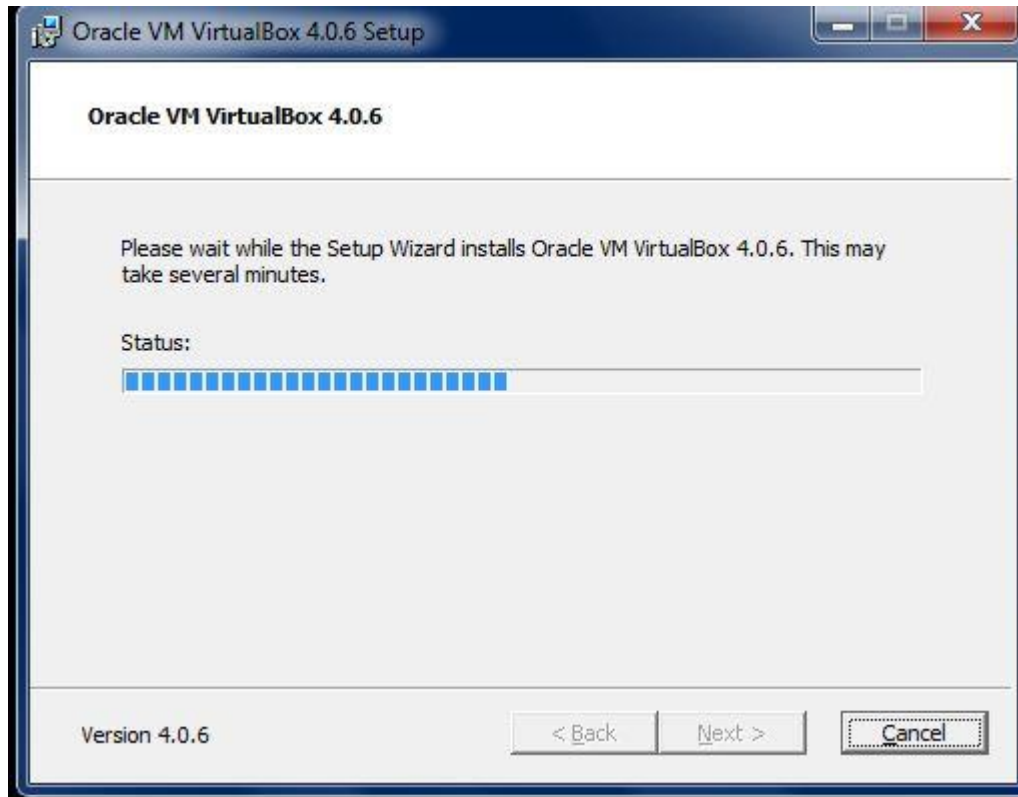
### Virtual box چه کاربردی دارد ؟

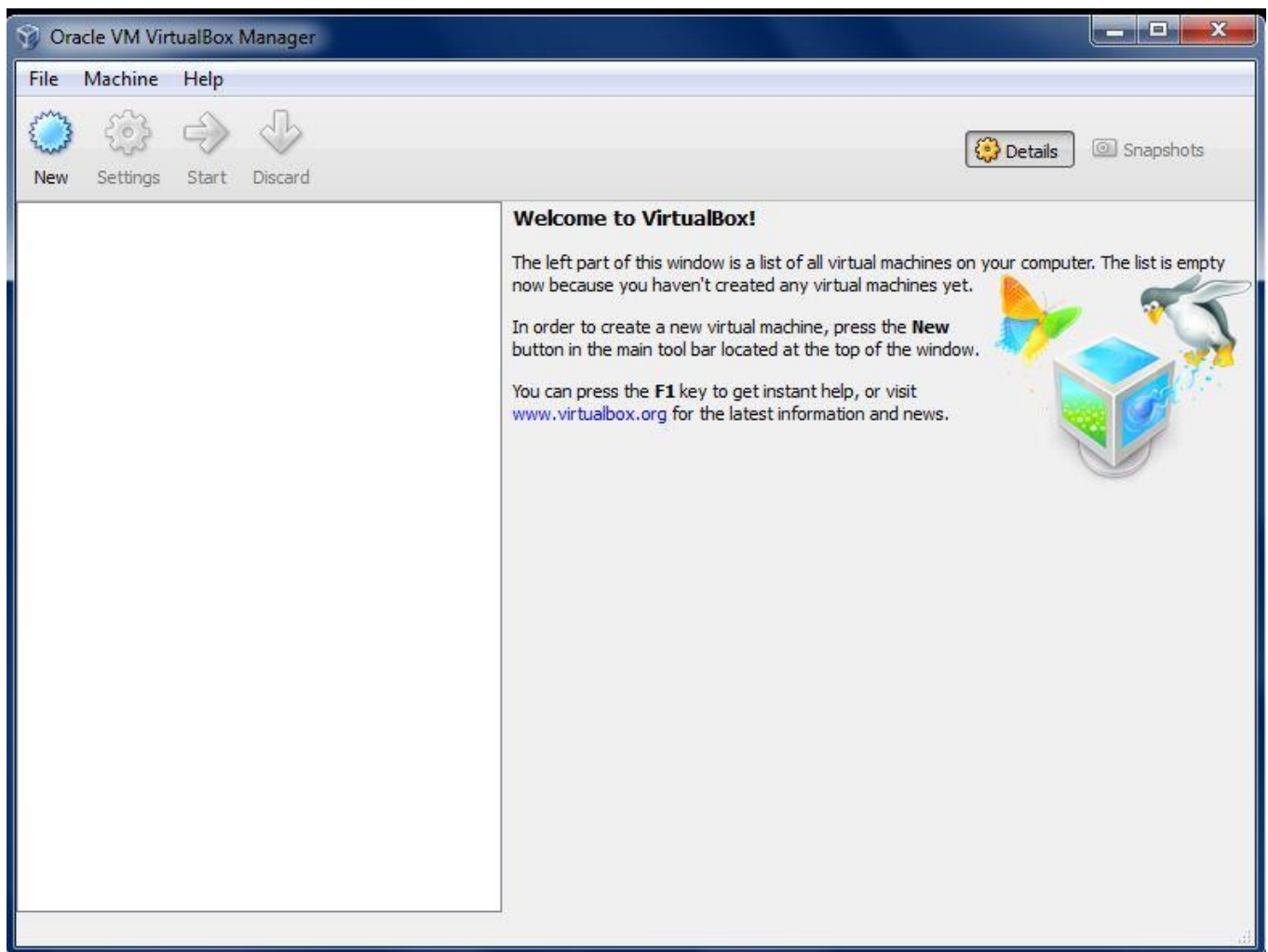
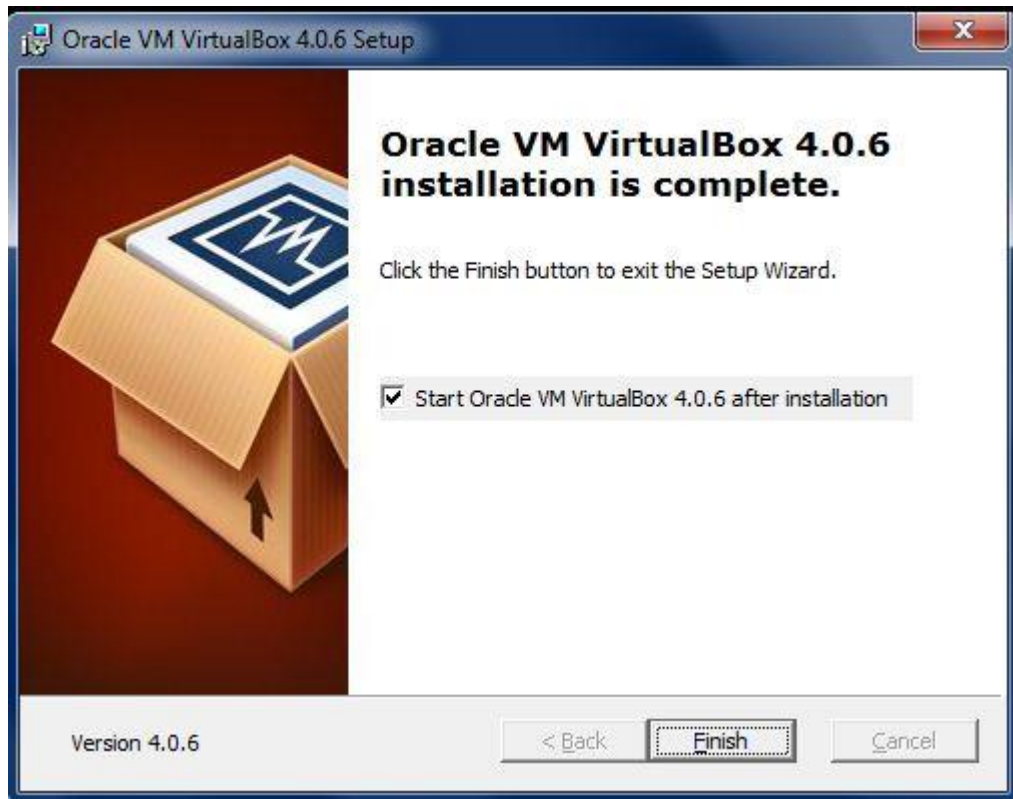
کاربردهای گوناگونی را می‌توان برای آن برشمرد که در فهرست زیر برخی از آن‌ها را مشاهده می‌کنید.

- نصب چندین سیستم عامل ناسازگار بر روی یک سیستم
- کاربرد آموزشی در زمینه‌های شبکه و برنامه‌نویسی شبکه
- کاربرد در زمینه‌ی برنامه‌نویسی برای گسترش برنامه‌های تحت شبکه
- کاربرد در زمینه‌ی برنامه‌نویسی برای آزمایش سازگاری برنامه در سیستم‌عامل‌های گوناگون
- اجرای برنامه‌های شک انگیز و اطمینان انگیز به داشتن ویروس
- ساخت برنامه‌های قابل حمل در سیستم‌عامل کاملاً دست‌نخورده





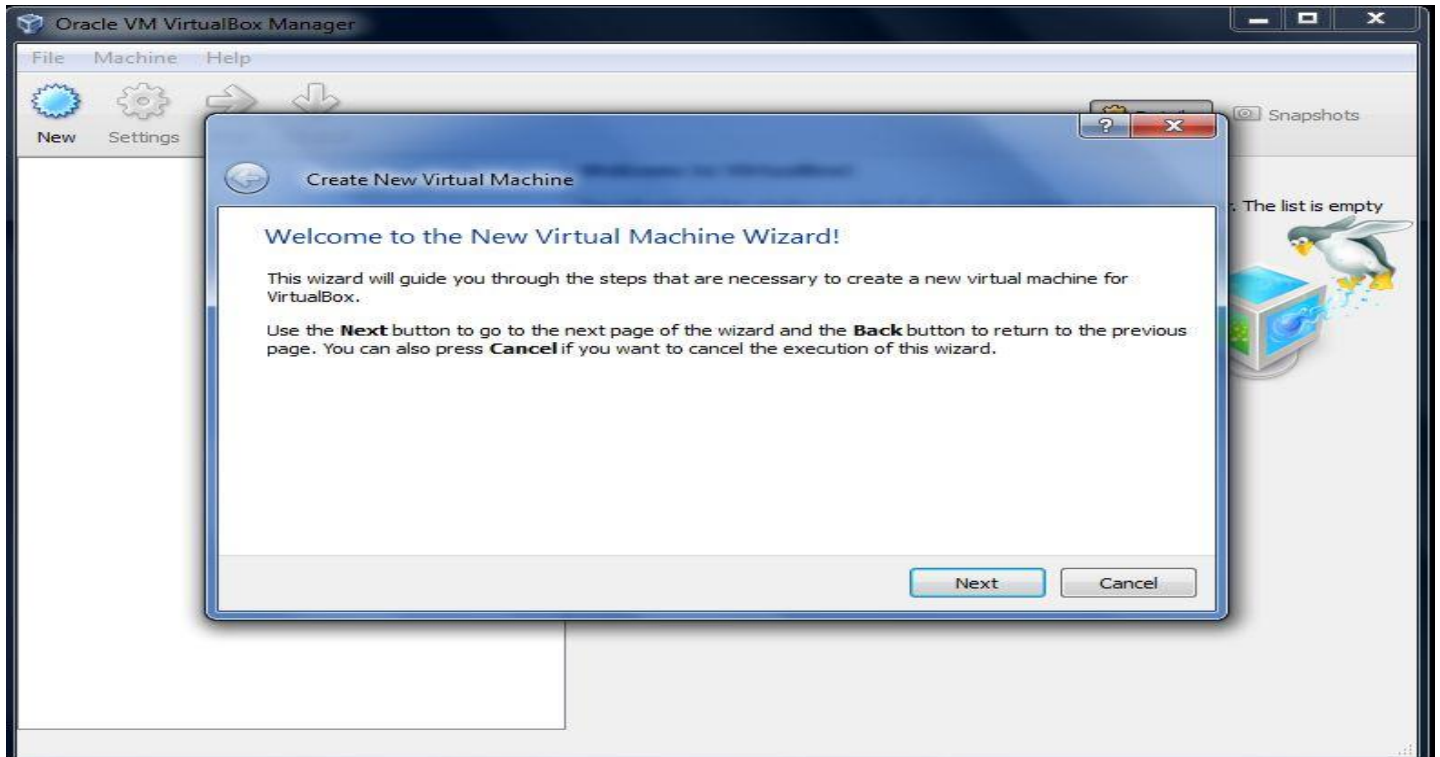


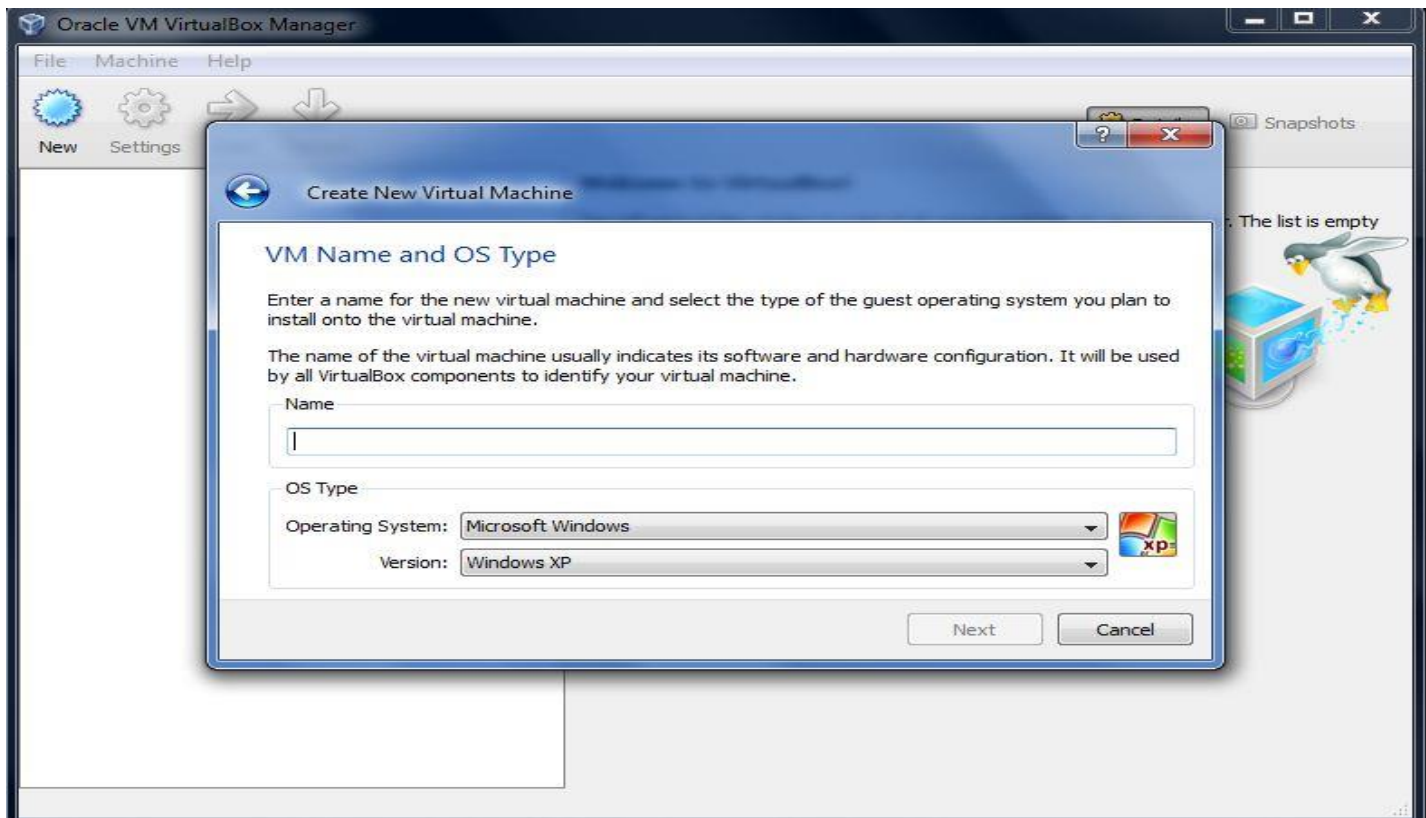




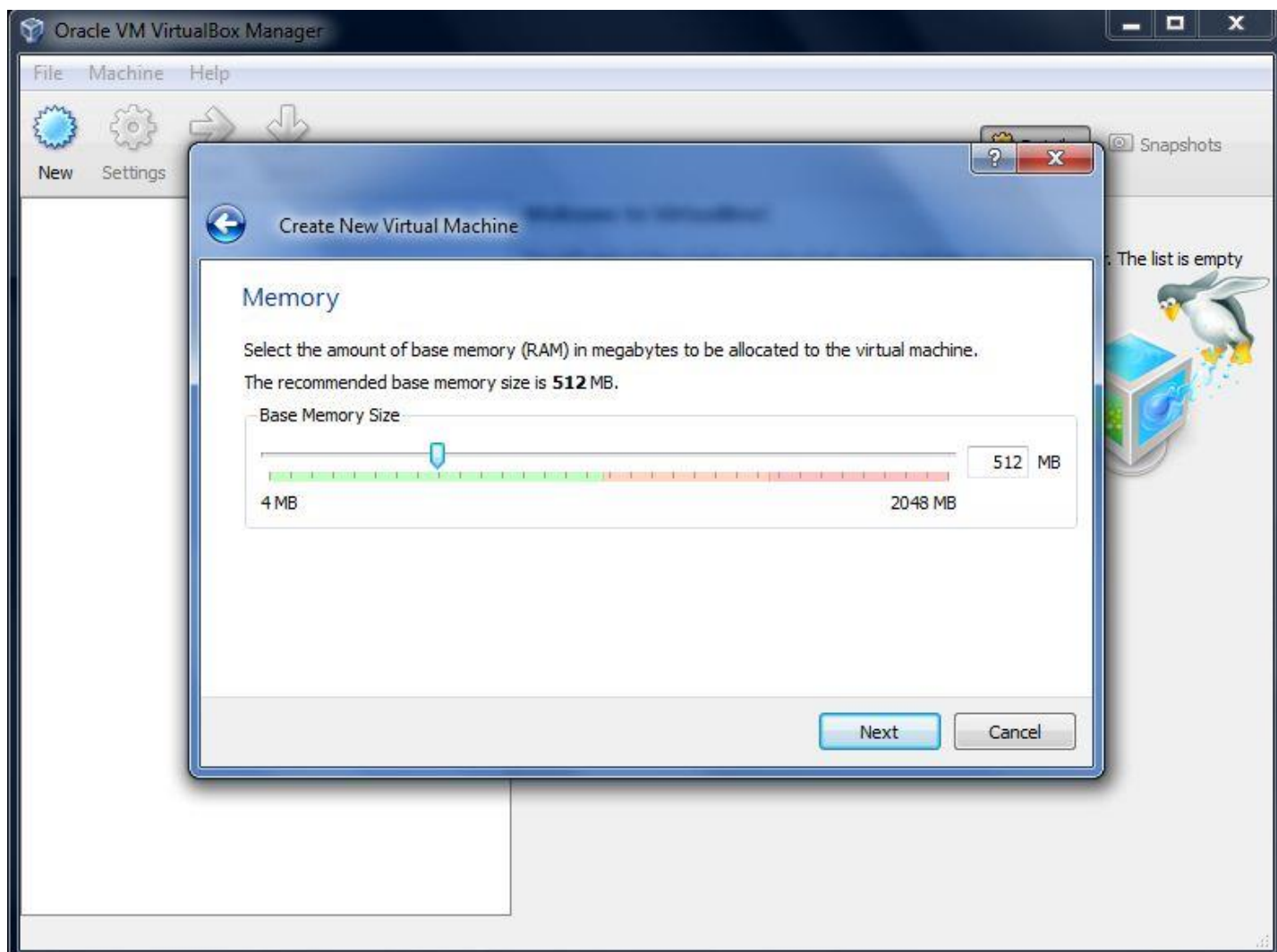
## چگونه می‌توان در **virtual box** یک سیستم‌عامل **linux suse** نصب کرد ؟

پس از پایان نصب **virtual box** ، برنامه را اجرا می‌کنیم گام نخست برای نصب یک سیستم‌عامل ، ساخت یک سیستم مجازی جدید است برای انجام این کار برنامه را اجرا می‌کنیم و بر روی کلید **new** کلیک کنید . در پنجره‌ای که باز می‌شود کلید **next** را بزنید.

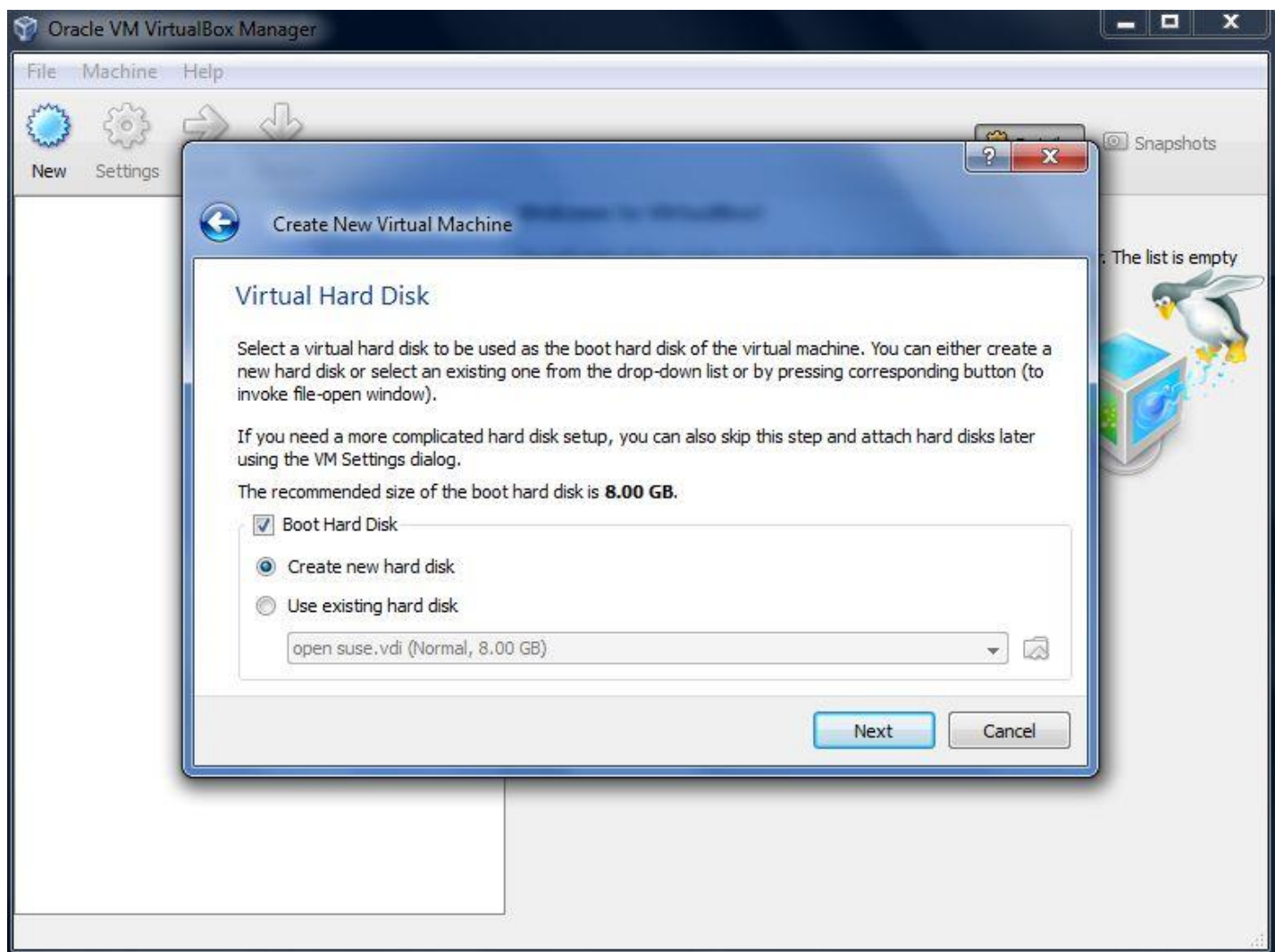




در جعبه متن قسمت **name** نامی برای سیستم مجازی جدید تایپ کنید و در بخش **os type** نوع سیستم عامل و ویرایش یا توزیع آن سیستم عامل را مشخص کنید.



در پنجره بالا انتخاب مقدار حافظه سیستم مجازی را انجام دهید با نگرش به گزینه‌ای که در بخش **os type** انتخاب کردید **virtual box** مقدار بهینه حافظه را به شما نشان می‌دهد بخش سبزرنگ دامنه مقدارهای ایمن و بخش قرمز رنگ ، دامنه نایمن را نشان می‌دهند . فراموش نکنید که هر چه مقدار حافظه انتخاب شده بیشتر باشد فضای خالی حافظه سیستم میزبان (سیستم عامل اصلی شما ) کاهش خواهد یافت . چنانچه مقدار حافظه را بیش از اندازه بالا ببرید ممکن است سیستم عامل از دیسک سخت به عنوان حافظه مجازی استفاده کند ( کاهش سرعت و افزایش فرسایش دیسک ) و یا ممکن است **virtual box** نتواند سیستم جدید را راه اندازی کند . بهتر است مقدار پیشنهادی را تغییر ندهید پس از ایجاد سیستم مجازی نیز در صورت نیاز می‌توانید این مقدار را تغییر دهید در پنجره بعد :



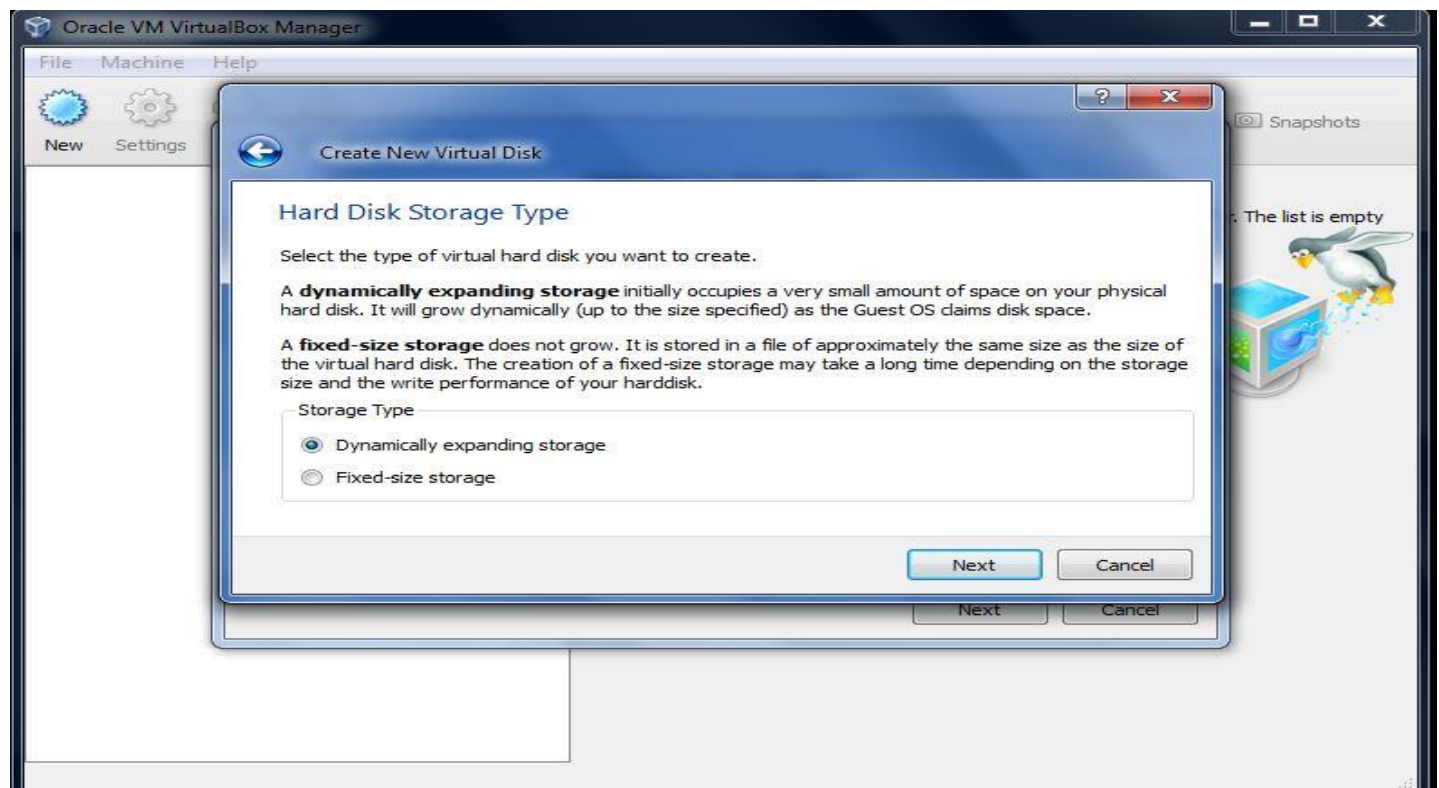
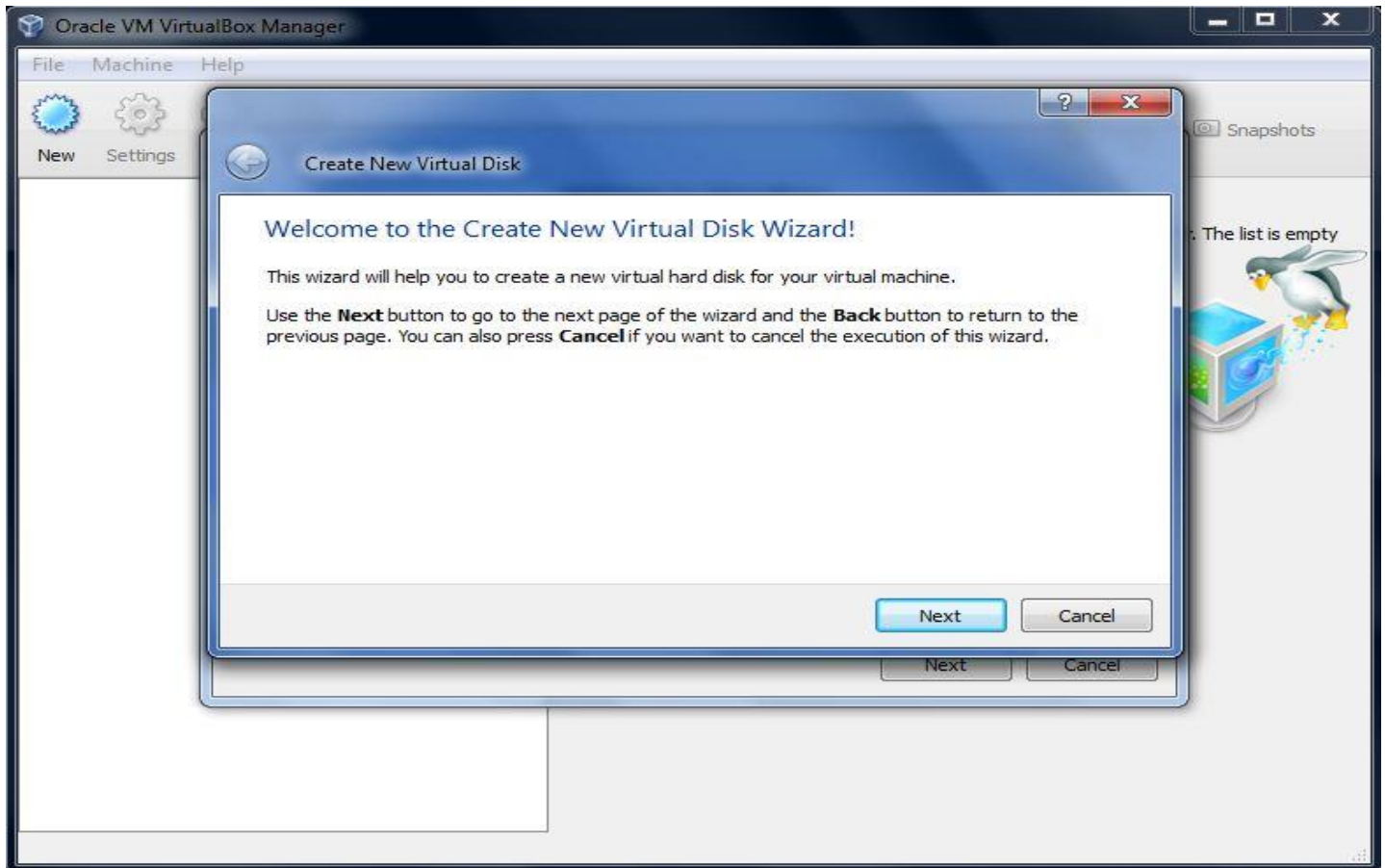
در پنجره بالا تنظیمات دیسک سخت مجازی سه گزینه پیش روی شماست :

۱- استفاده نکردن از دیسک سخت ( hard disk )

۲- ایجاد دیسک سخت جدید (مجازی )

۳- استفاده از دیسک سخت مجازی که پیش تر در اختیار داشته‌اید

برای ایجاد دیسک سخت جدید گزینه‌ها را دست‌نخورده باقی بگذارید و کلید **next** را بزنید



دیسک‌های مجازی در **virtual box** بر سه گونه هستند :

۱- دیسک مجازی بر روی فایل با حجم متغیر

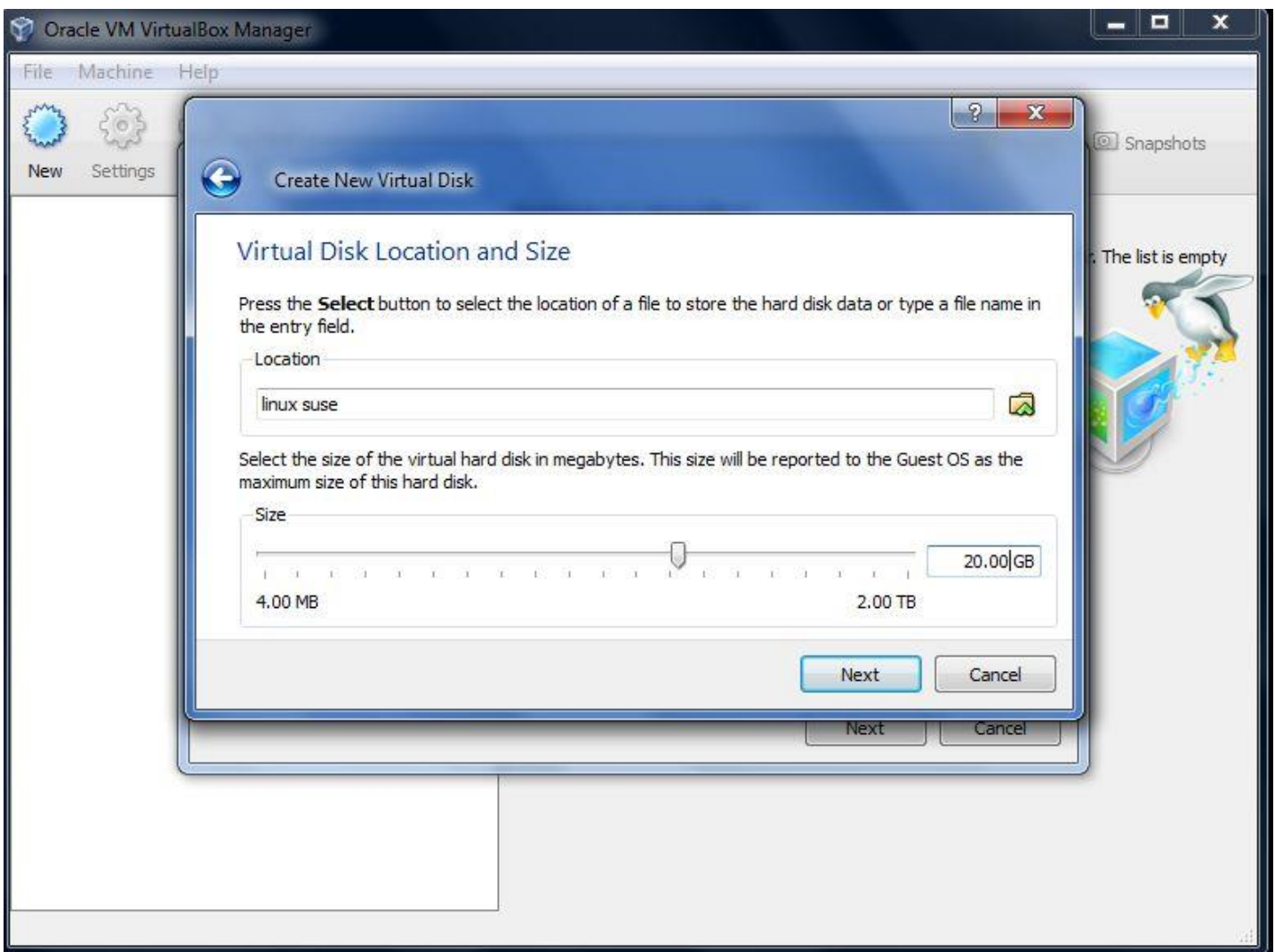
۲- دیسک مجازی بر روی فایل با حجم ثابت

۳- دیسک مجازی تغییرات

گونه دوم (فایل با حجم ثابت) در سنجش با گونه نخست (فایل با حجم متغیر) می‌تواند کارایی بهتری داشته باشد درحالی‌که گونه نخست در مدیریت فضای خالی در دیسک سخت به شما یاری می‌رساند.

باگذشت زمان و استفاده از فایل با حجم متغیر اندازه فایل به‌سوی اندازه واقعی دیسک میل می‌کند هرچند مقدار واقعی فایل‌ها کمتر از فضای دیسک باشد.

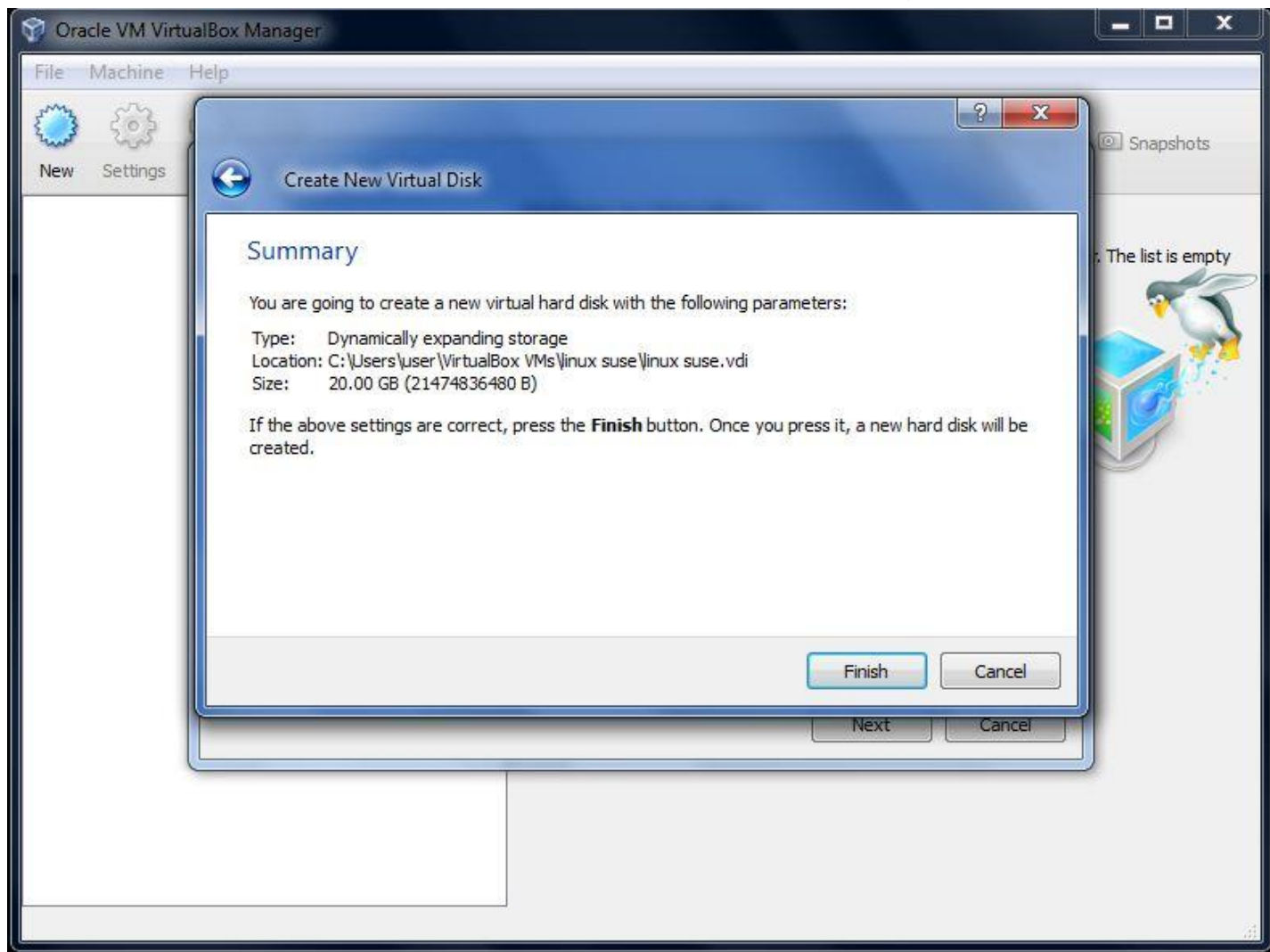
در پنجره بعد می‌توانید نام و حجم دیسک جدید را مشخص کنید.

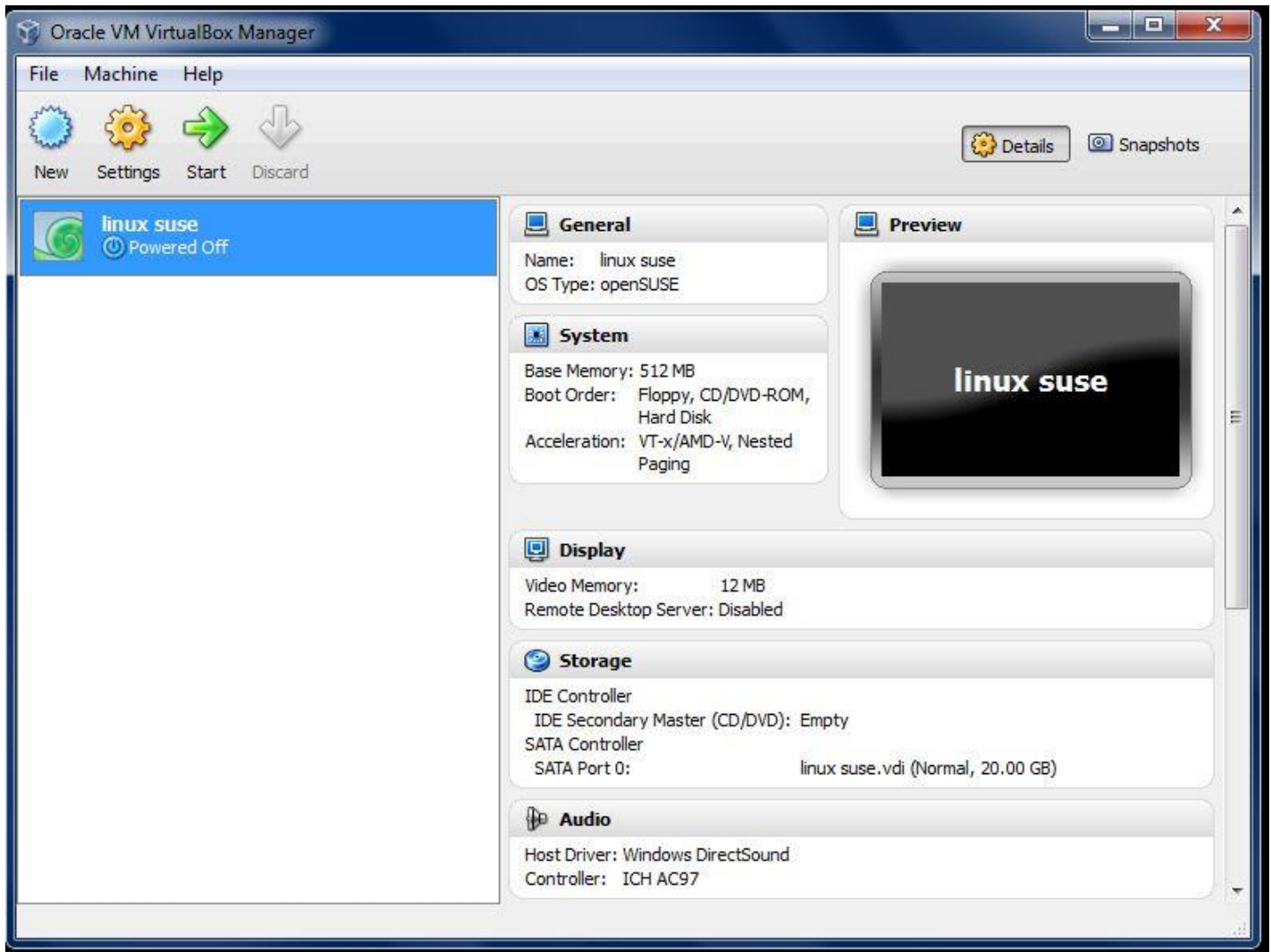


گزینه‌های پیش‌فرض در این پنجره نیز مانند گزینه‌های حافظه اصلی بر اساس سیستم‌عاملی که گزینش کرده بودید تعیین می‌شود چنانچه نمی‌دانید که چه مقدار فضا نیاز خواهید داشت مقدار را دست‌نخورده باقی بگذارید وگرنه کوشش کنید کمترین مقدار ممکن را وارد کنید.

با کلیک بر روی دکمه **next** به پنجره قبلی (ایجاد سیستم مجازی) باز می‌گردید.

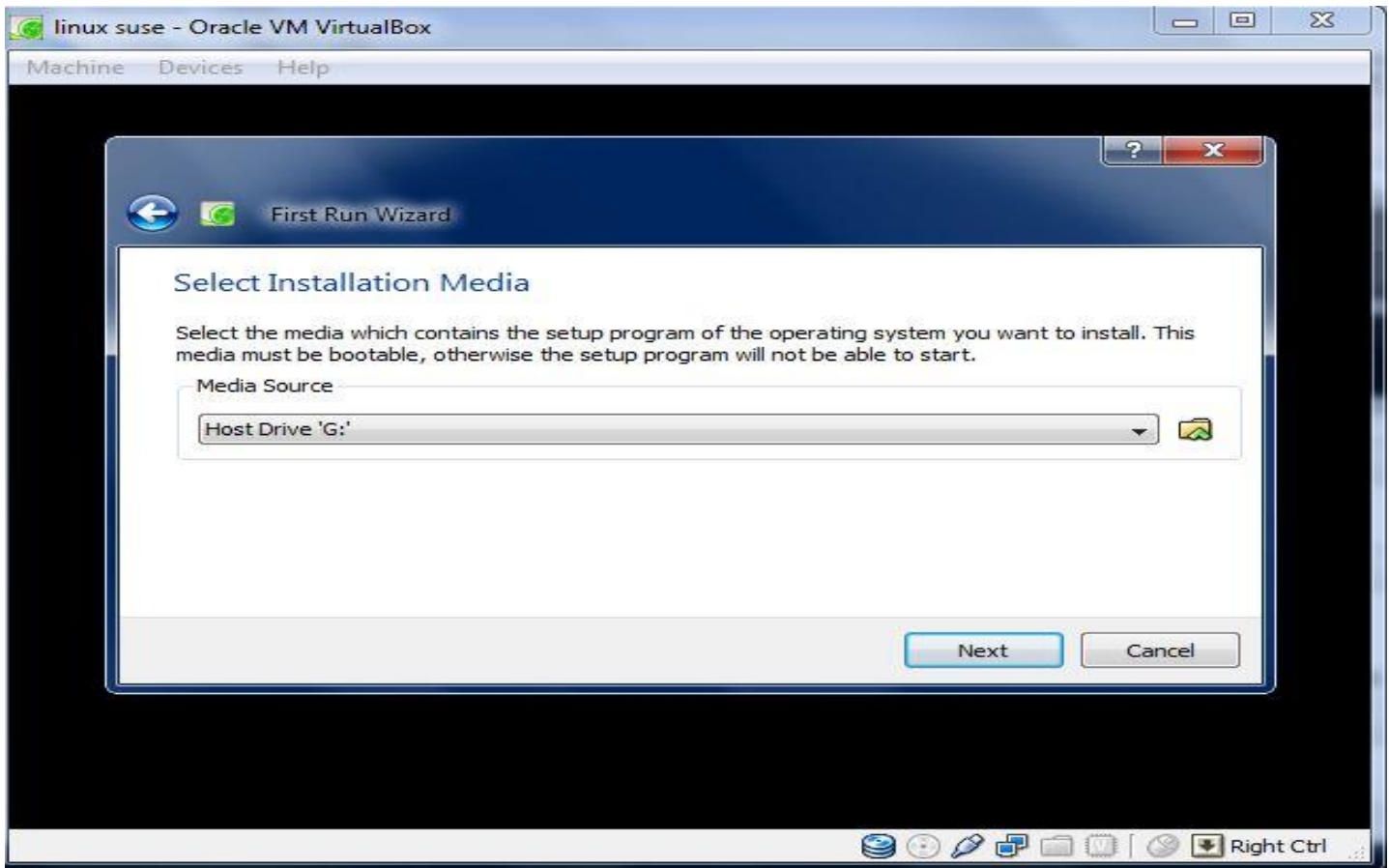
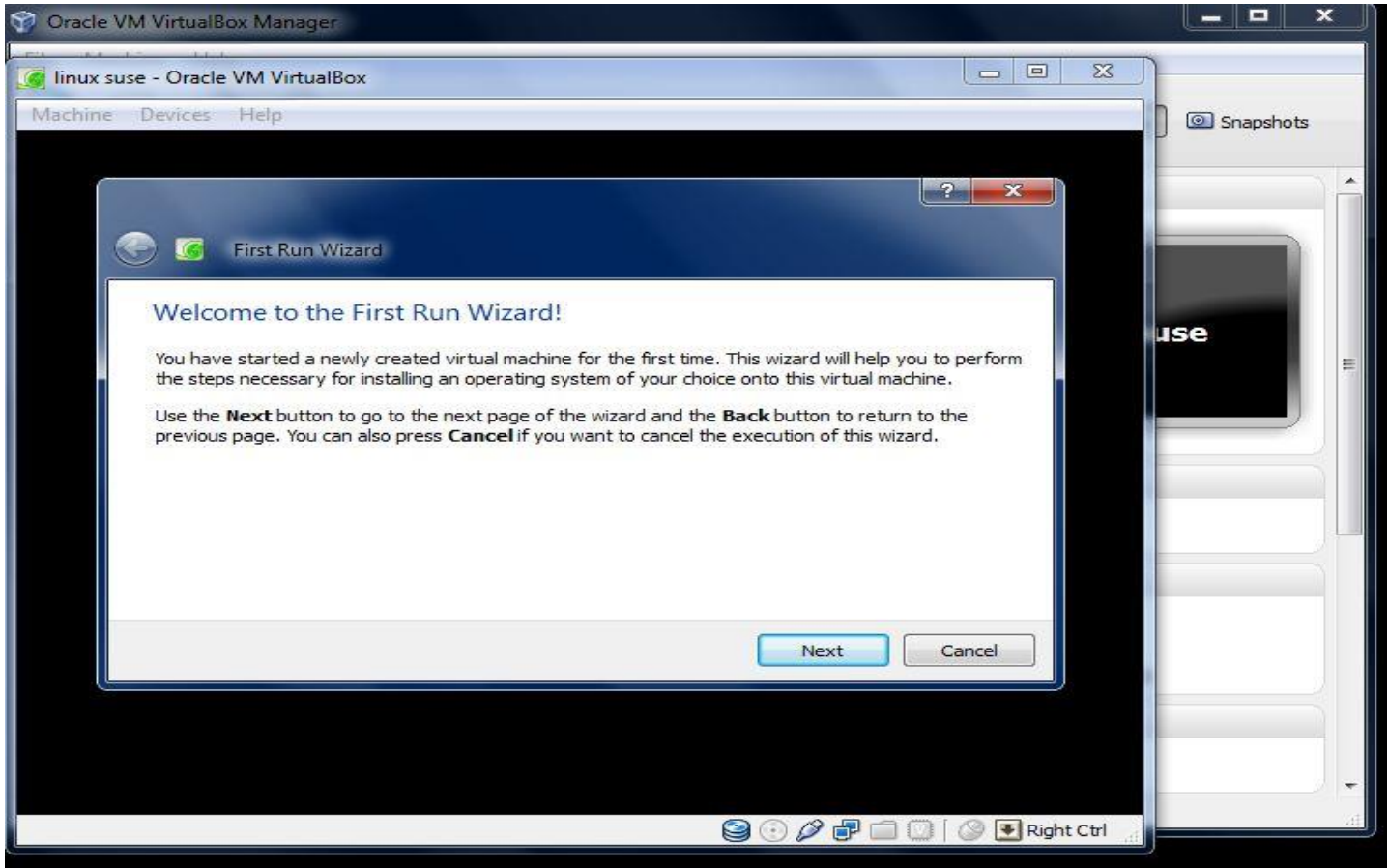
بر روی **finish** کلیک کنید تا سیستم مجازی شما ساخته شود.





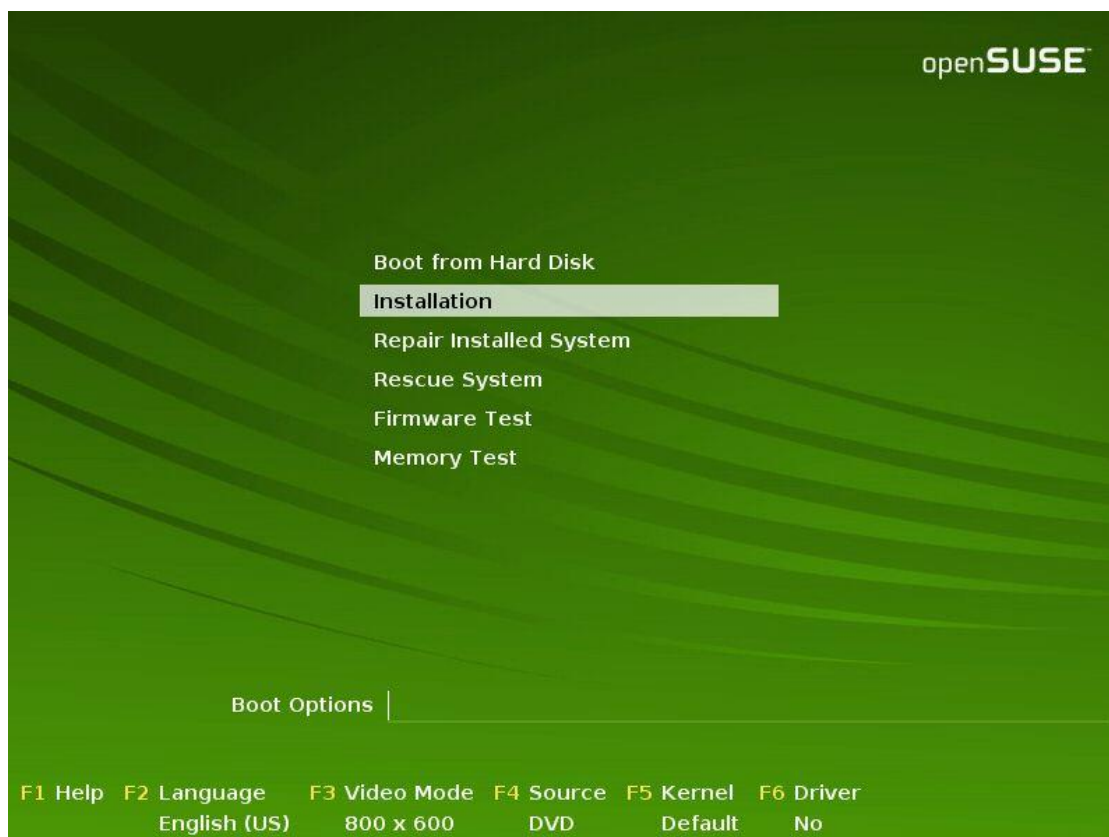
هم‌اکنون زمان آن است که سیستم‌عامل دلخواه را در سیستم مجازی جدید نصب کنید .  
با کلیک بر روی کلید **start** می‌توانید سیستم مجازی جدید را روشن کنید .

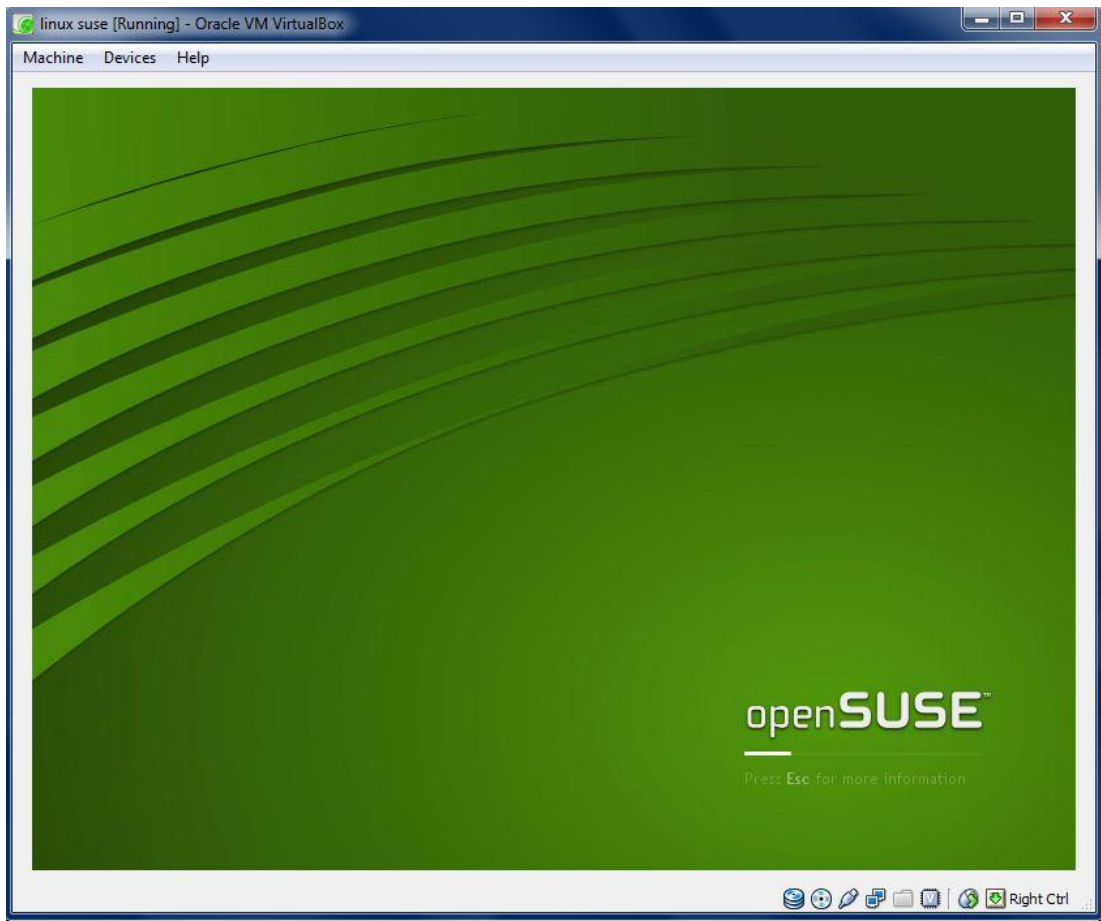
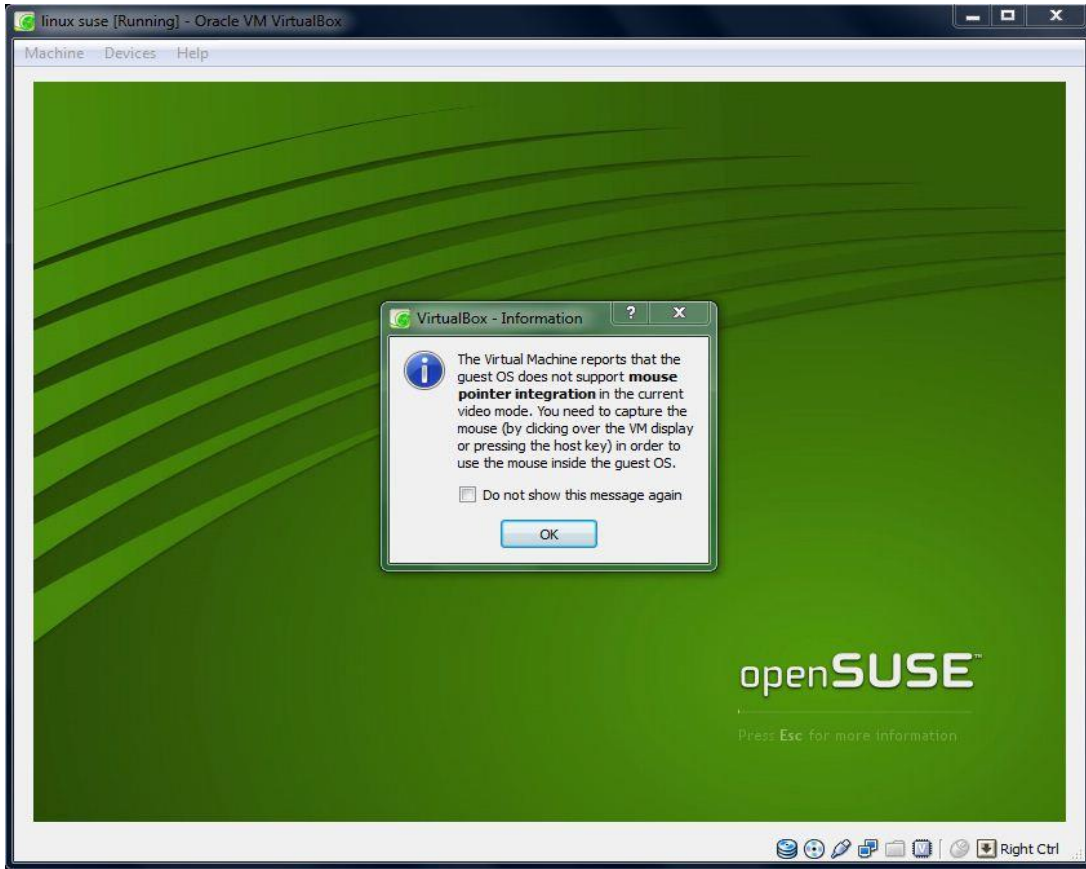


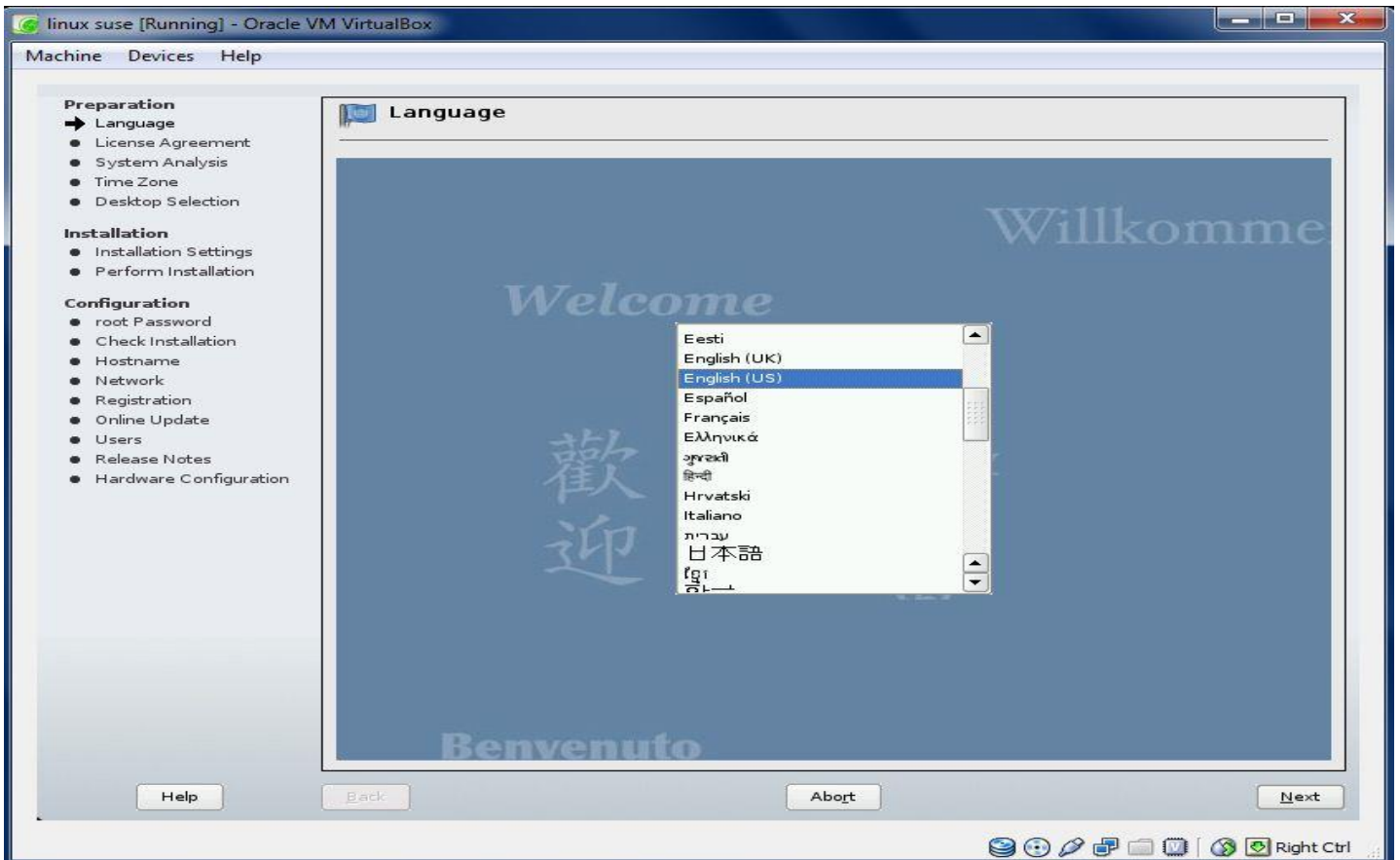
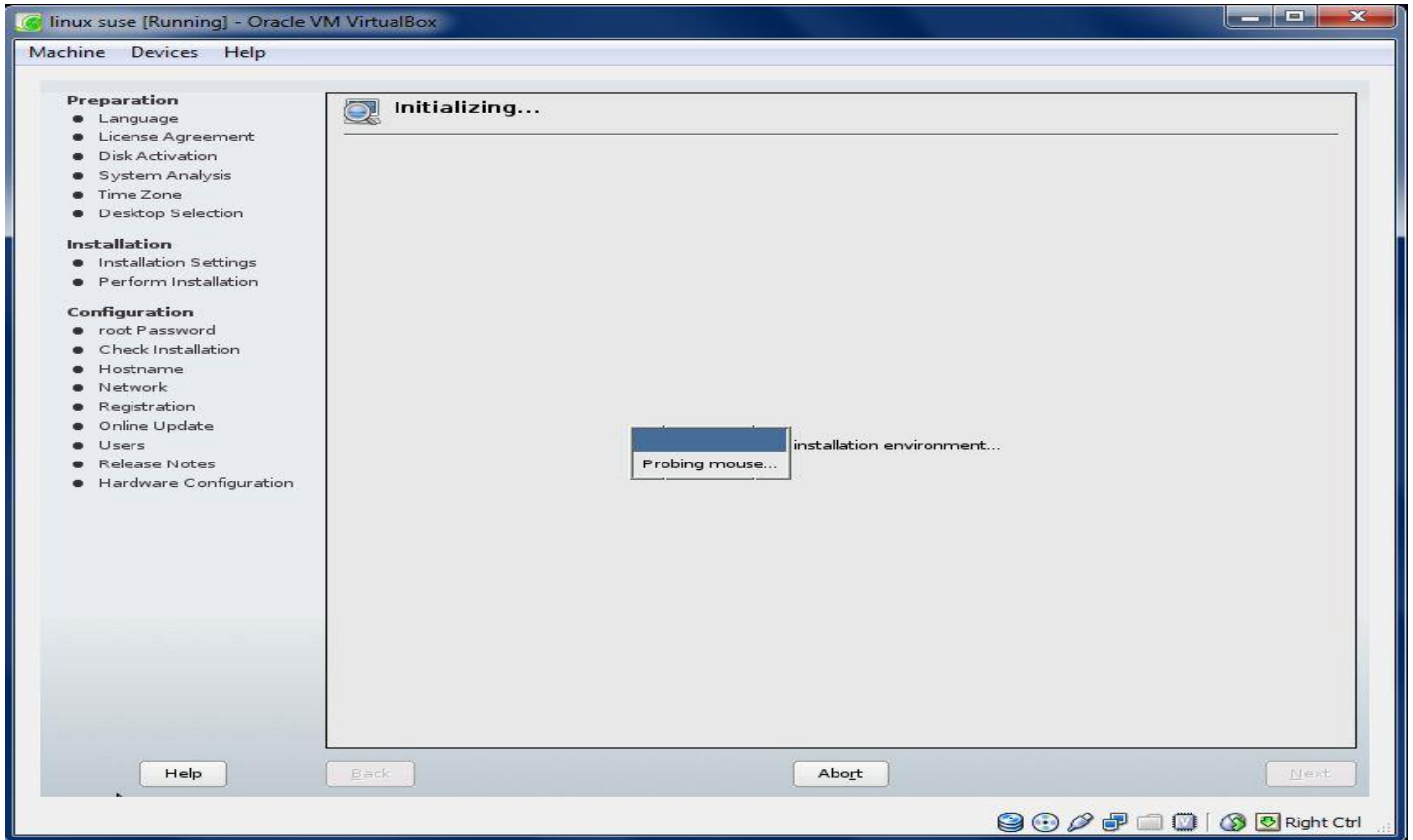


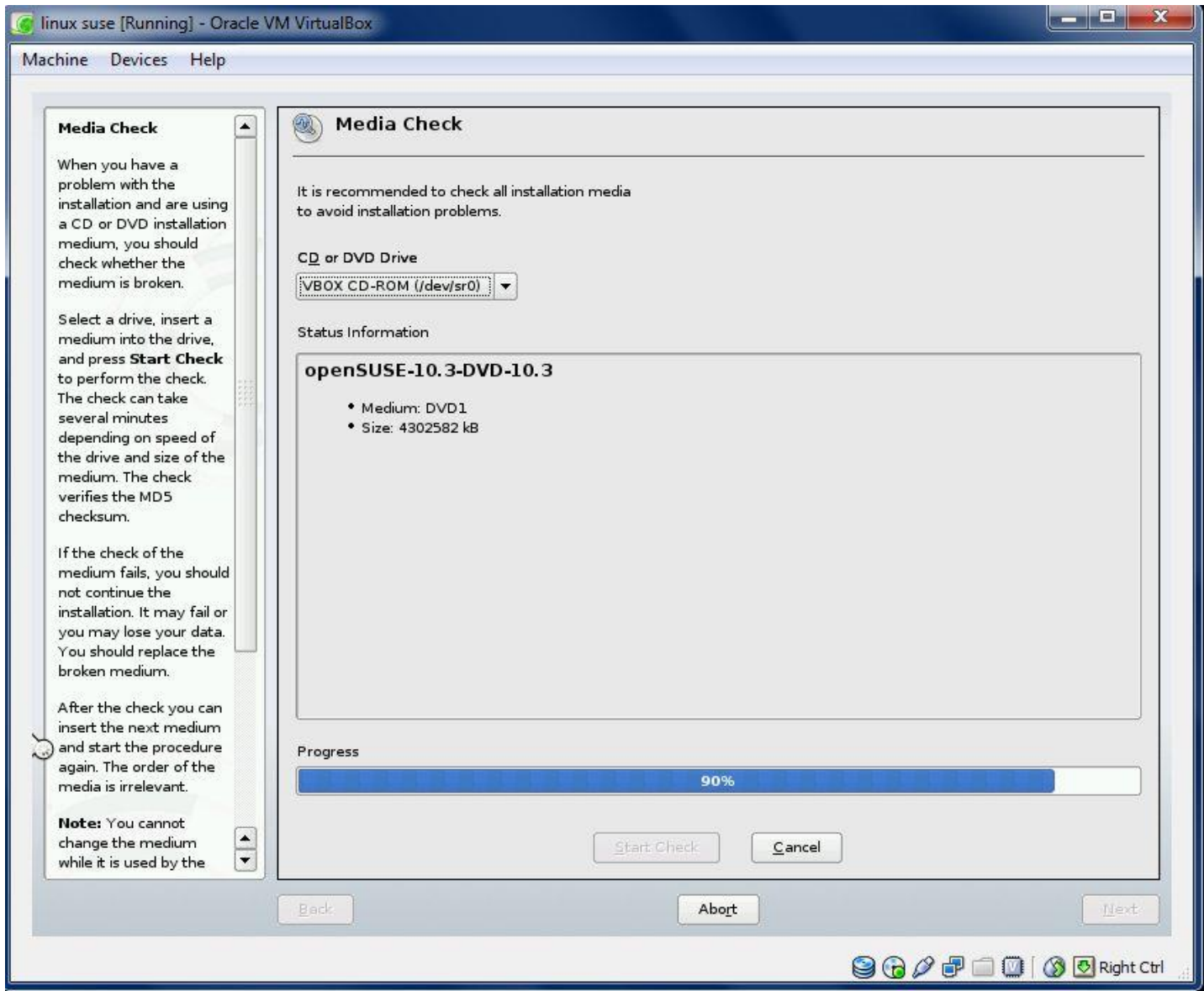
در پنجره بالا dvd linux suse و یا تصویر iso از لینوکس سوزی آن را که قبلاً با نرم افزار ultra iso و یا power iso گرفته اید ، را در درایو مجازی سیستم وارد کنید دیسک مورد نظر یا تصویر آن را به برنامه معرفی کنید و بر روی دکمه next کلیک کنید .

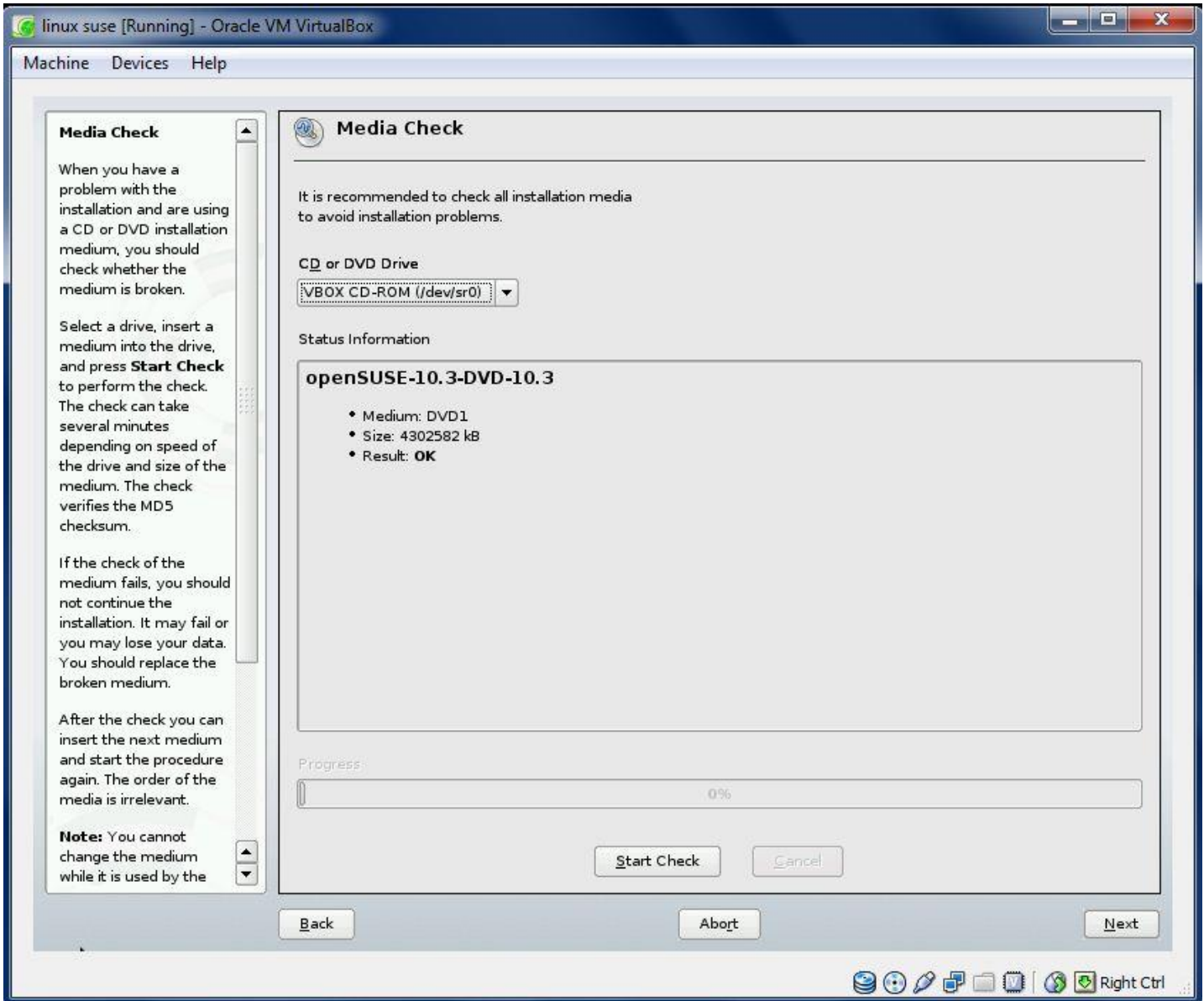
مشاهده خواهید کرد که همانند سیستم واقعی نصب سیستم عامل (لینوکس سوزی) آغاز می شود. نصب سوزی بسیار ساده است من از تمامی مراحل آن عکس گرفته و با دنبال کردن عکس ها می توانید شما هم این کار را انجام دهید :

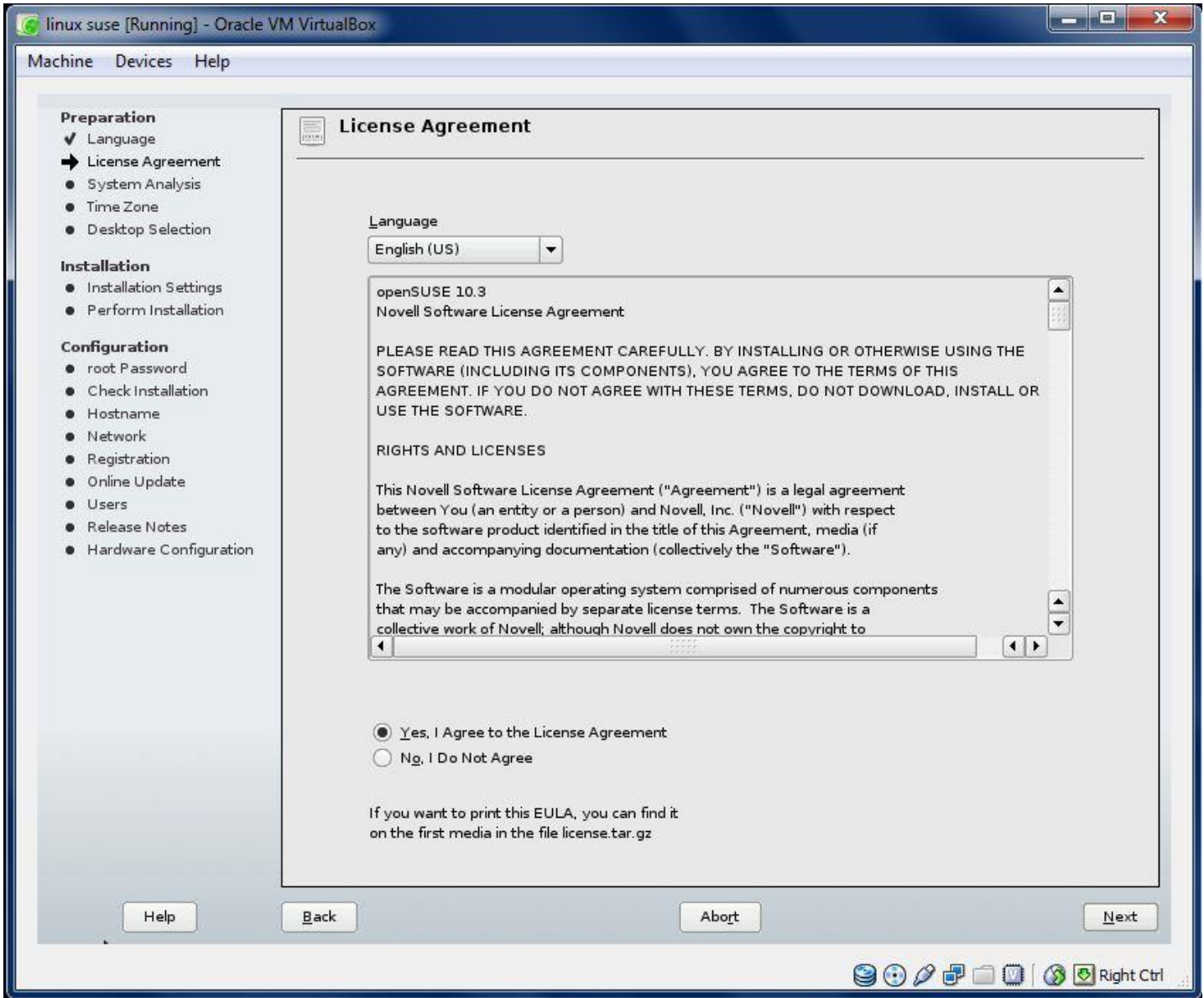


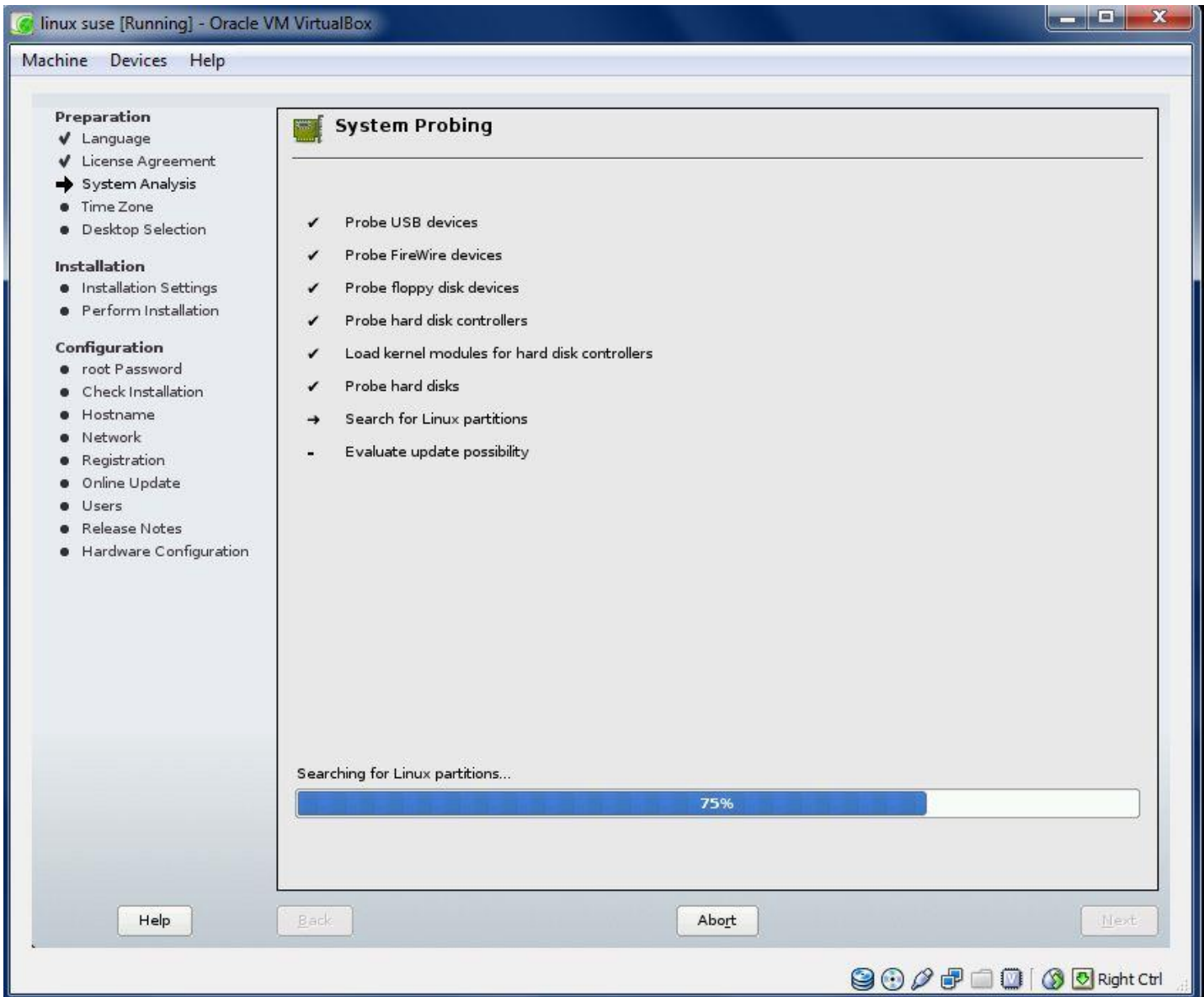




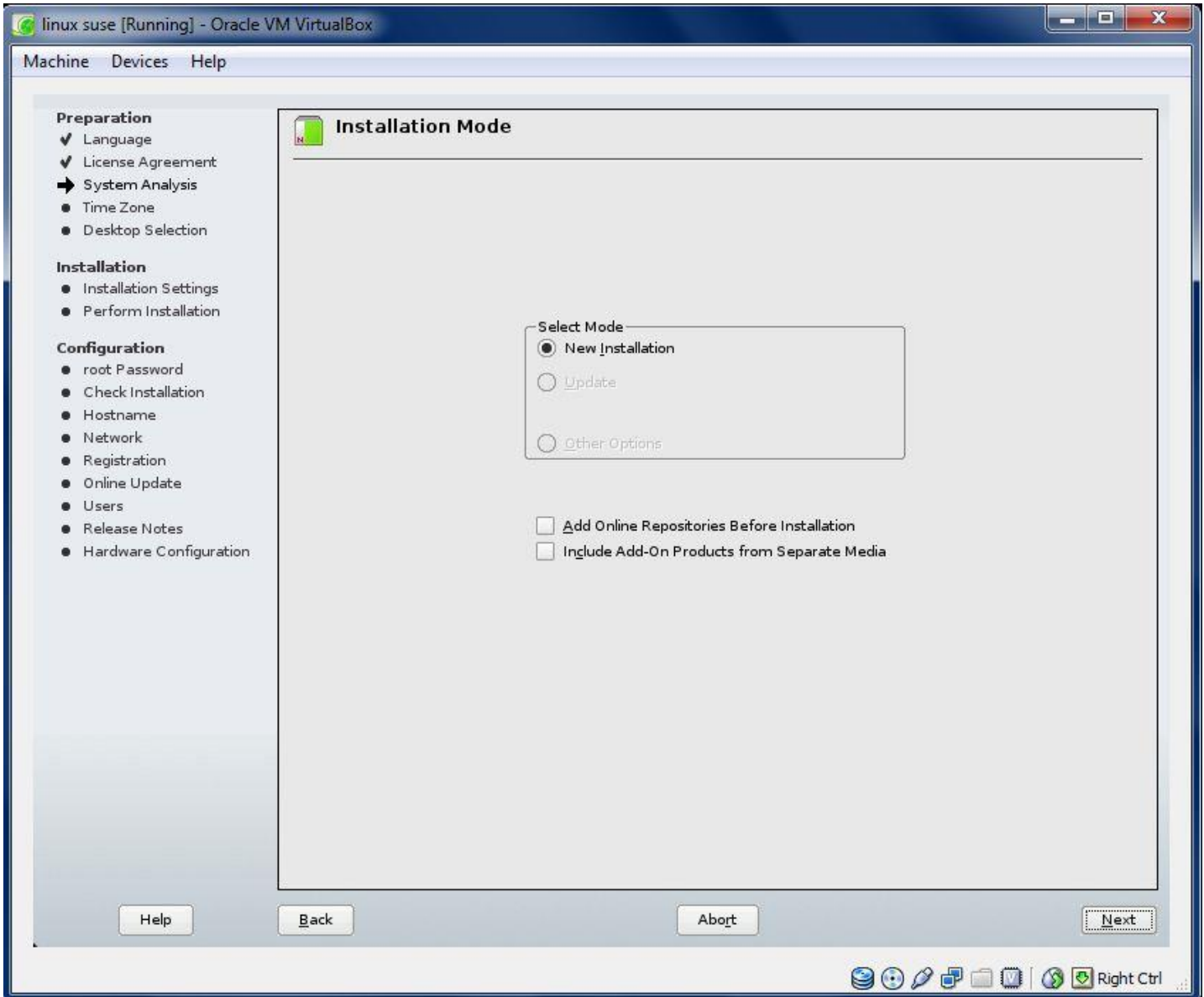


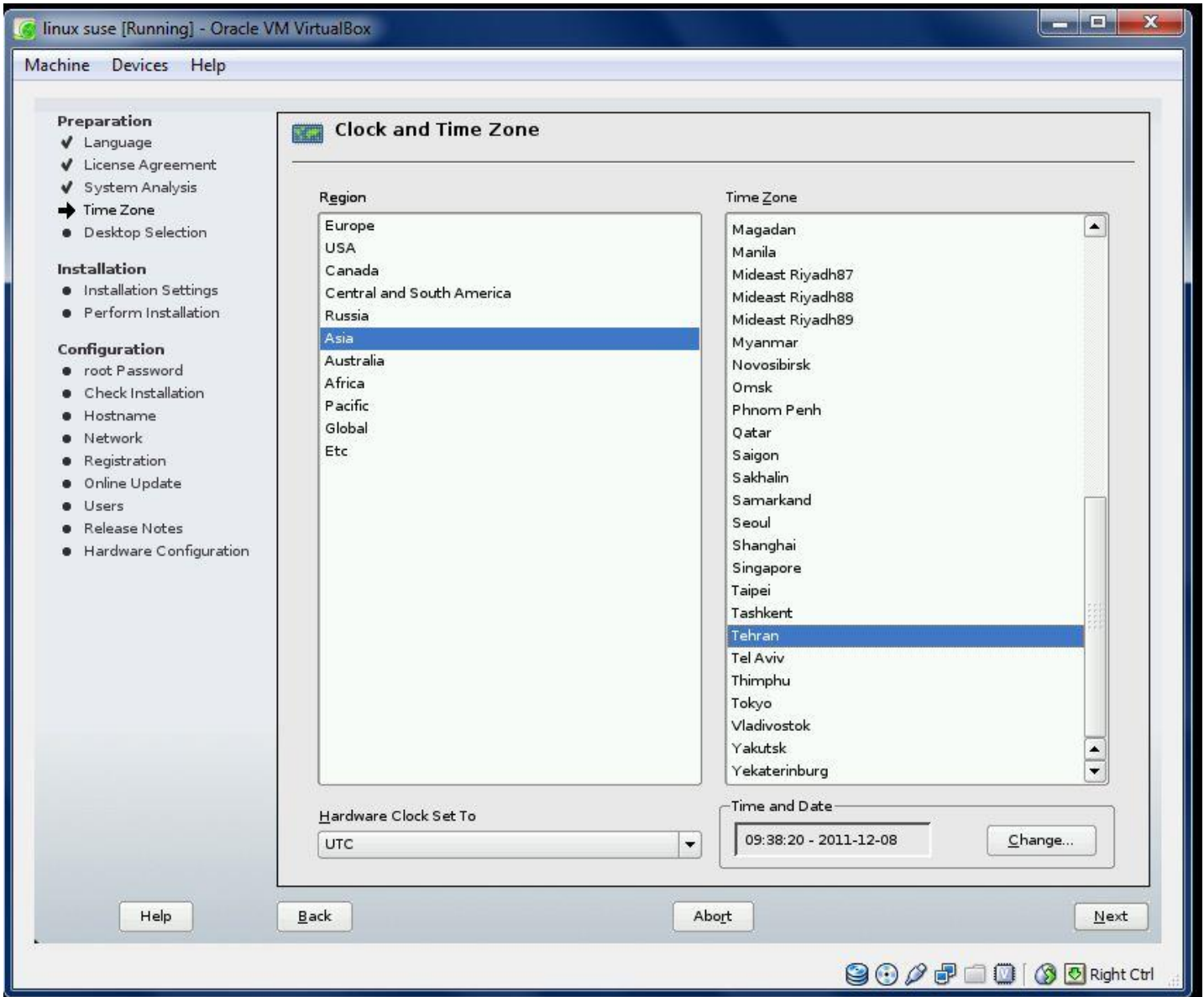




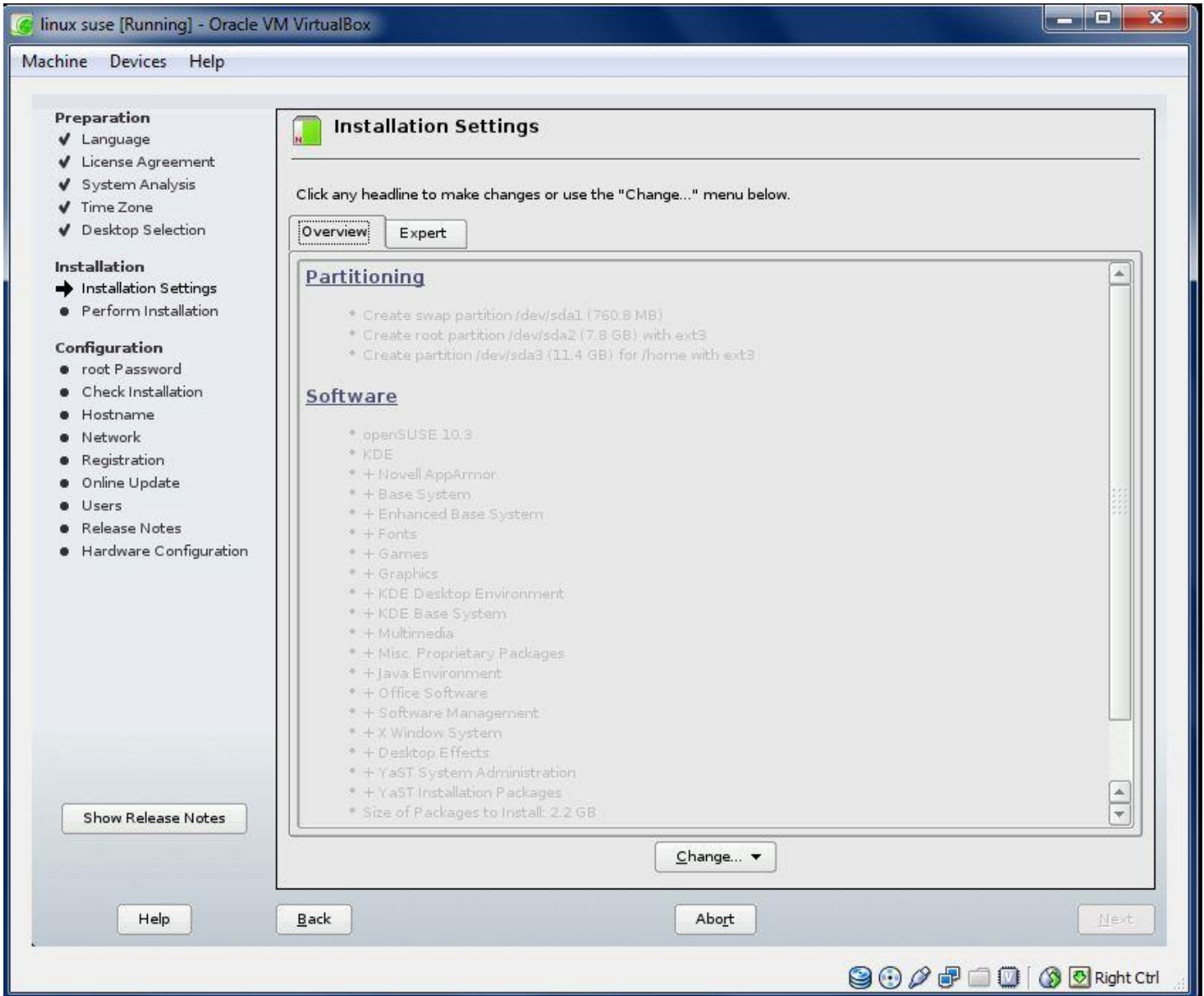


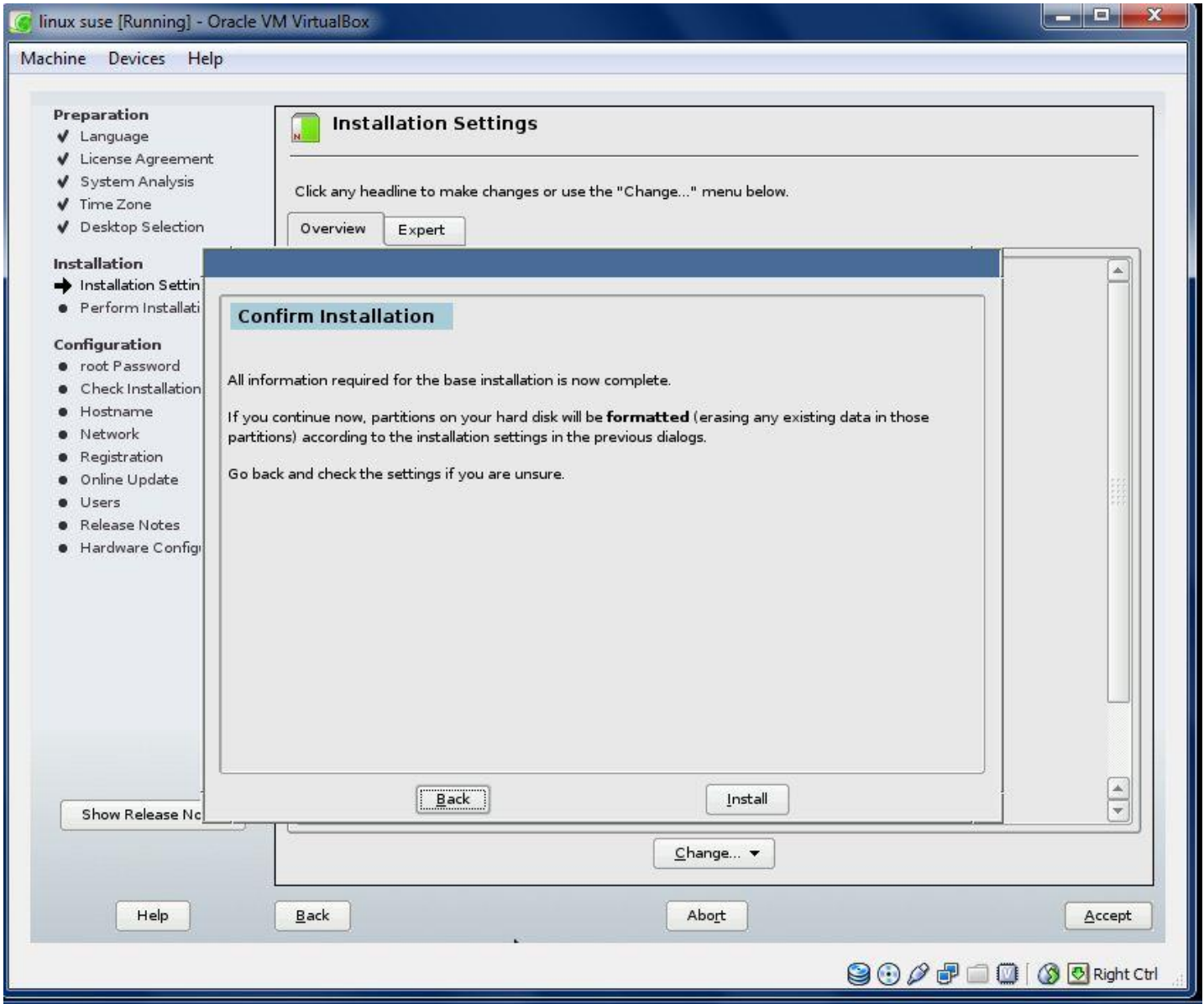


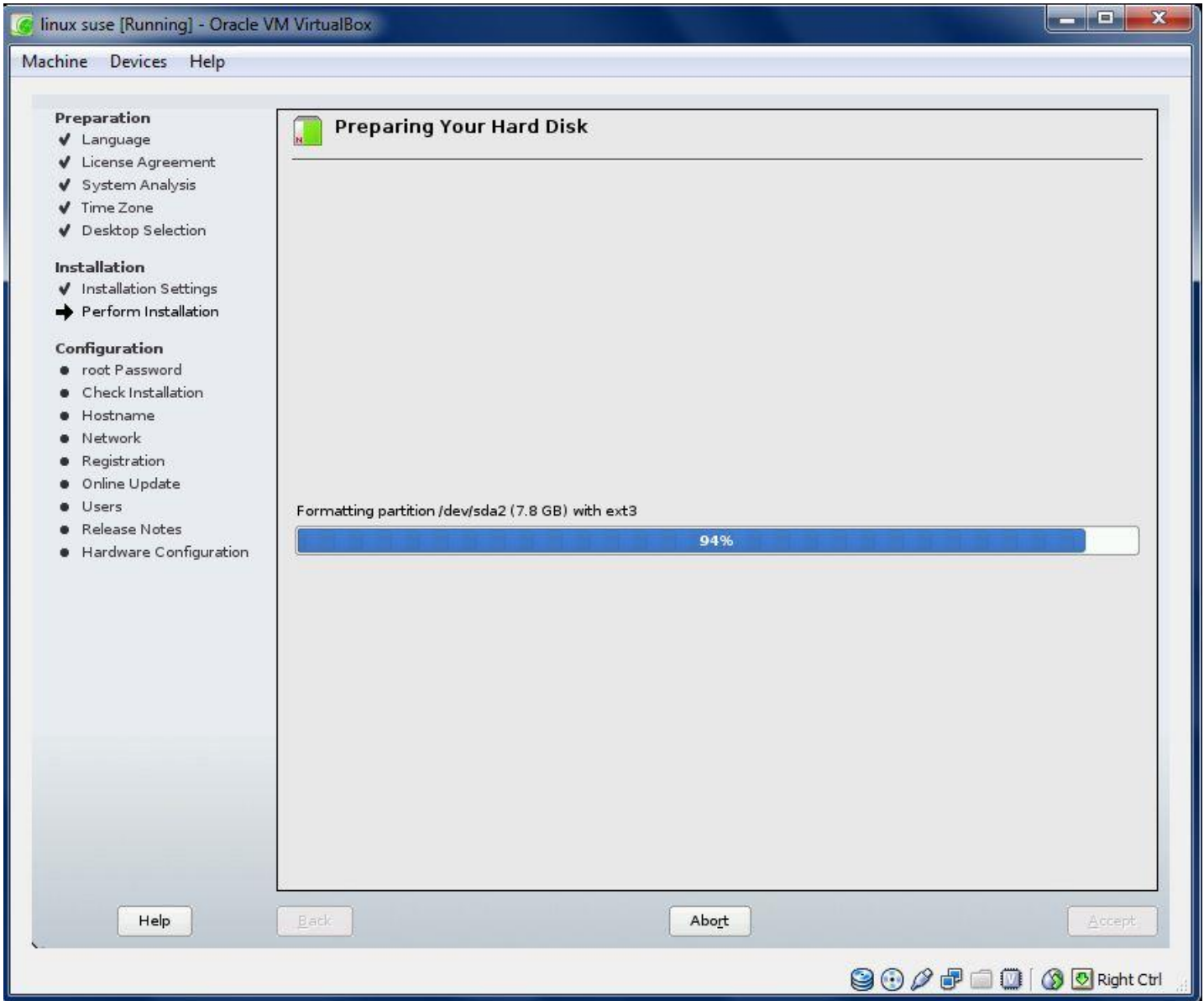


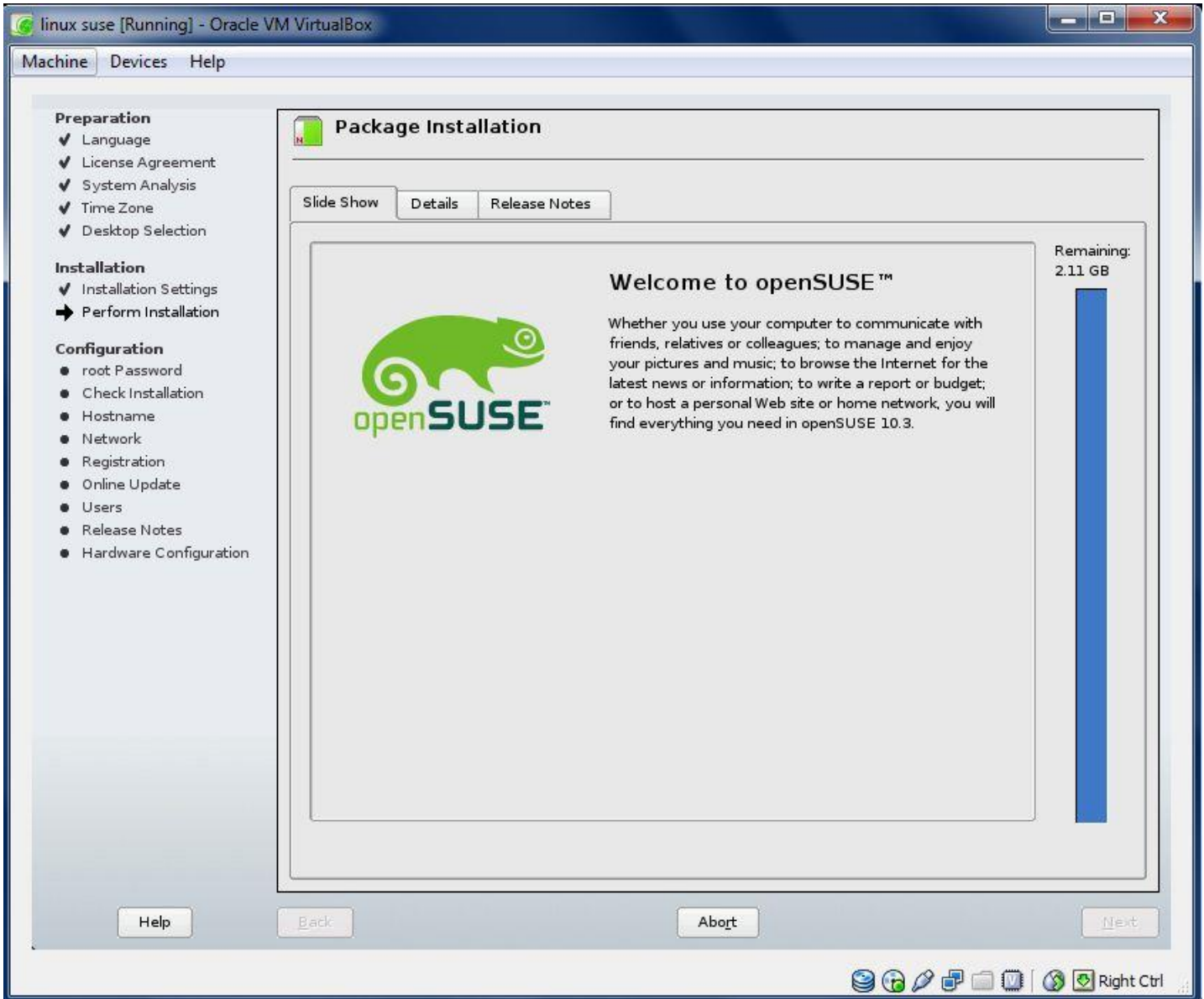


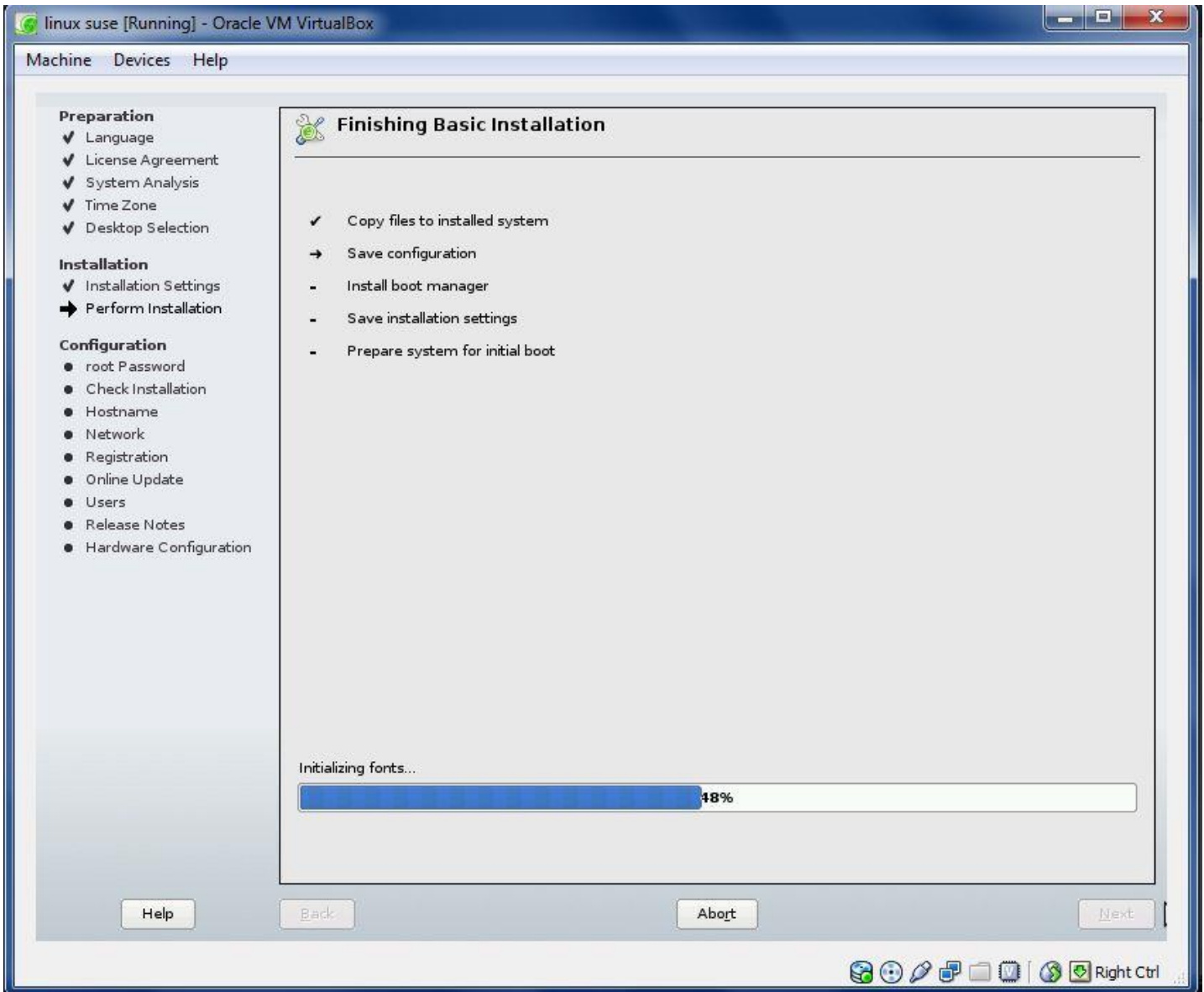




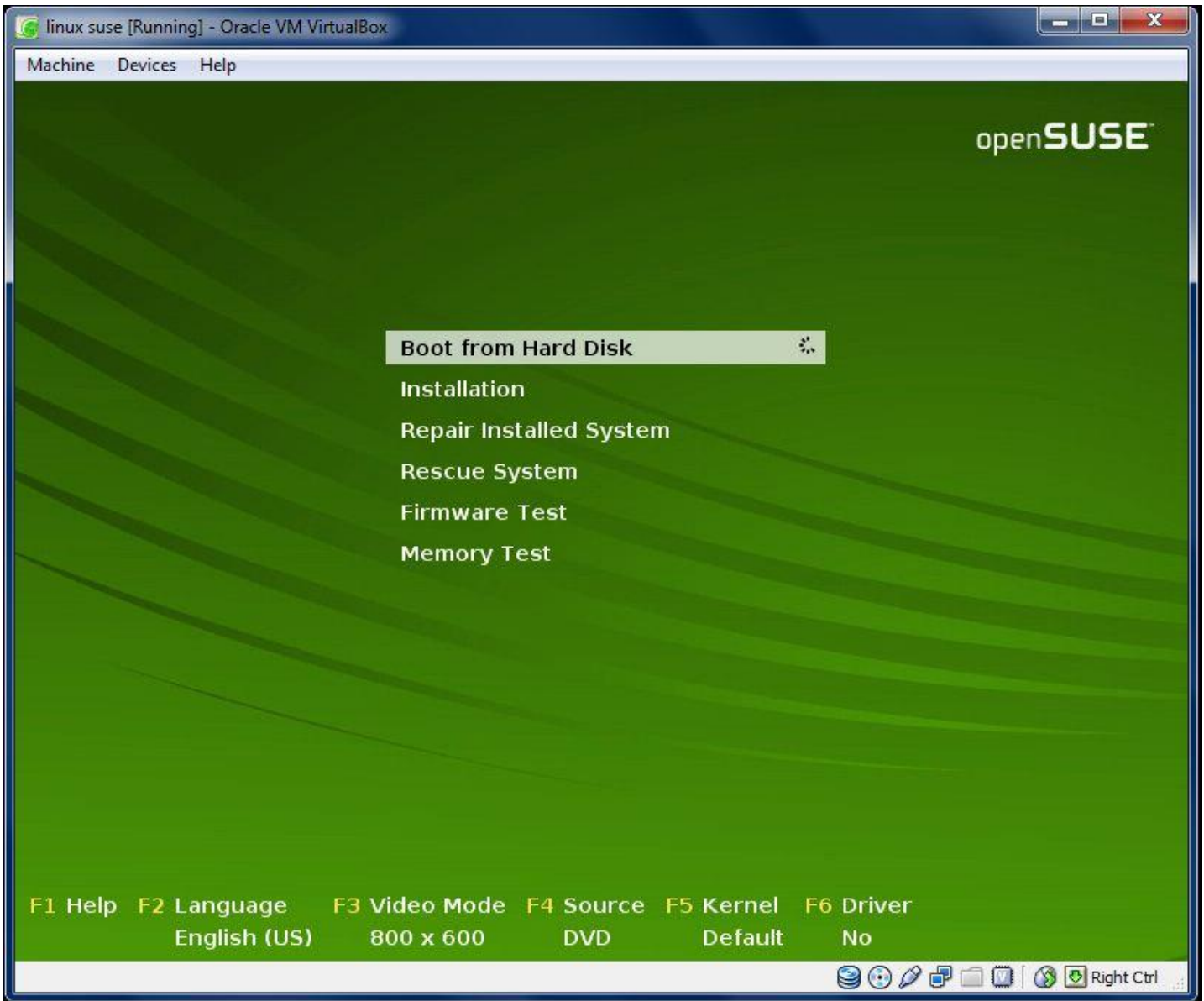












```
linux suse [Running] - Oracle VM VirtualBox
Machine  Devices  Help

openSUSE™

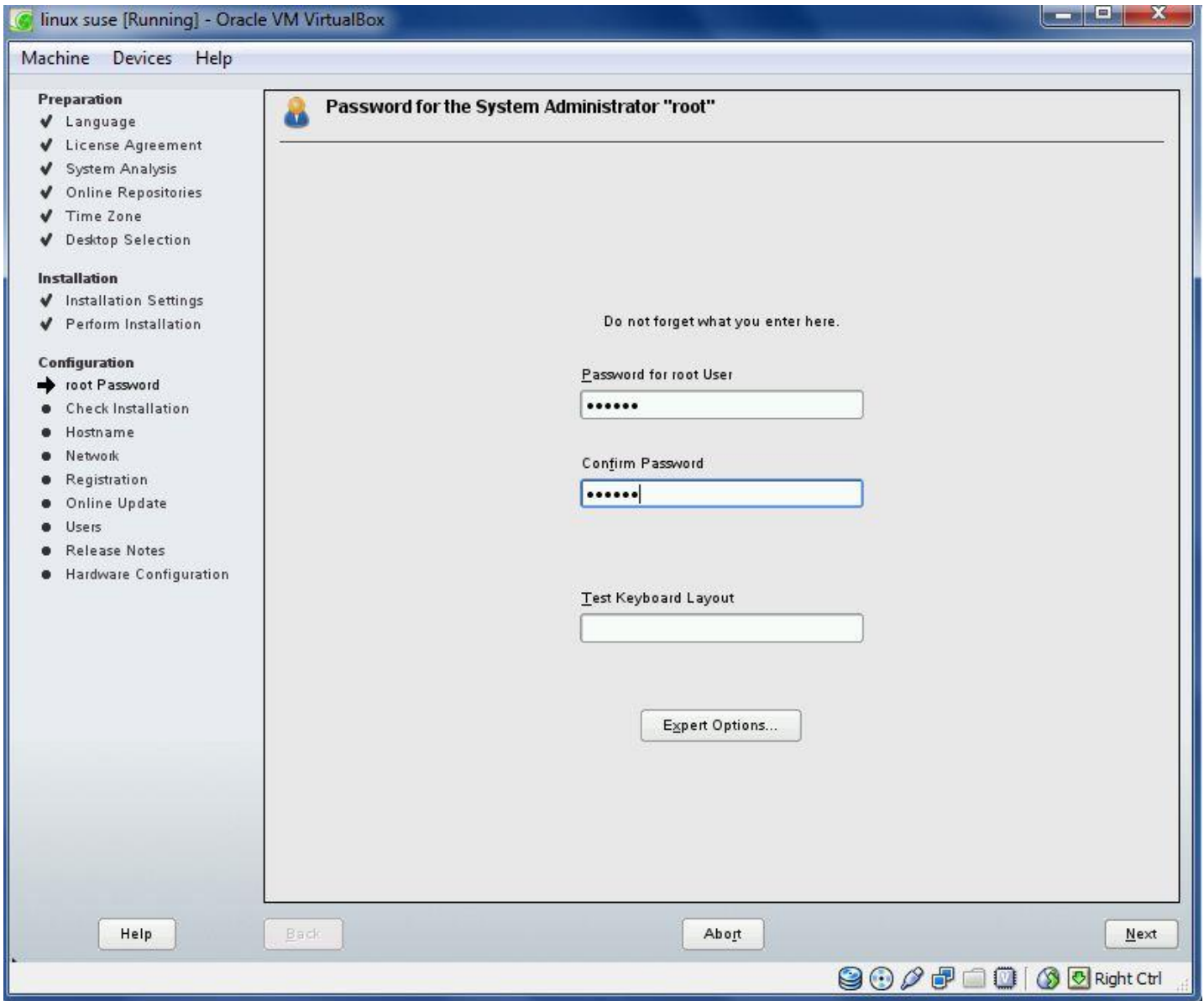
ata1.00: configured for UDMA/133
ata1: EH complete
sd 0:0:0:0: [sdal 41943040 512-byte hardware sectors (21475 MB)
sd 0:0:0:0: [sdal Write Protect is off
sd 0:0:0:0: [sdal Write cache: enabled, read cache: enabled, doesn't support DPO or FUA
sd 0:0:0:0: [sdal 41943040 512-byte hardware sectors (21475 MB)
sd 0:0:0:0: [sdal Write Protect is off
sd 0:0:0:0: [sdal Write cache: enabled, read cache: enabled, doesn't support DPO or FUA
Creating /var/log/boot.msg done
Mounting securityfs on /sys/kernel/security done
Loading AppArmor module done
Loading AppArmor profiles done
Activating remaining swap-devices in /etc/fstab... done
Configuring serial ports...
Configured serial ports done
Setting current sysctl status from /etc/sysctl.conf
net.ipv4.icmp_echo_ignore_broadcasts = 1
net.ipv4.conf.all.rp_filter = 1
fs.inotify.max_user_watches = 65536 done

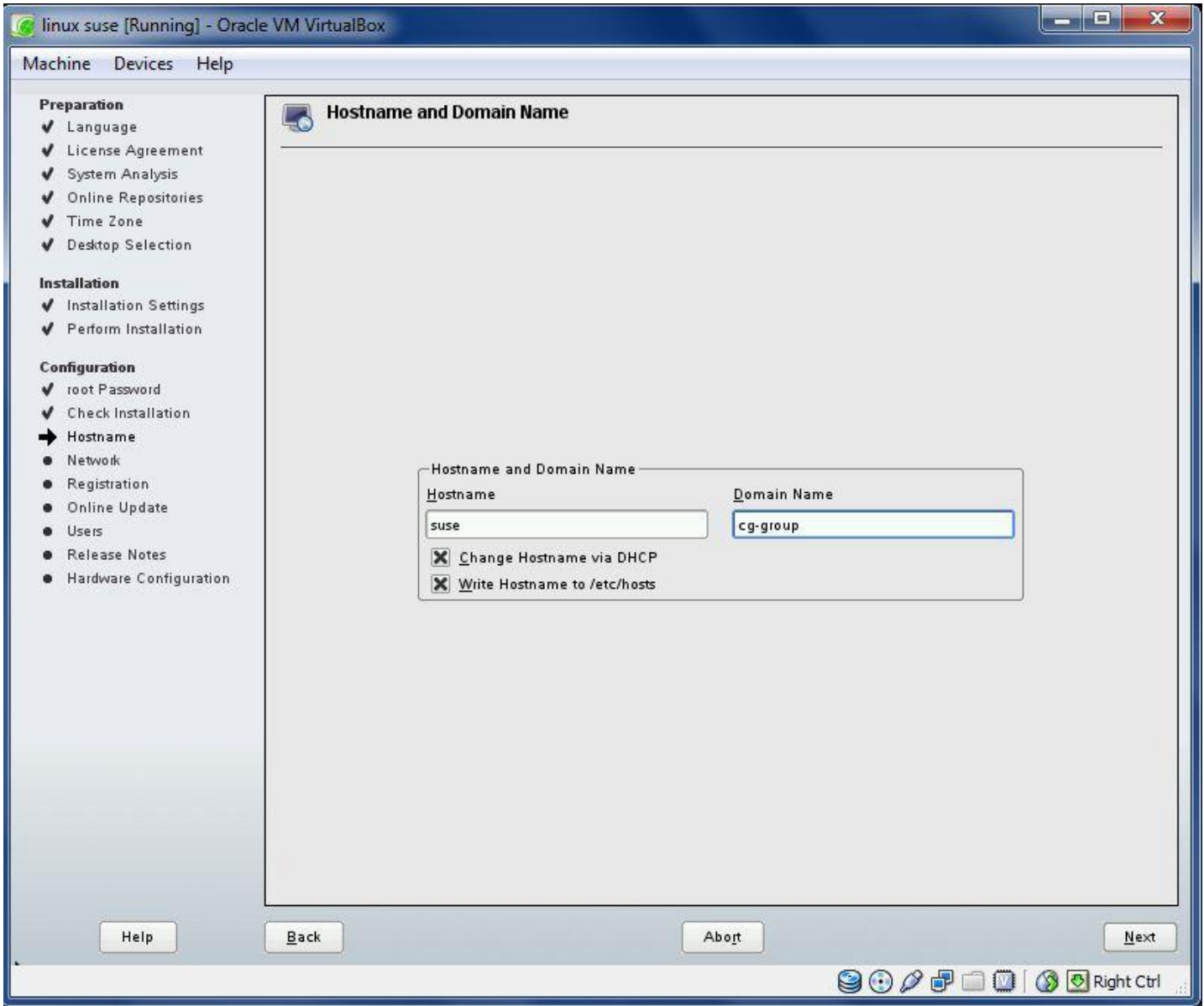
Enabling syn flood protection done
Disabling IP forwarding done

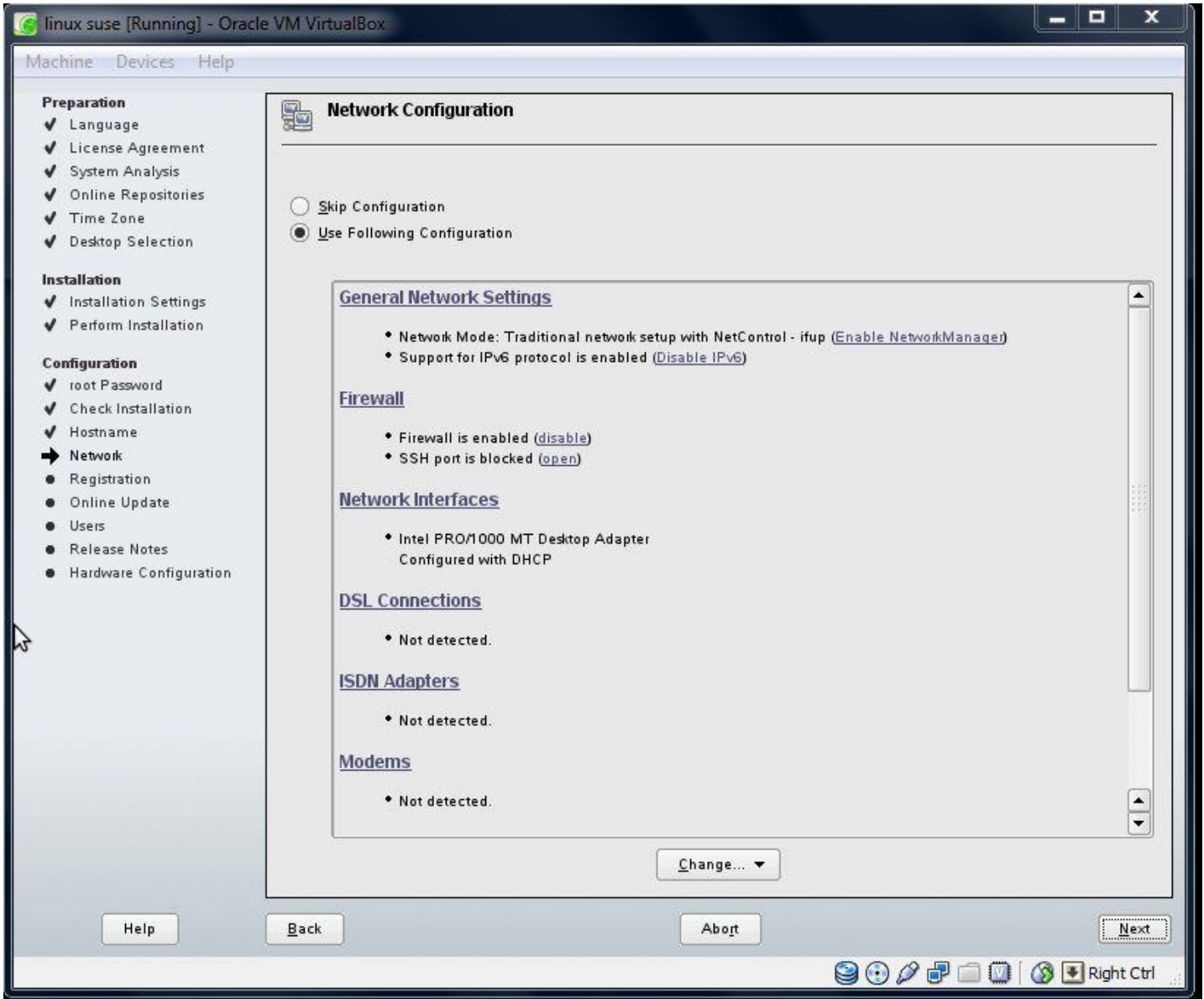
Setting up hostname 'linux' done
Setting up loopback interface lo done
lo IP address: 127.0.0.1/8
Checking for network time protocol daemon (NTPD): unused
Can't determine current runlevel done

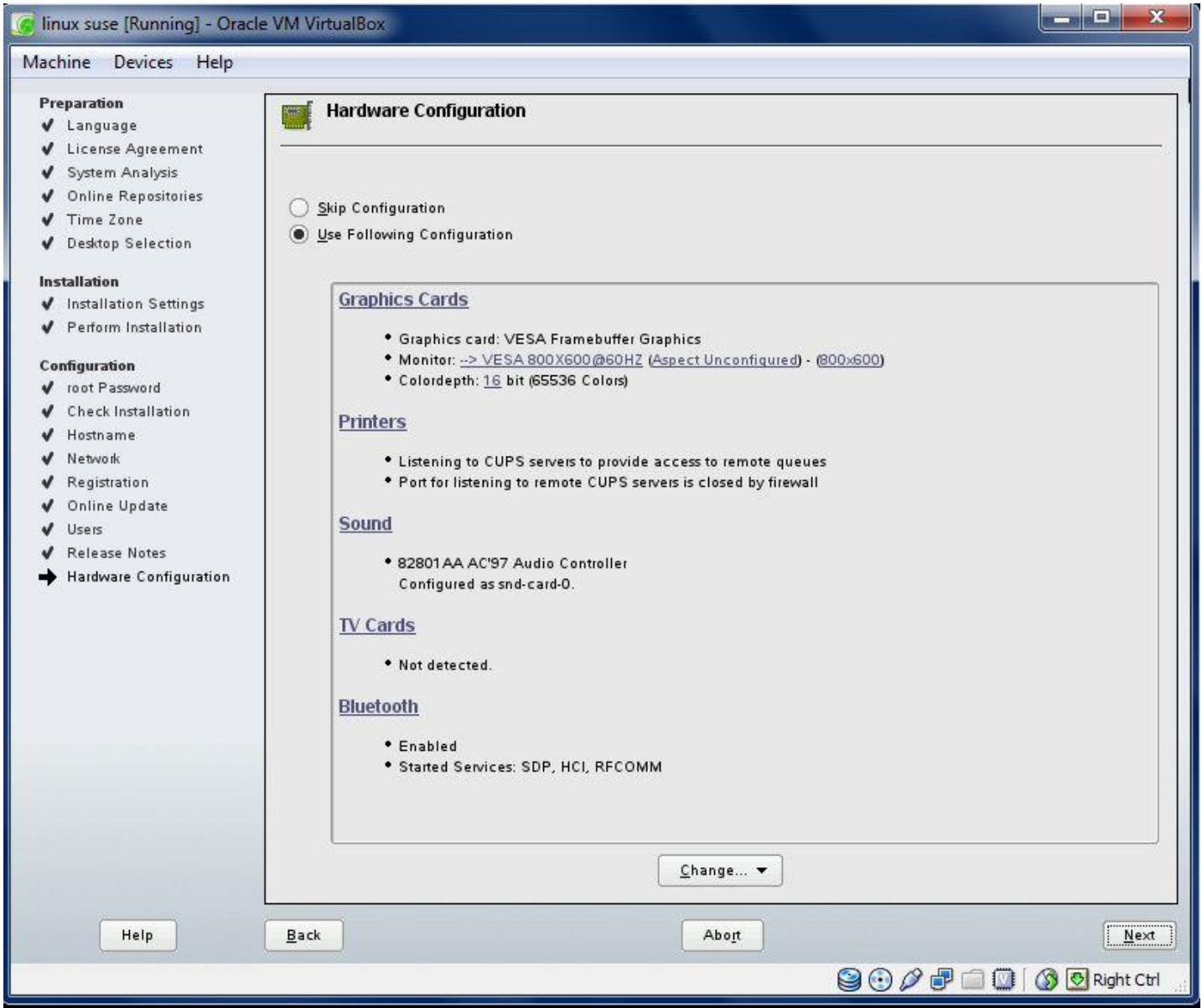
System Boot Control: The system has been set up
Skipped features: boot.cycle
System Boot Control: Running /etc/init.d/boot.local done

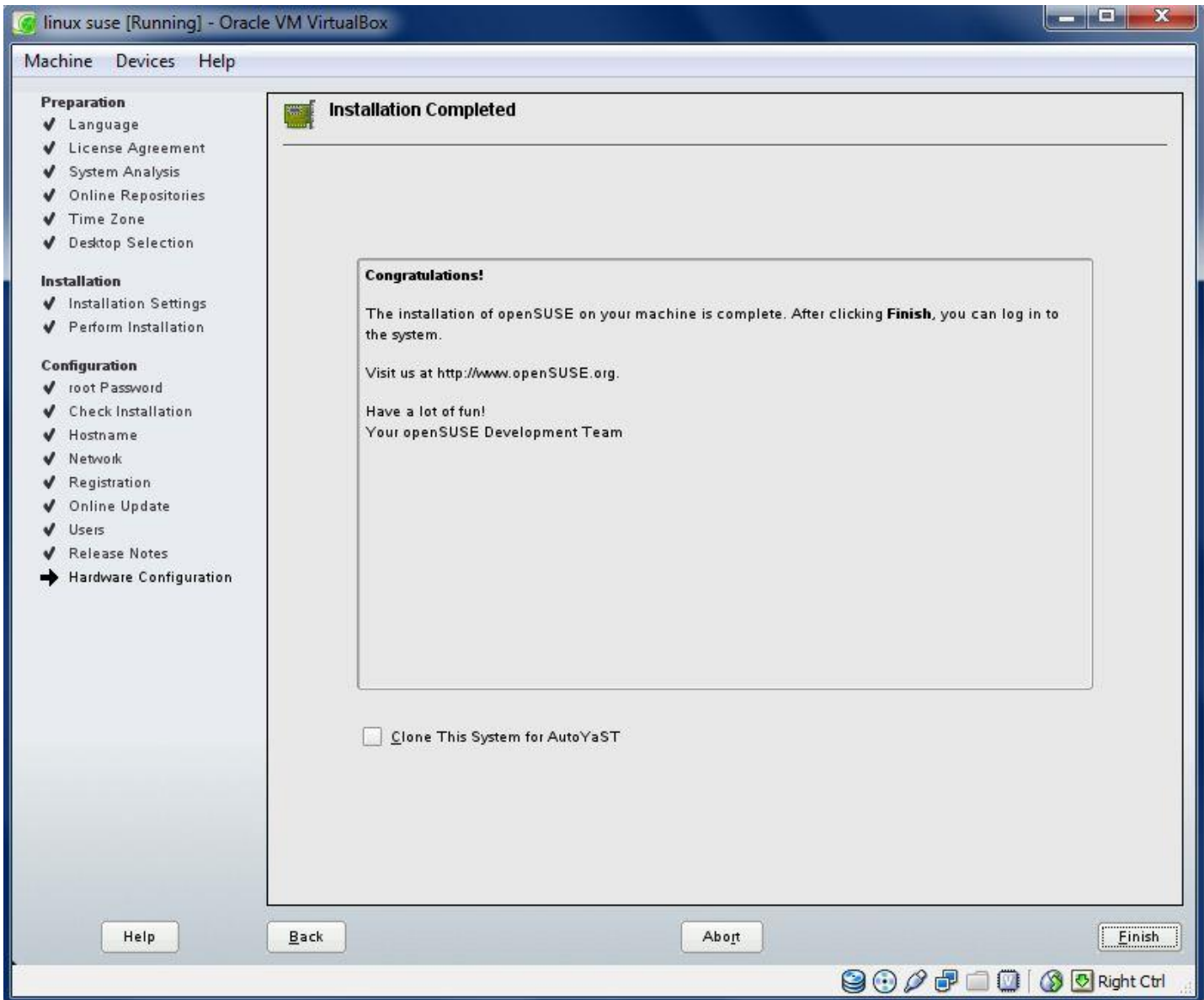
Right Ctrl
```





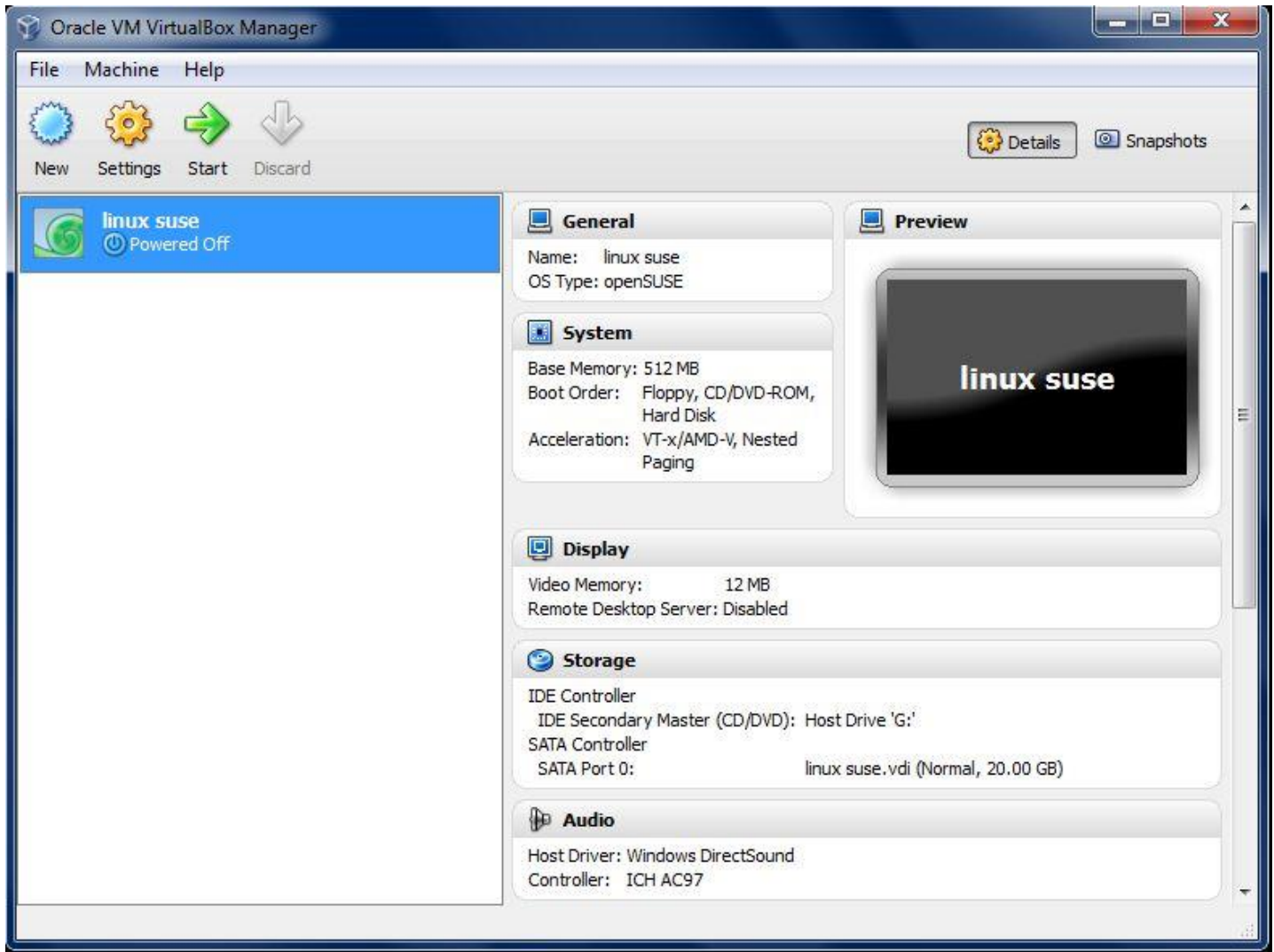












۱. چندین سیستم‌عامل می‌توانند در کنار هم و به‌صورت کاملاً ایزوله روی یک سرور (فیزیکی) نصب شوند.
۲. نگهداری راحت‌تر است.
۳. دسترسی بالا (High Availability)
۴. بازیابی راحت‌تر است.

## ۱- وقتی یک ماشین مجازی نصب می‌کنیم، و داخل آن لینوکس نصب می‌کنیم چه قابلیت‌هایی از آن گرفته می‌شود؟

در این حالت ماشین مجازی که ما نصب کرده‌ایم نمی‌تواند سی پی یو را به‌صورت کامل مدیریت کند، یعنی تمامی منابع سی پی یو را در اختیار ندارد، البته در بعضی از مجازی‌سازها این مشکل برطرف شده است، به‌عنوان‌مثال در مجازی‌ساز parallel Desktop در سیستم‌عامل مکیتاش این نرم‌افزار به این‌گونه سخت‌افزار را مدیریت می‌کند که به‌عنوان‌مثال اگر سیستم مجهز به یک سی پی یو چهار هسته‌ای باشد، یک هسته را کاملاً به سیستم‌عامل مجازی در حال اجرا اختصاص می‌دهد که این خصوصیت در کارایی خیلی تأثیر می‌گذارد. در vmware چنین خصوصیتی وجود ندارد که سی پی یو به‌صورت اختصاصی به سیستم‌عامل مجازی اختصاص داشته باشد، از تفاوت‌هایی که می‌توان ذکر کرد در خصوص قابلیت‌های سیستم‌عامل نصب‌شده به‌صورت بوت در قیاس با نصب به‌صورت مجازی می‌توان گفت: از نظر شبکه سیستم‌عاملی که به‌صورت بوت نصب‌شده می‌تواند مدیریت بهتری روی شبکه داشته باشد مثل سرورهای Delicate در قیاس با vps ها ...

سیستم‌عامل‌هایی که به‌صورت مجازی نصب می‌شوند، حافظه به‌صورت اختصاصی دریافت می‌کنند، ولی در هنگامی که بار زیادی بر روی سیستم اصلی باشد این حافظه‌ی اختصاصی نیز تحت تأثیر قرار گرفته و کارایی مورد انتظار را نخواهد داشت. پورت‌های مانند port 1394 که برای کارهای حرفه‌ای مثل میکس می‌باشد را ماشین مجازی پشتیبانی نمی‌کند و فقط پورت‌های usb و شبکه را پشتیبانی می‌کند.

و کارت گرافیک ماشین مجازی بستگی دارد که آن ورژن از ماشین مجازی تا چه قدر را پشتیبانی می‌کند و به کارت گرافیک کامپیوتر ما هر چه قدر هم بالا باشد آخرین ورژن ماشین مجازی فعلاً تا 128mb از ram برمی‌دارد در صورتی که ممکن است حافظه کارت گرافیک ما 2GB باشد.

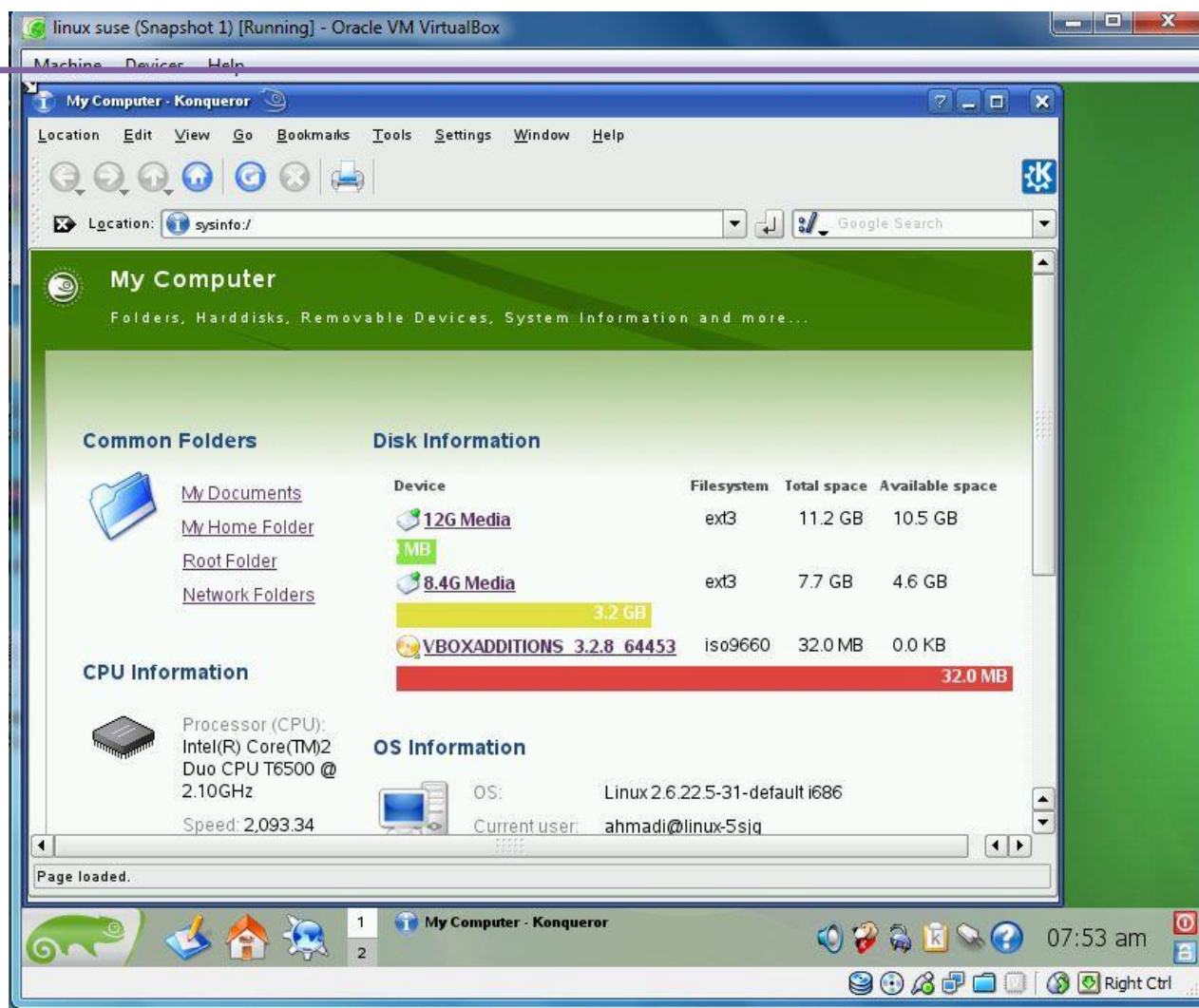
## ۲- نحوه‌ی انتقال فایل از سیستم‌عامل نصب‌شده‌ی مجازی به سیستم‌عامل میزبان؟

نحوه‌ی انتقال فایل‌ها در مجازی‌سازهای مختلف کمی متفاوت است، ولی روش معمول انتقال فایل از طریق شبکه به این صورت است که شبکه‌ی سیستم‌عامل میزبان به اشتراک سیستم‌عامل مجازی درمی‌آیند در این صورت سیستم‌عامل مجازی یک آدرس شبکه خواهد داشت که در این صورت می‌تواند با میزبان تبادل اطلاعات داشته باشد، این یکی از روش‌های معمول و امکان‌پذیر در تمامی مجازی‌سازها است، روش‌های ابتدایی هست، مثل انتقال فایل با استفاده از یک فلش مموری که خوب این روش جالبی نیست، چون باید اطلاعات بر روی فلش ریخته شود و بعد دوباره به سیستم‌عامل

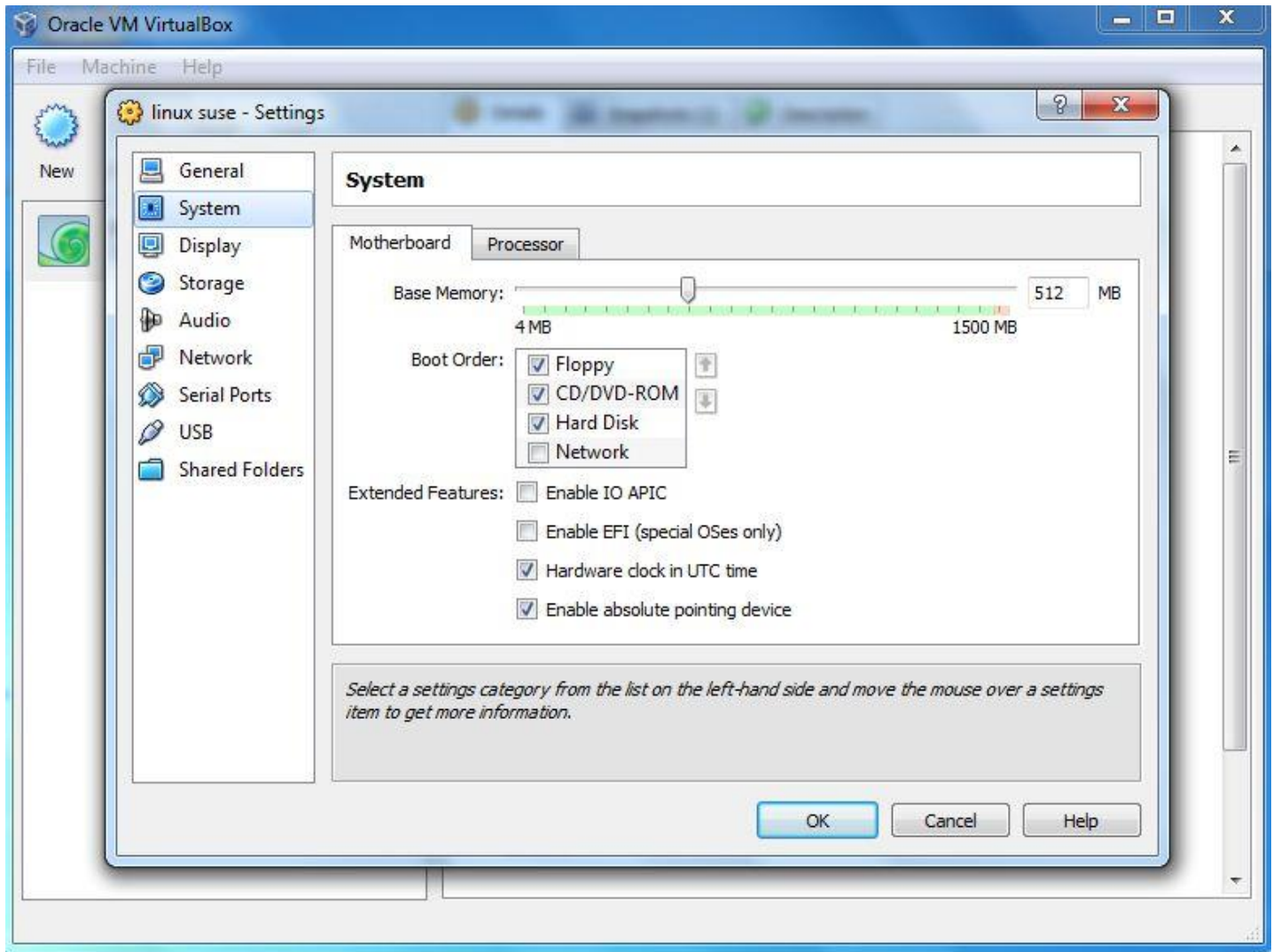
مجازی انتقال یابد .... بیشتر گفتیم روش انتقال فایل بسته به نوع مجازی ساز می تواند متفاوت باشد ، به عنوان مثال در مجازی ساز VirtualBox می توانیم فولدری را در سیستم عامل میزبان به اشتراک بگذاریم که در این صورت می توان فایل های مورد نظر را در آن فولدر ریخته و انتقال دهیم.

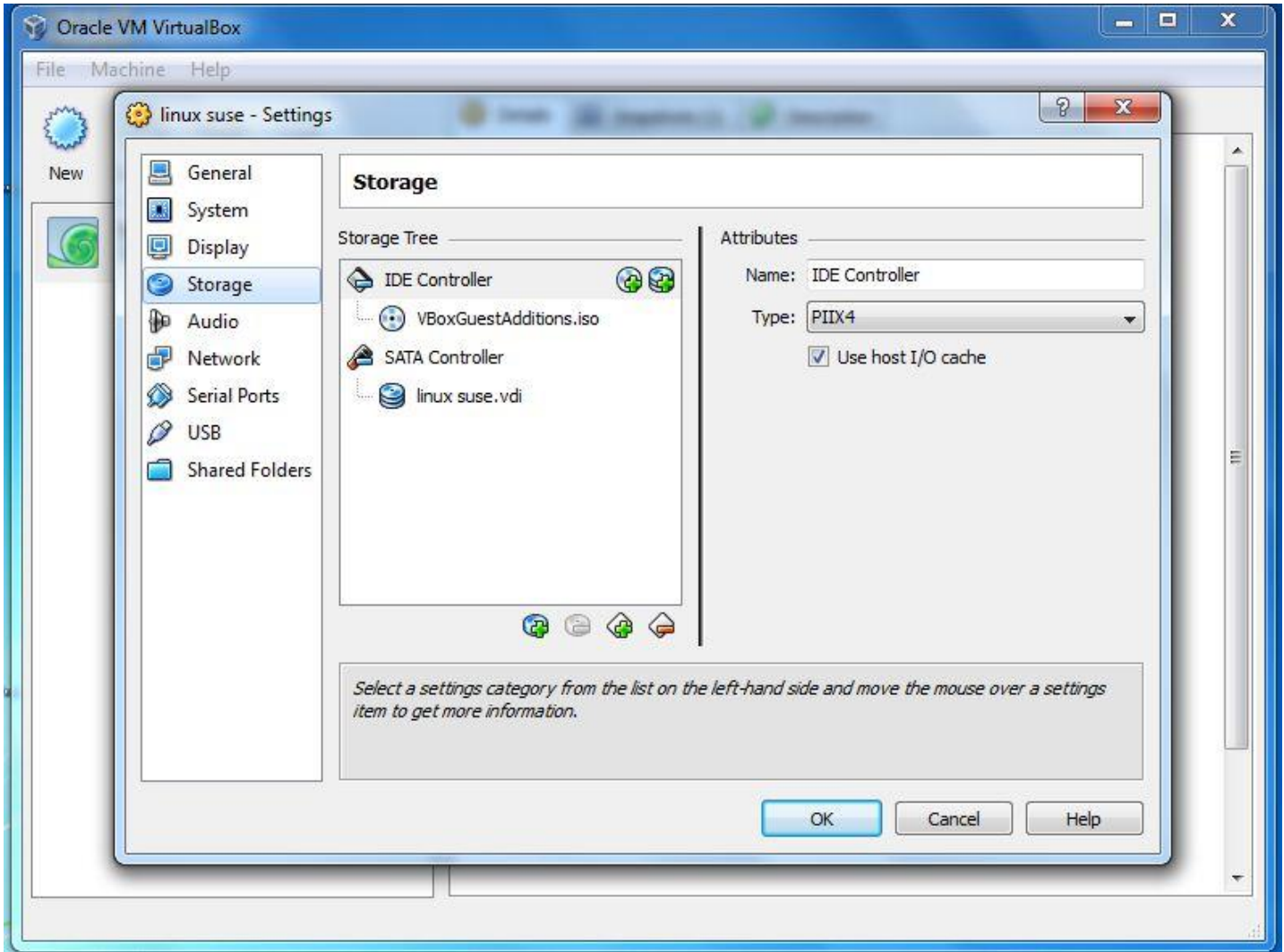
### ۳- در صورتی که یک فایل با پسوند خاص داشته باشیم که در ویندوز به درستی اجرا می شود ولی در لینوکس این فایل اجرا نمی شود! به چه نحوی می توانیم این فایل را در لینوکس اجرا کنیم؟

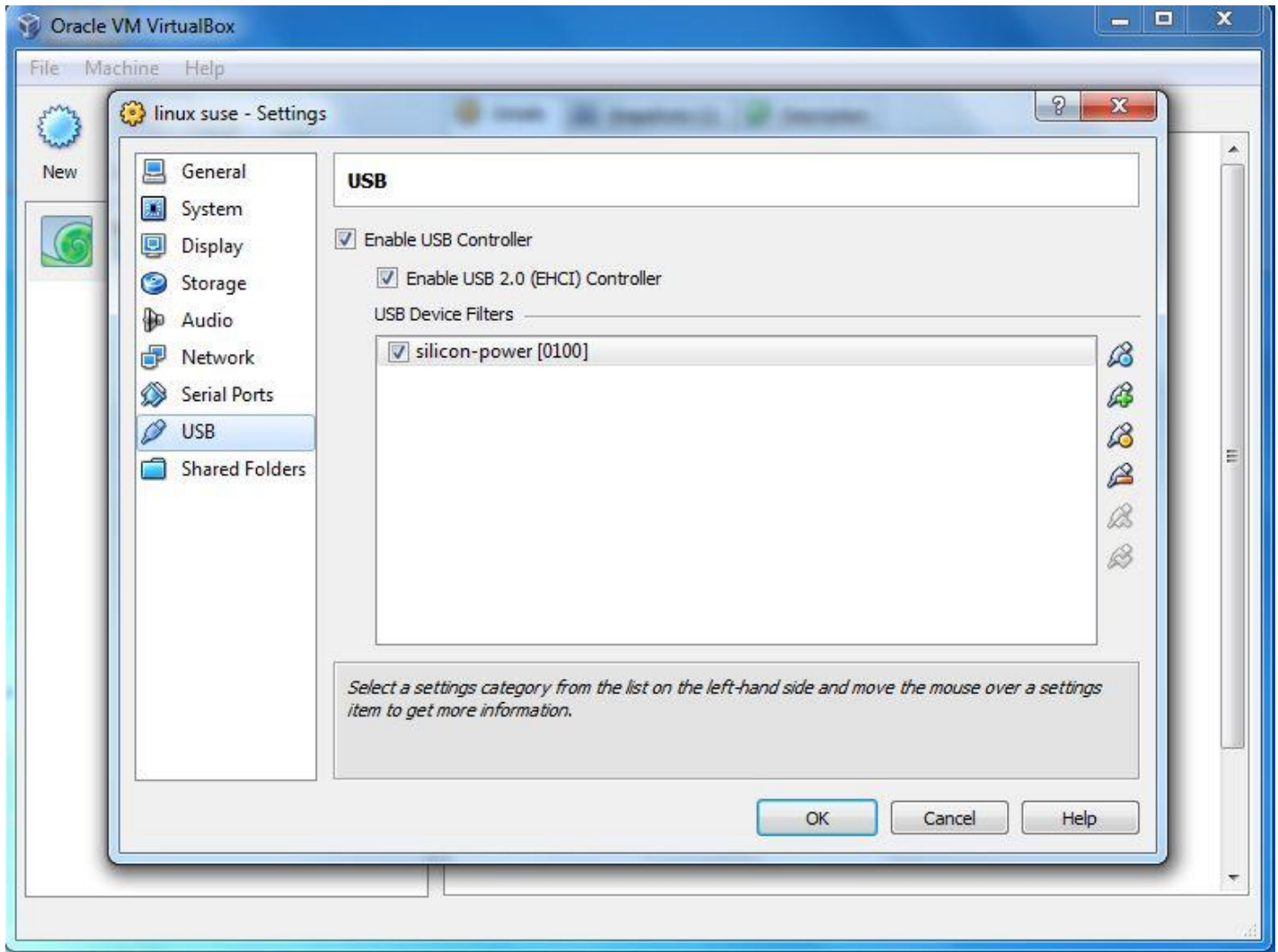
باید نرم افزار مربوطه به آن را نصب کنیم در لینوکس تا فایل مورد نظر در آن اجرا شود ، اگر برنامه ی مربوطه به آن ورژن لینوکسی نداشت ، می توانیم فایل برنامه ی ویندوزی را با استفاده از برنامه ی wine تحت ویندوز را در لینوکس اجرا کنیم ، برنامه ی و این مانند یک مجازی ساز برنامه های .exe را اجرا می کند در لینوکس سوزی هم فلش و هم dvd drive شما به راحتی شناخته می شود و با باز کردن پنجره my computer در قسمت disk information قرار دارند.

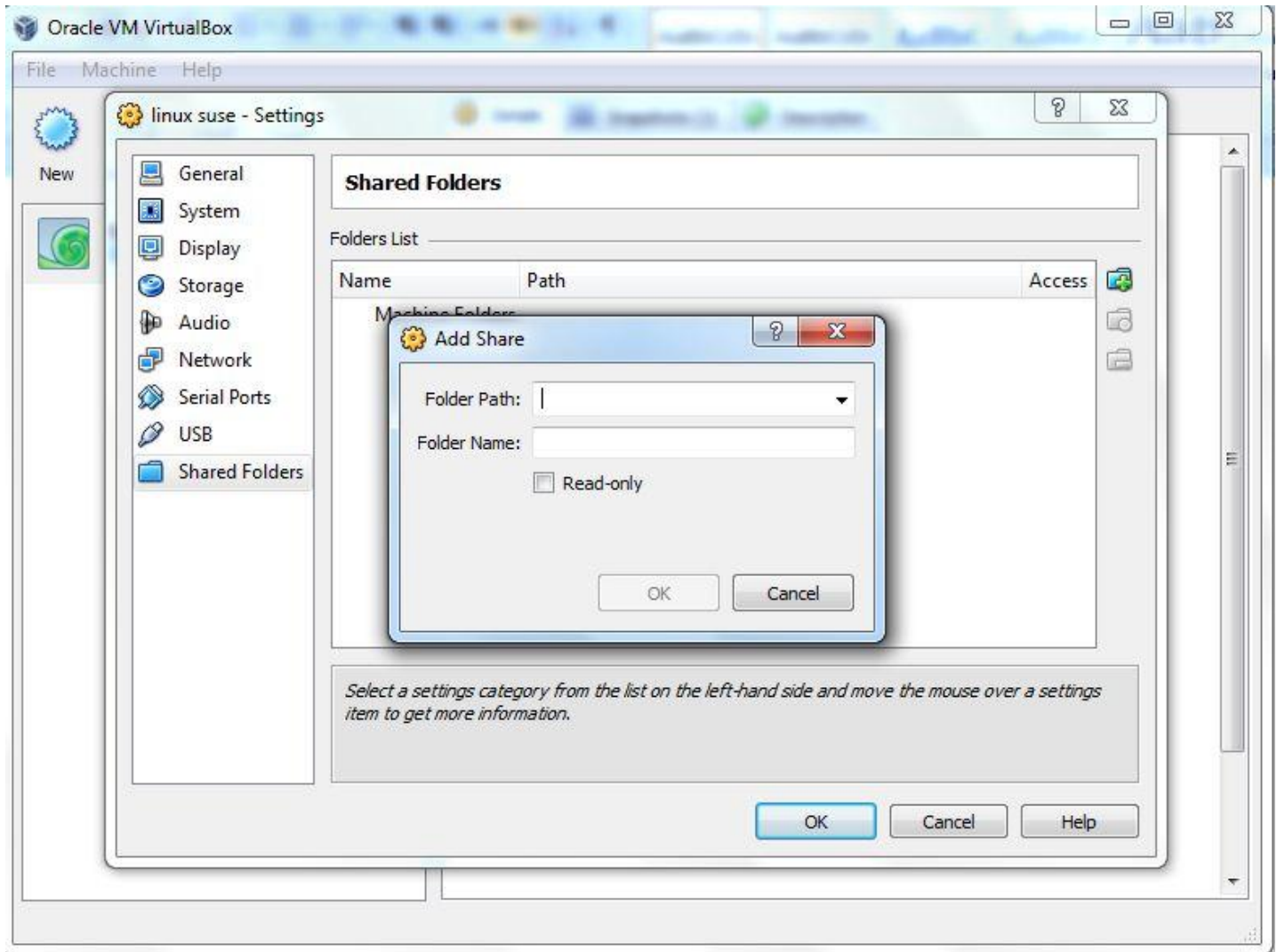


تنظیمات مجدد فضای اختصاص داده شده از حافظه اصلی و هارد دیسک و اضافه کردن **devic** جدید و **share** کردن در **virtual box** با زدن کلید **setting** در صفحه نخست نرم افزار **virtual box** امکان پذیر است :









## دستور کار

- ۱- تحقیق کنید چند نرم افزار ماشین مجازی تاکنون وجود دارد و کدام بهتر است ؟
- ۲- فولدری را با نام خود share کرده فایل های موردنیاز خود را در آن بریزید و آنها را در لینوکس انتقال داده و استفاده کنید؟
- ۳- با استفاده از نرم افزار wine یکی از نرم افزارهای ویندوزی موردنیاز خود را در لینوکس اجرا کنید؟
- ۴- تاریخچه مختصری از سیستم عامل لینوکس و مزیت های آن نسبت به ویندوز را بنویسید؟
- ۵- در مورد توزیع های مختلف لینوکس و اینکه کدام توزیع مناسب تر است تحقیق کنید؟

## طریقه ی نصب و برپا سازی Wine

Wine مخفف Wine Is Not Emulator متن باز است. wine برای اجرای نرم افزارهای ویندوزی در Linux , FreeBSD , Max os X و Solaris به کار می رود. wine برای اجرای نرم افزارهای ویندوزی احتیاجی به مایکروسافت

ویندوز ندارد. **wine** بر پایه ی **API** های ویندوز ساخته شده است. در اوبونتو شما می توانید آن را به راحتی از مخزن رسمی **wine** در **Launchpad** نصب کرد. برای نصب مرحله ی بعد را بخوانید.

نصب **wine** در اوبونتو بسیار ساده است. برای شروع کار با دستور زیر مخزن **wine** را به لیست مخازن خود اضافه کنید.

```
echo deb http://wine.budgetdedicated.com/apt jaunty main | sudo tee -a /etc/apt/sources.list
```

سپس کلید مخزن را دانلود و نصب کنید:

```
.gpg -O- | ۲۶۳EE۳۸۷wget -q http://wine.budgetdedicated.com/apt/ - sudo apt-key add
```

و سپس **wine** را نصب کنید:

```
sudo apt-get update  
sudo apt-get install wine
```

سپس با دستور زیر می توانید ورژن **wine** نصب شده را ببینید:

```
wine --version
```

### تنظیمات اولیه

قبل از استفاده از **wine** شما باید درایور غیرواقعی **C:** را بسازید. برای این کار دستور زیر را وارد کنید.

```
winecfg
```

با اجرای این دستور پوشه ی مخفی ای در پوشه ی خانگی شما درست می شود. این پوشه شامل تنظیمات **Registry** است که در ویندوز نیز استفاده می شود. با دستور بالا شما می توانید تنظیمات اولیه ی **wine** را انجام دهید. همچنین می توانید کتابخانه های ویندوزی یعنی **dll** را اضافه کنید. همچنین این کار را می توانید از طریق خط فرمان انجام دهید که در ادامه بحث خواهد شد.

### طریقه ی نصب نرم افزار با wine

برای اجرای نرم افزارهای ویندوزی با **wine** شما ۴ مرحله را در پیش دارید.

- دانلود برنامه از سایت موردنظر (مثال: **download.com**)
- قرار دادن در پوشه ی موردنظر (مثال: پوشه ی خانگی)



- رفتن به پوشه‌ی مربوطه از طریق خط فرمان با دستور `cd ~/app` (مثال: `cd`)
- اجرای نرم‌افزار ویندوزی با `wine` (مثال: `wine example.exe`)

### افزودن کتابخانه

برای افزودن کتابخانه می‌توانید از دو روش استفاده کنید.

- یا از کتابخانه‌ها که در ویندوز نصب شده‌اند استفاده کنید. (می‌توانید در این آدرس پیدا کنید  
`C:\WINDOWS\system`)
- یا از اینترنت دانلود و استفاده کنید.

سپس کتابخانه‌ها را به آدرس زیر انتقال دهید:

```
~/~wine/drive_c/windows/system
```

### اجرای چند برنامه‌ی ساده

`wine` به‌طور پیش‌فرض چند برنامه رو به‌طور پیش‌فرض دارد. در زیر به برخی از آن‌ها اشاره می‌شود.

: regedit

```
wine regedit
```

: cmd

```
wine cmd
```

: notepad

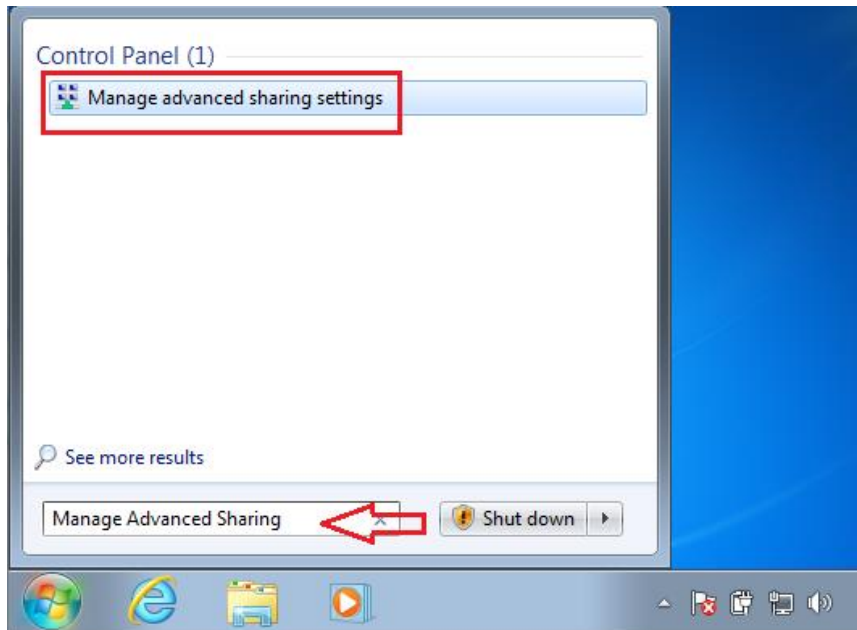
```
wine notepad
```

### اشتراک‌گذاری فایل بین اوبونتو و ویندوز

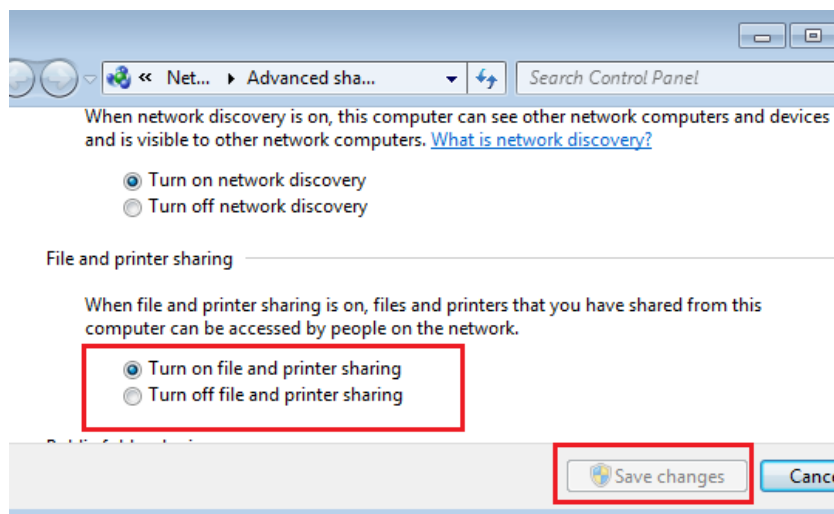
با آموزش بسیار کوتاه زیر یاد می‌گیرید که چگونه بین دو سیستم‌عامل ویندوز و اوبونتو منابع خودتان رو به اشتراک بگذارید قبل از شروع باید از اتصال دو سیستم به هم اطمینان پیدا کنید.

برای اشتراک‌گذاری منابع بین دو سیستم‌عامل ویندوز و اوبونتو مراحل زیر رو دنبال کنید :

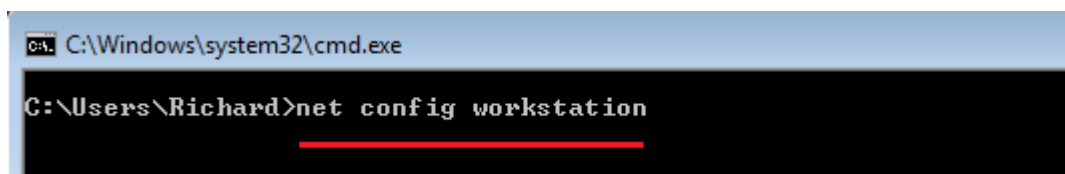
(۱) وارد سیستم‌عامل ویندوز بشوید و "Manage Advanced Sharing" را توی منوی استارت تایپ کنید و اجرا کنید



۲) تیک گزینه‌ی Turn on file and printer sharing رو بزیند .



۳) به منوی استارت برین تایپ کنید cmd تا command باز بشود و در آن تایپ کنید.



از نام سیستم و Workstation domain یک یادداشت بردارید .

```

C:\Windows\system32\cmd.exe

C:\Users\Richard>net config workstation
Computer name          \\WINDOWS7
Full Computer name    Windows7
User name              Richard

Workstation active on
NetBI_Tcpip_{32BD5B76-A291-42FC-85F5-ECD53F5E0EF8} <080027

Software version      Windows 7 Professional
Workstation domain    PENGUIN
Logon domain          Windows7
  
```

۴) حال وارد اوبونتو بشوید و ترمینال رو باز کنید و دستور زیر رو در آن وارد کنید تا برنامه‌ی Samba نصب شود.

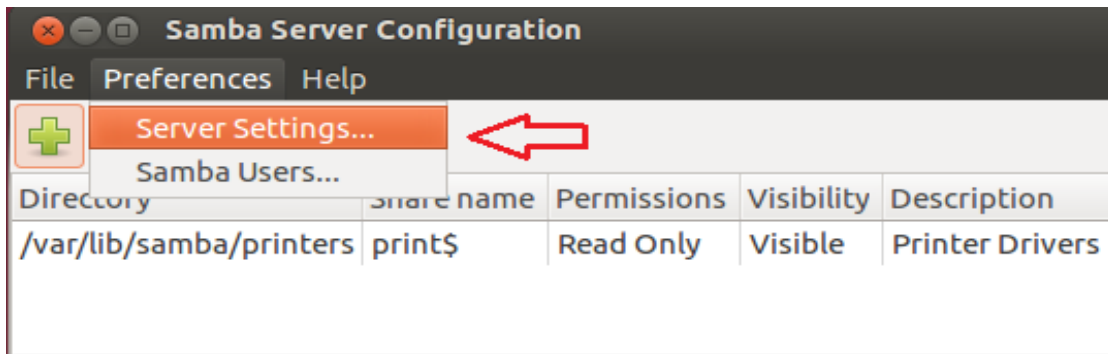
**sudo apt-get install python-glade2 system-config-samba**

۵) بعد از نصب برنامه‌ی Samba رو جست‌وجو و اجرا کنید.

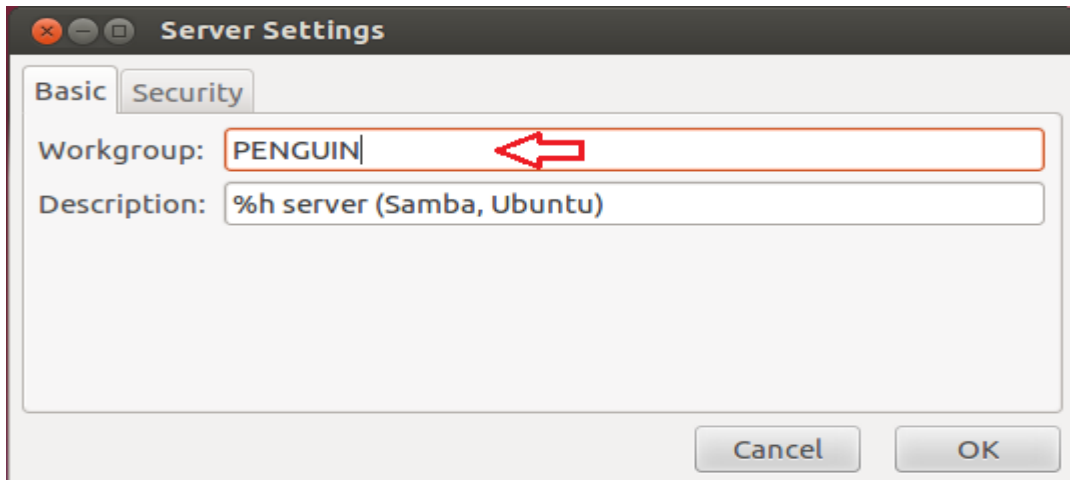


Samba کمک می‌کند تا بتوانید بین دو سیستم عامل لینوکس و ویندوز ارتباط برقرار کنید.

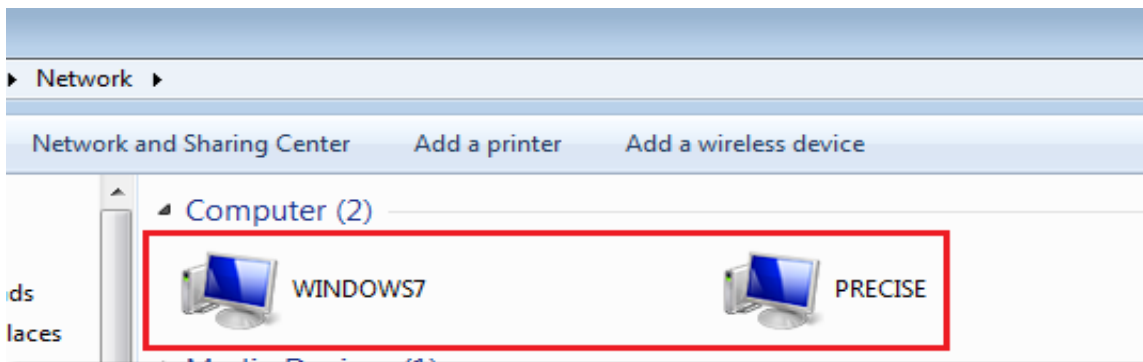
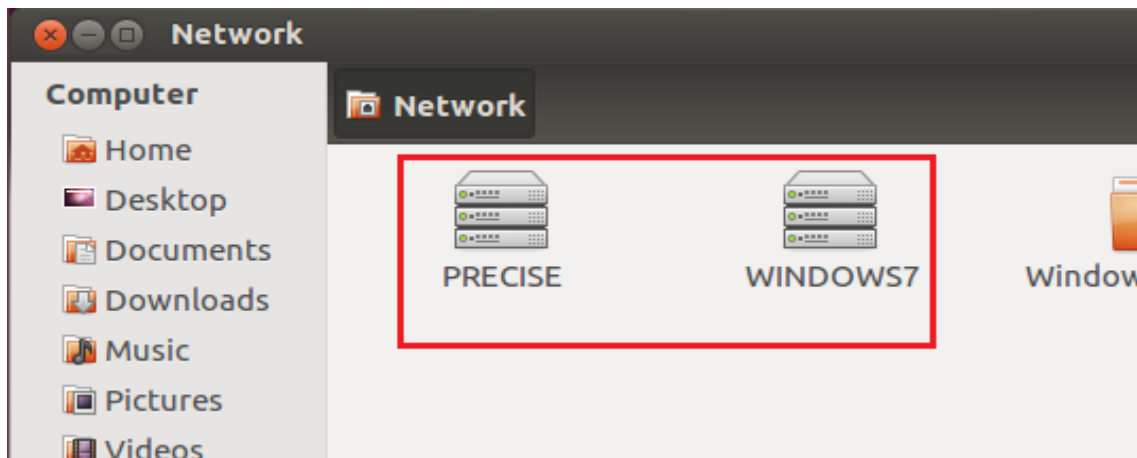
۶) به مسیر "Preferences -> Server Settings" بروید.



حالا Workgroup سیستم خودتان رو که در مرحله‌ی ۳ دیدیم رو در قسمت مربوطه بنویسید و Ok را بزنید.



۷) هر دو کامپیوتر رو ری استارت کنید و وارد Network بشوید ( برای اوبونتو توی Dash نام Network رو جست و جو کنید و برای ویندوز به My Computer برین و از منوی سمت چپ Network سیستم رو باز کنید یا توی Run آدرس IP سیستم مورد نظرتان رو وارد کنید. )



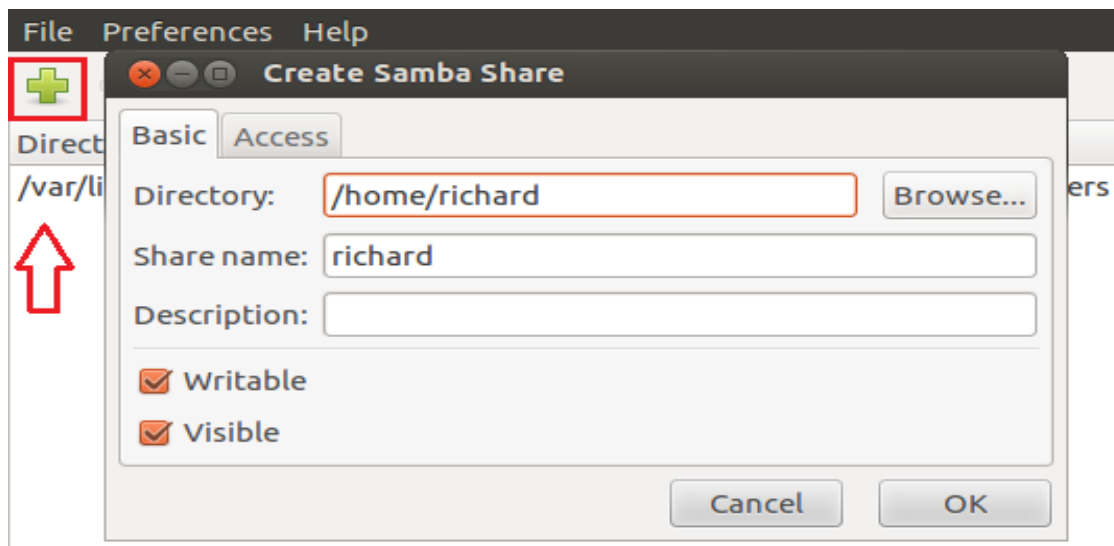
۸) توی اوبونتو بر روی منابع به اشتراک گذاشته‌ی ویندوز کلیک کنید و user , password و ویندوزتان رو وارد کنید. برای تنظیمات Samba به ترمینال اوبونتو برین و دستور زیر رو وارد کنید.

**sudo smbpasswd -a username**

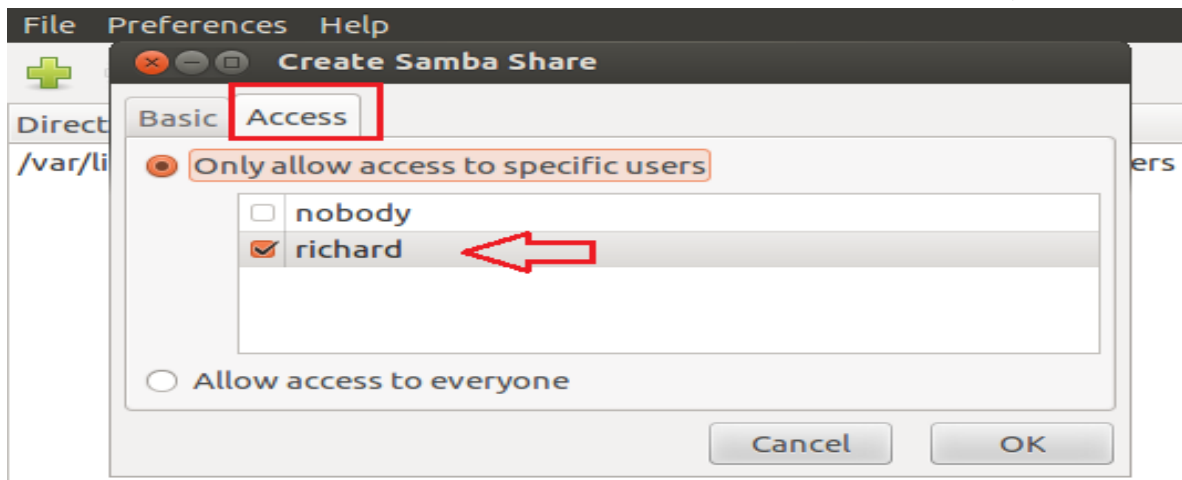
به جای `username` نام کاربری سیستم خودتان رو وارد کنید و حالا باید پسوردی به `Samba` بدین تا داخل ویندوز بتوانید منابع به اشتراک گذاشته‌ی اوبونتو رو ببینید .

```
richard@Precise: ~  
richard@Precise:~$ sudo smbpasswd -a richard
```

۹) برای به اشتراک گذاری فایل‌ها در اوبونتو برنامه‌ی `Samba` رو باز کنید و بر روی گزینه‌ی `Add` که به شکل یک مربع است کلیک کنید و فایل موردنظرتان را انتخاب کنید و `ok` کنید تا در شبکه به اشتراک گذاشته شود.



و بعد وارد تب `Access` شوید و میزان دسترسی برای فایل‌ها رو مشخص کنید که با انتخاب نام کاربری اوبونتو تنها در صورت داشتن رمز `Samba` سیستم‌های دیگر شبکه می‌توانند به اطلاعات شما دسترسی داشته باشند.



## شبکه محلی دو ویندوز در VmWare

ابتدا در محیط vmware دو windows xp نصب کرده .

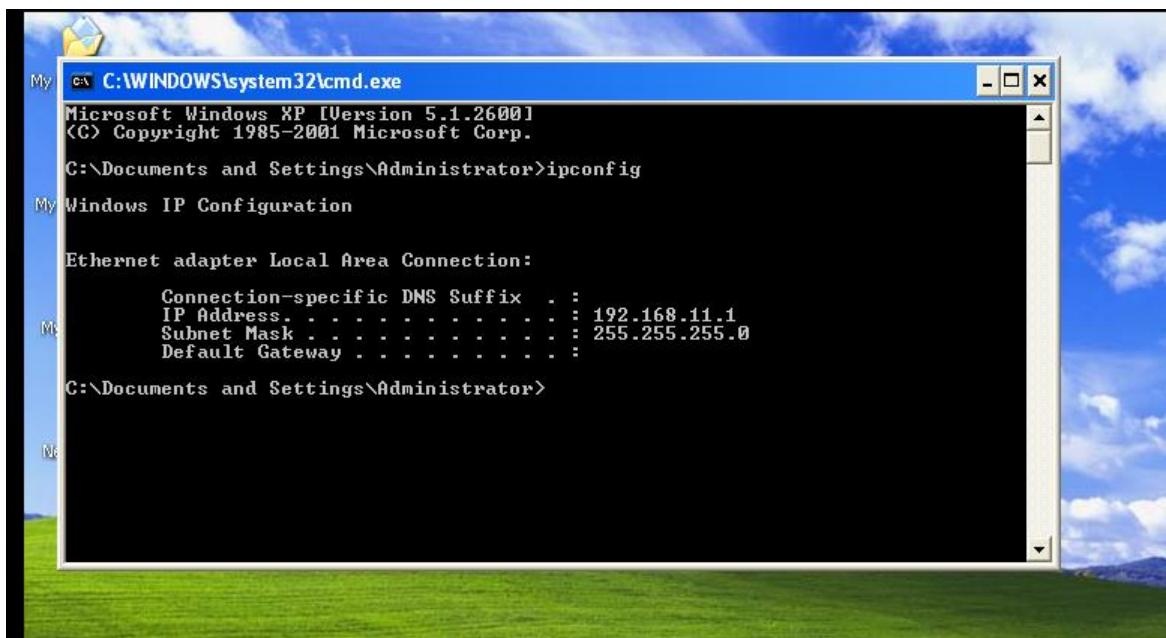
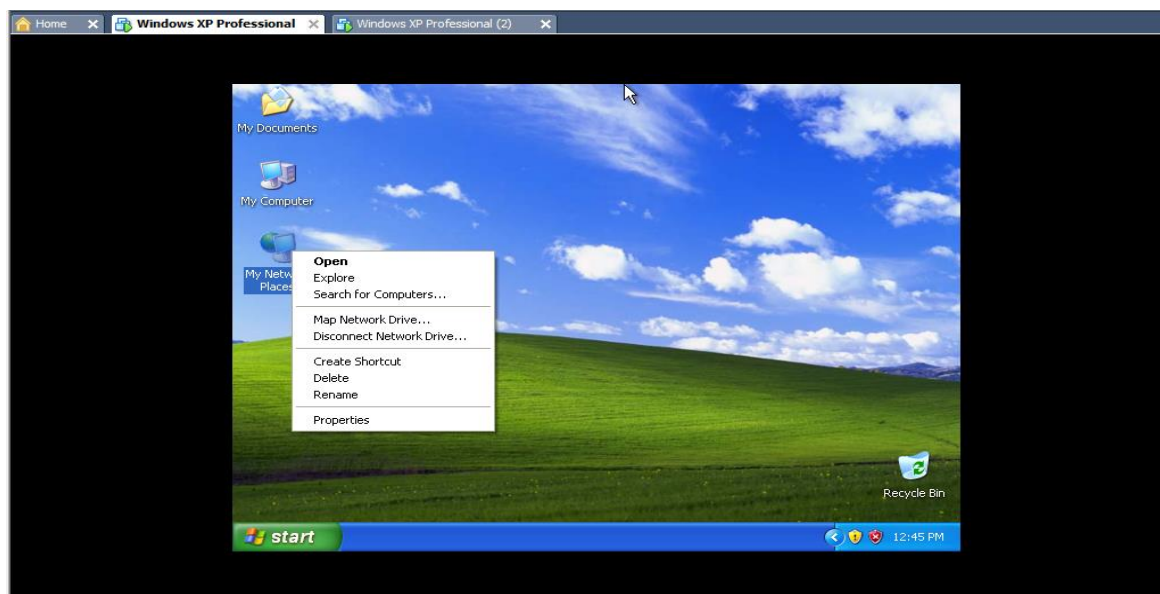
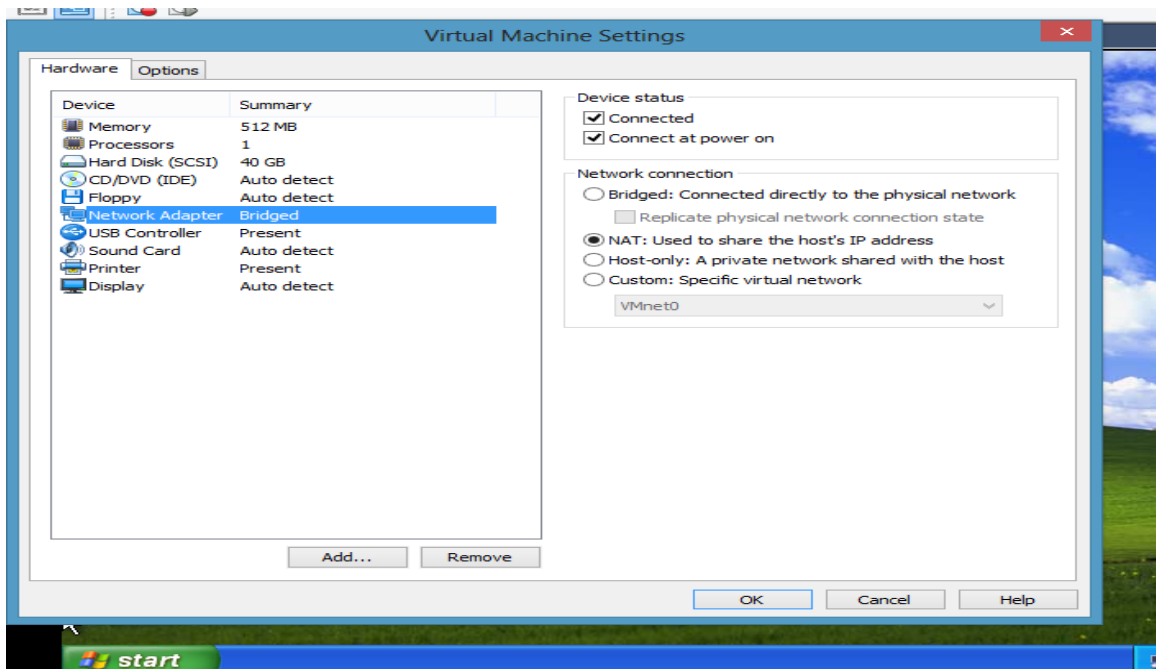
در هر دو ویندوز طبق شکل زیر تنظیمات کارت شبکه را انجام می دهیم.

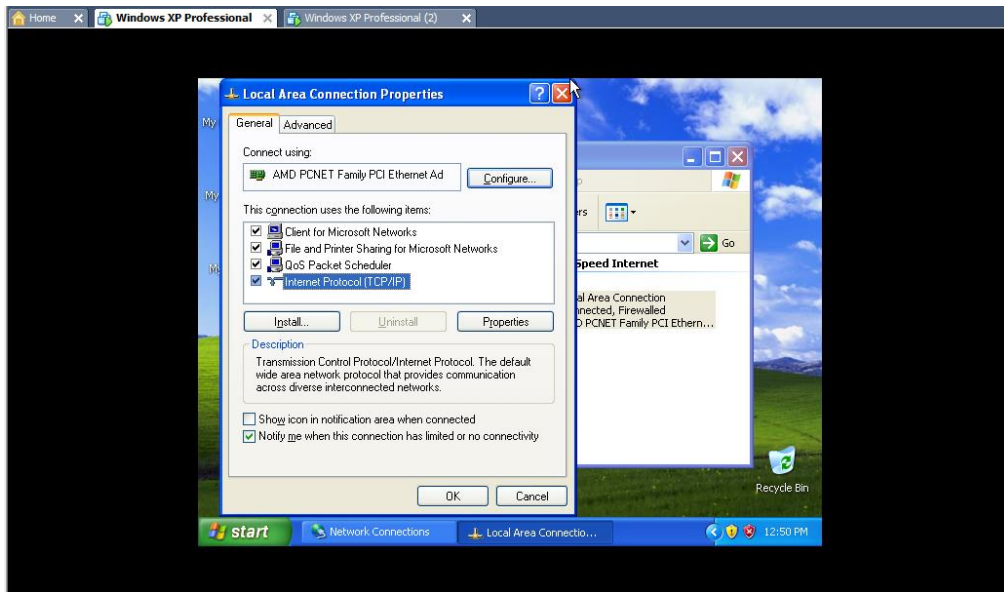
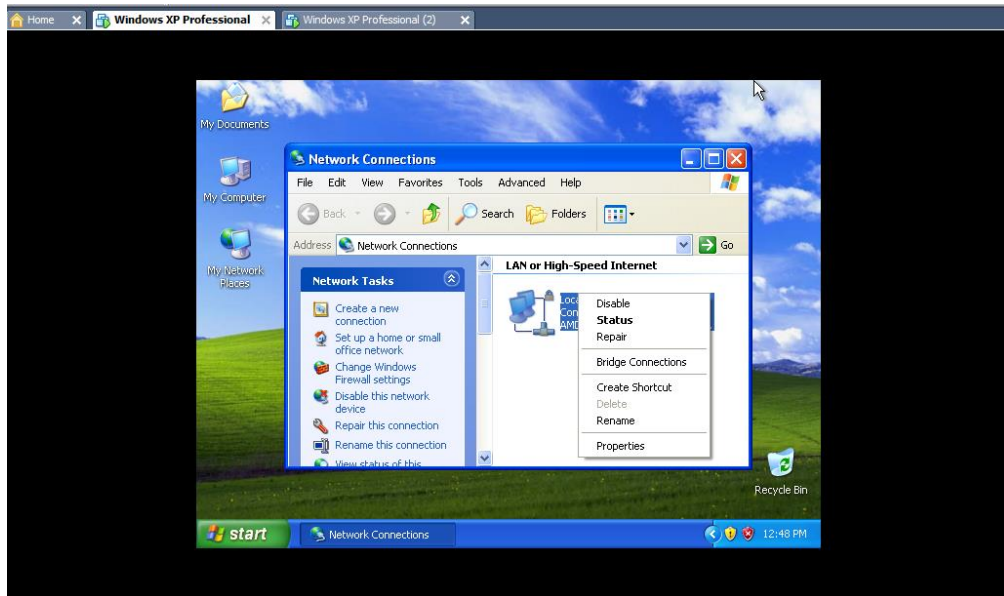


و OK را می زنیم این تنظیمات در هر دو ویندوز باید انجام شود.  
در هر دو ویندوز طبق مسیر زیر برای به دست آوردن IP عمل می کنیم.

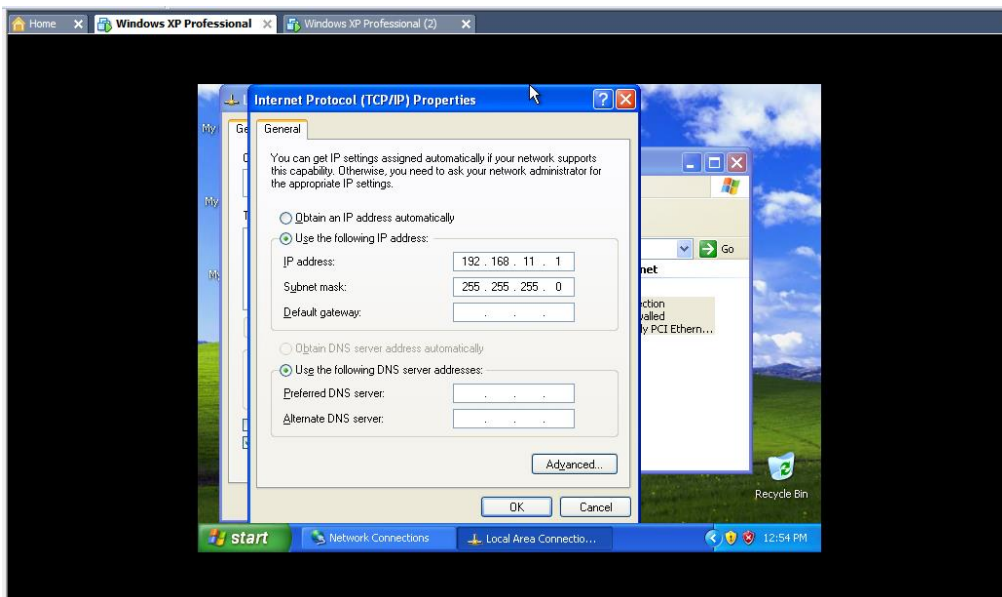
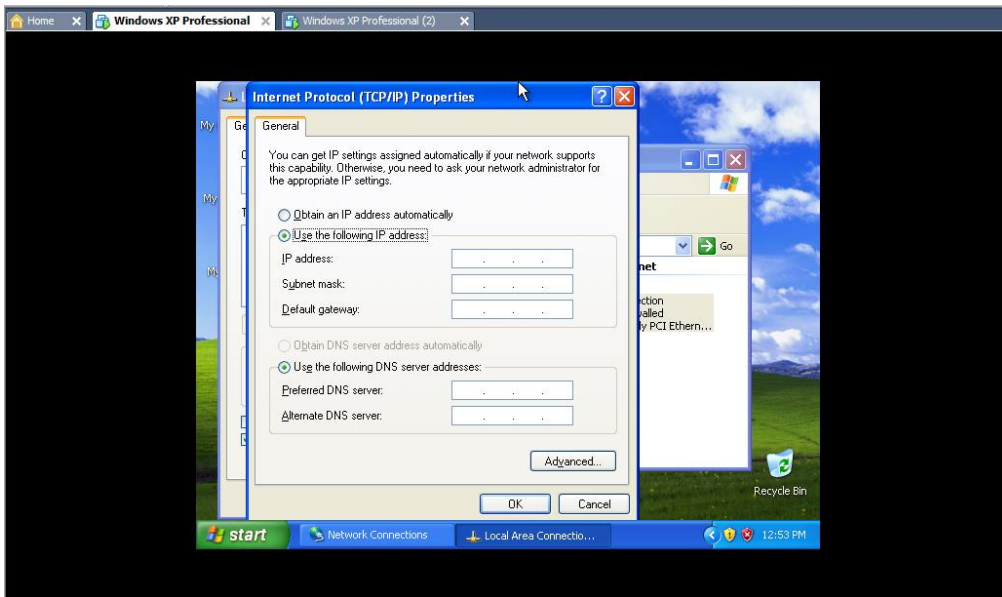
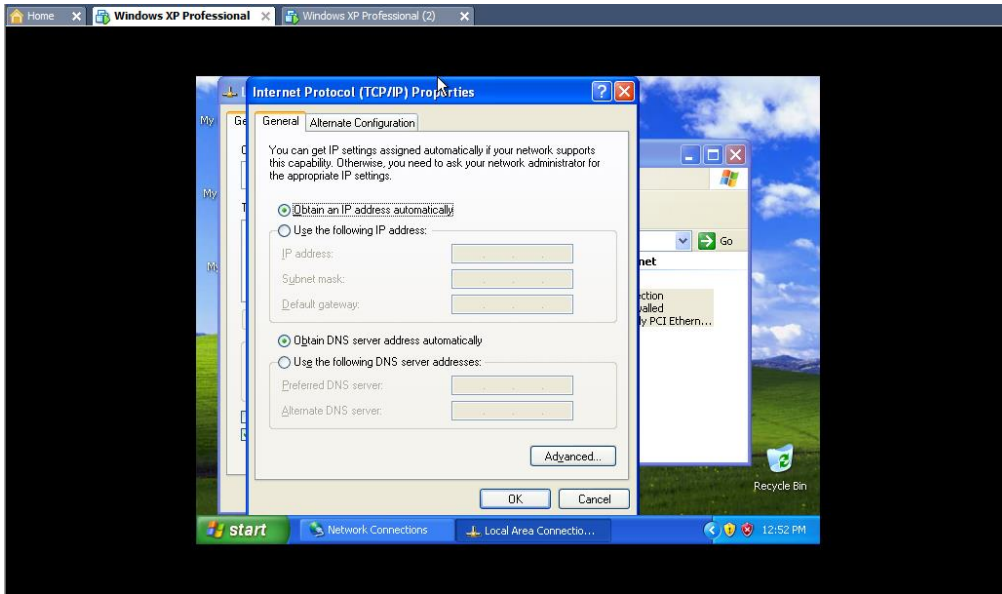
*Run → cmd → ipconfig*

در هر دو ویندوز روی My Network Places راست کلیک کرده و گزینه Properties را انتخاب می کنیم. پنجره Network Connection باز می شود. در این پنجره روی آیکن Local Area Connection راست کلیک کرده و گزینه Properties را می زنیم. در تب General روی گزینه Internet Protocol (TCP/IP) یکبار کلیک کرده و دکمه Properties را می زنیم. سپس تیک use the following IP Address را فعال می کنیم و در قسمت IP Address ، IP موردنظر خود را وارد کرده و OK می کنیم.

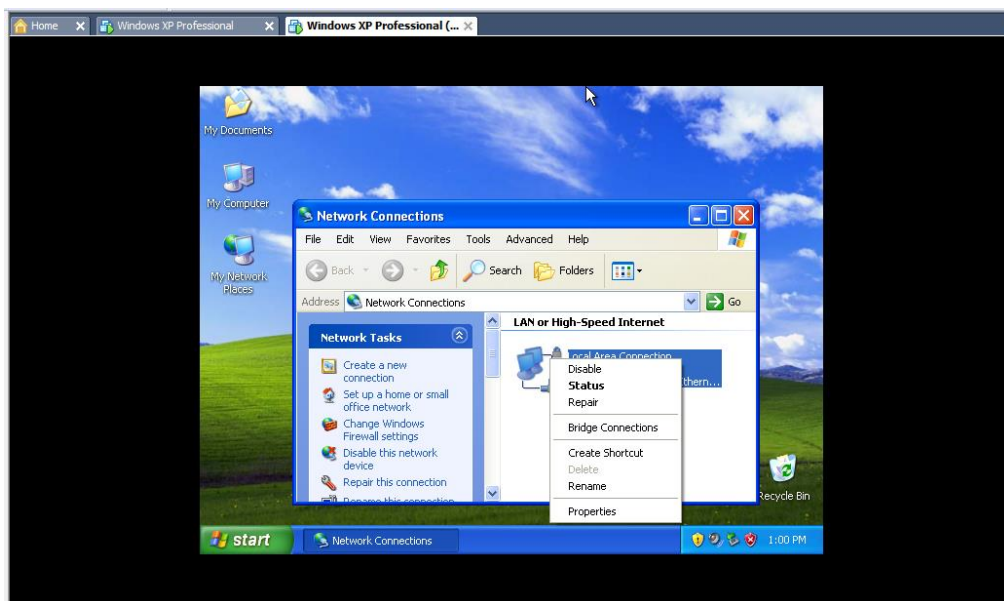
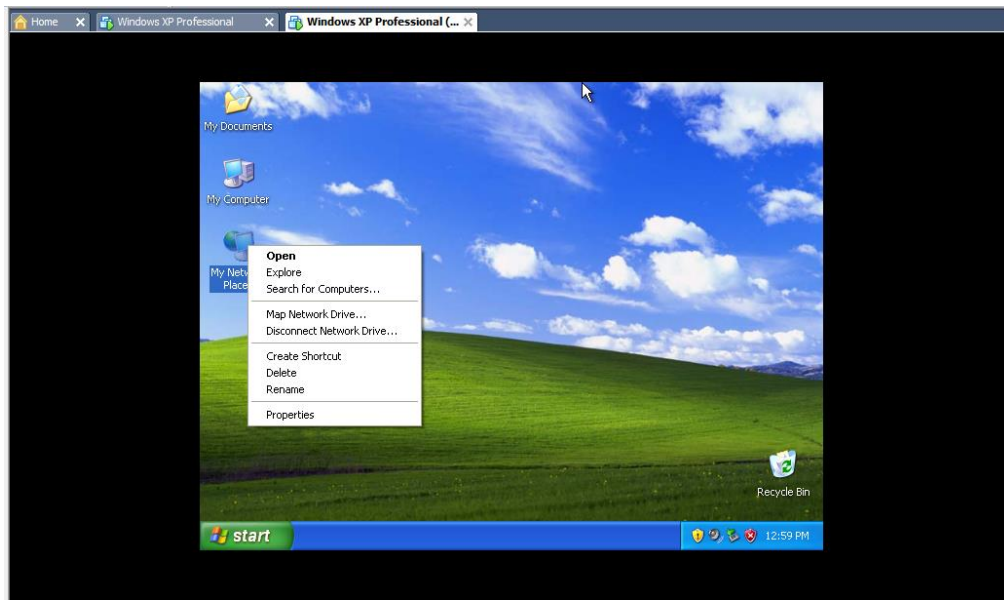


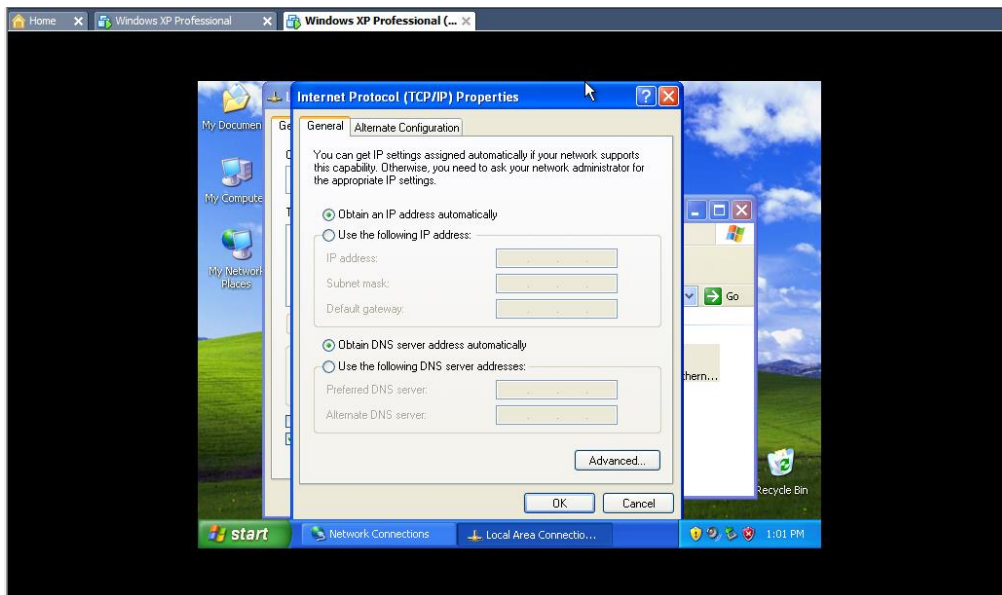
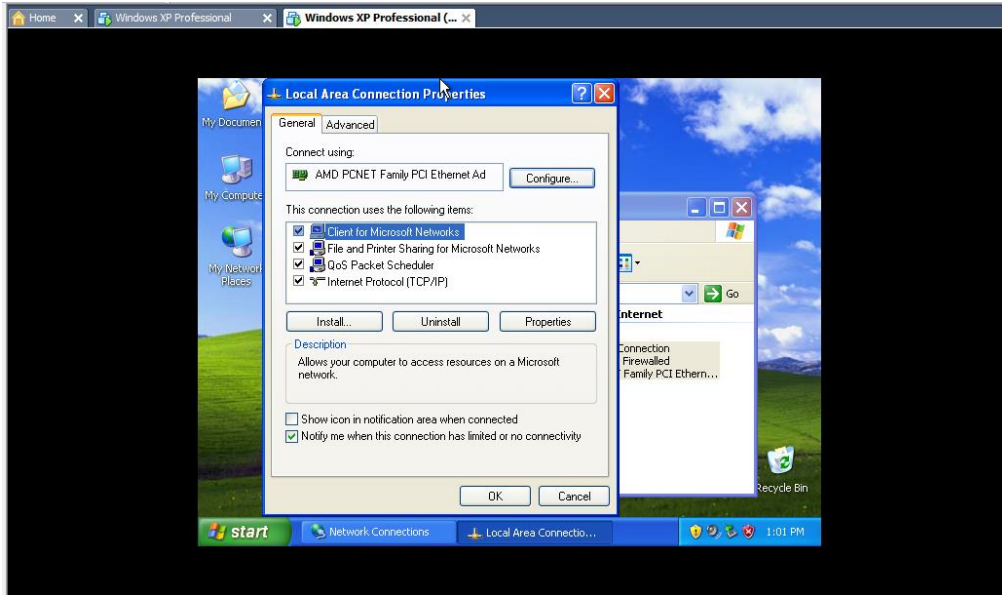


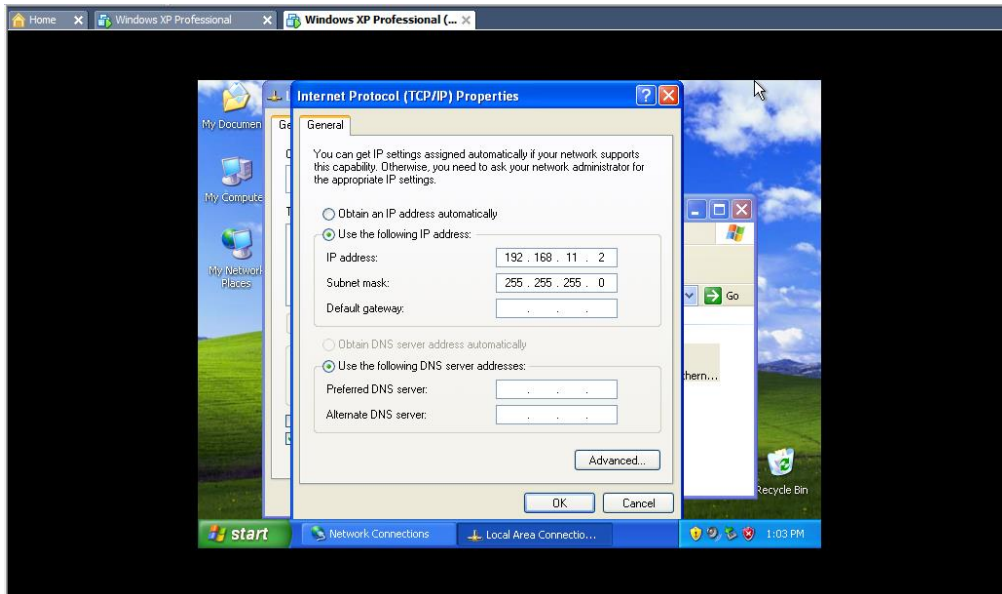




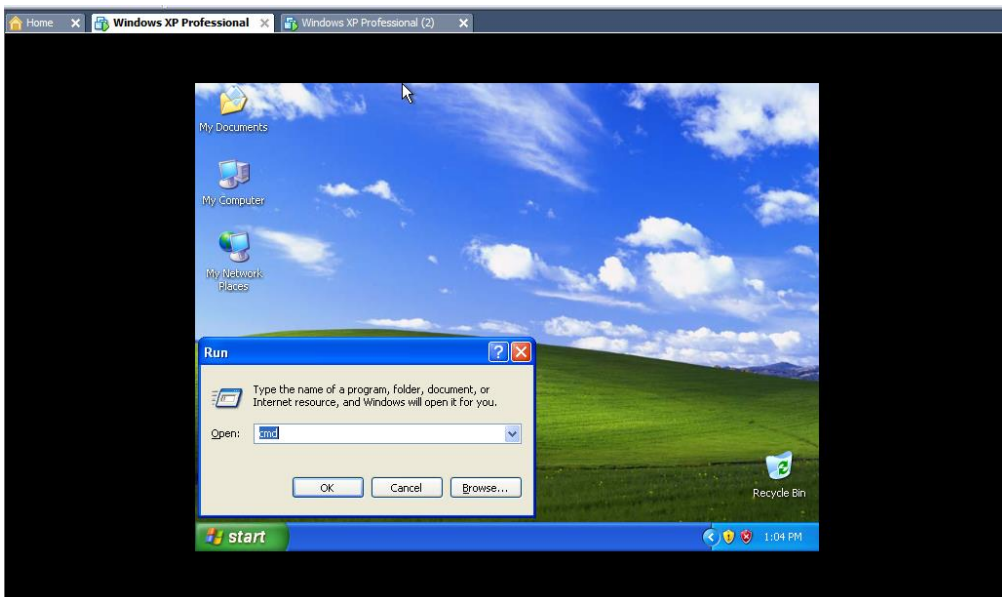
تا اینجا به ویندوز اول ، IP اختصاص داده ایم .  
همین کار را برای ویندوز دیگر انجام می دهیم .

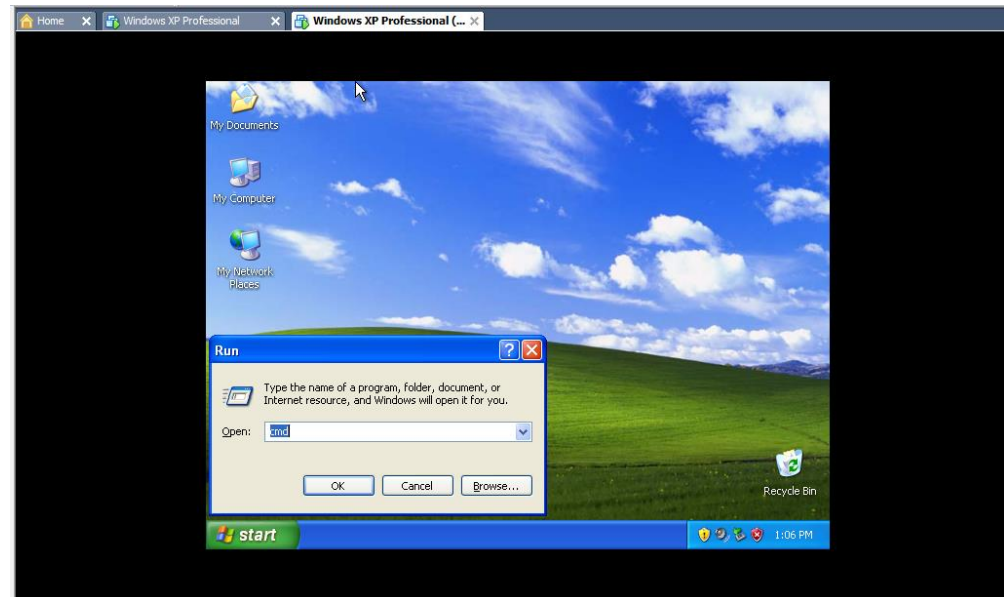
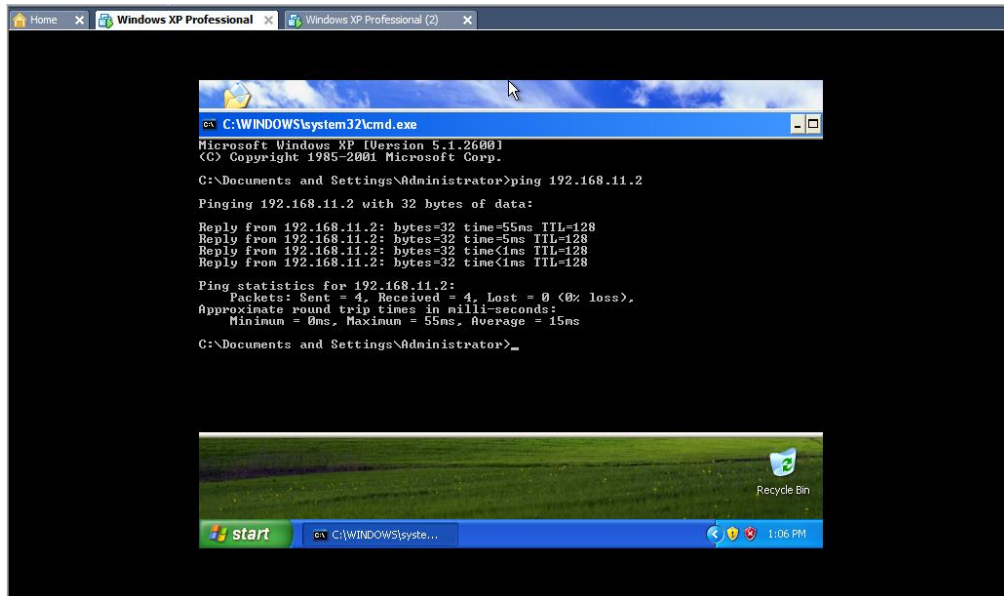
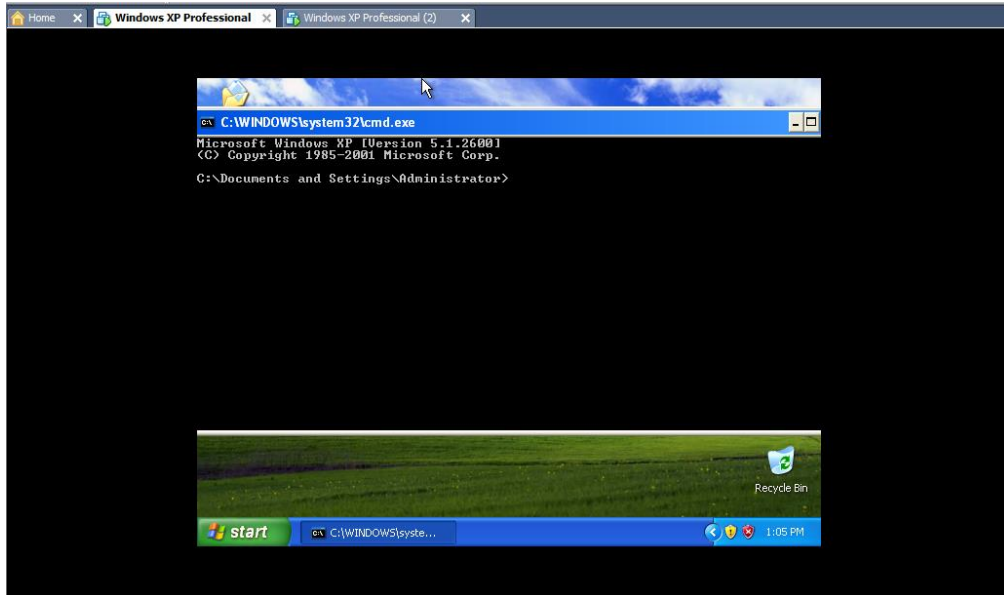


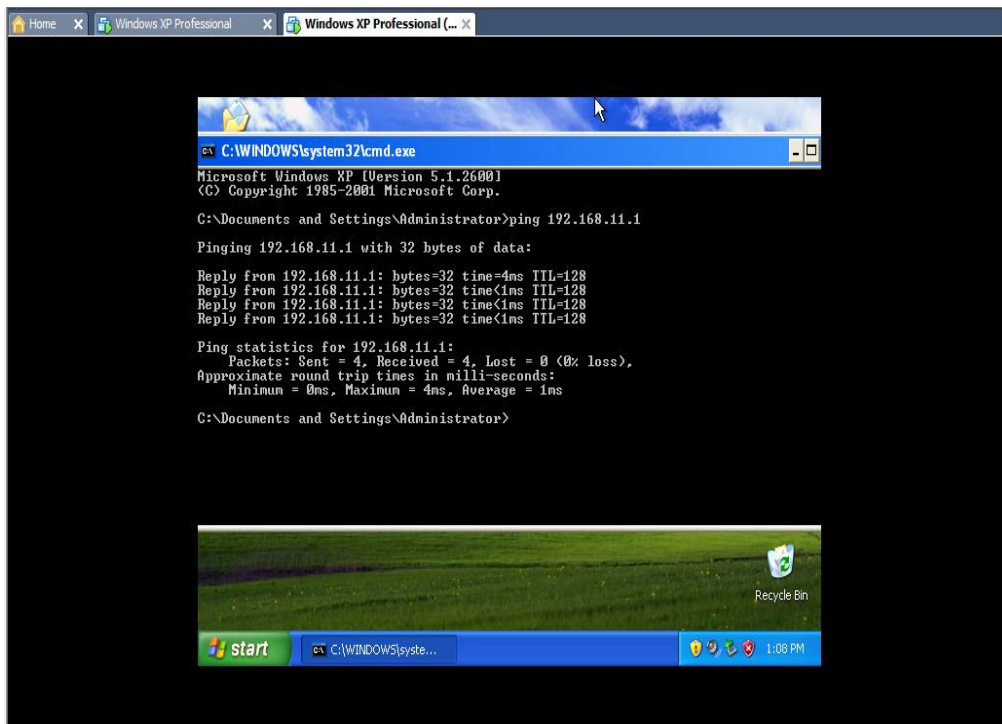
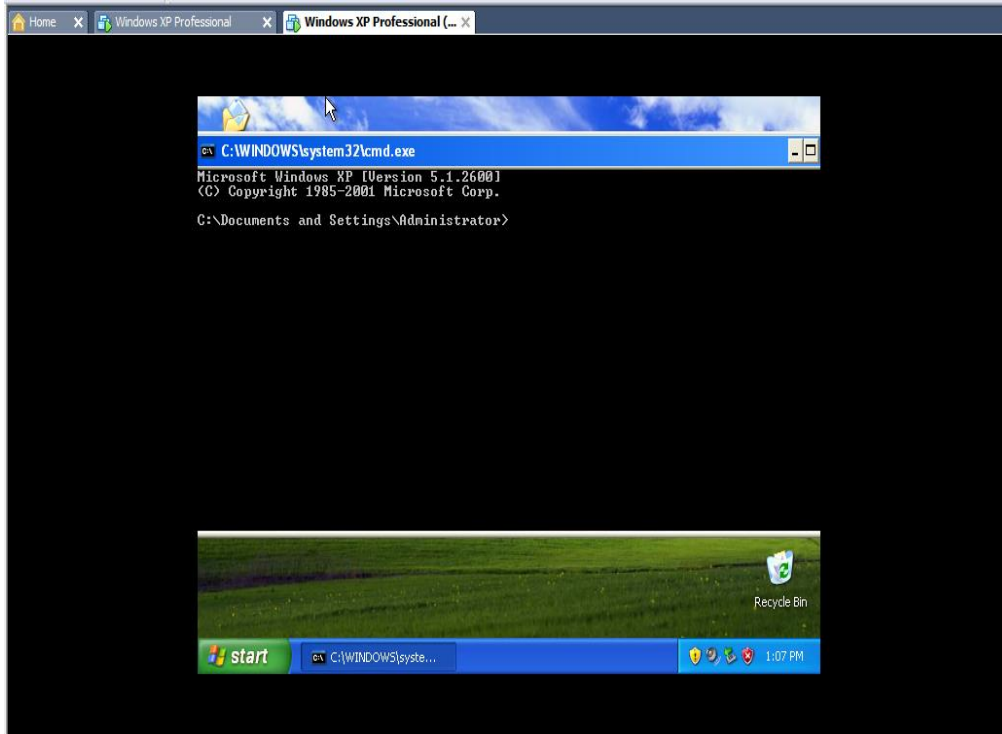


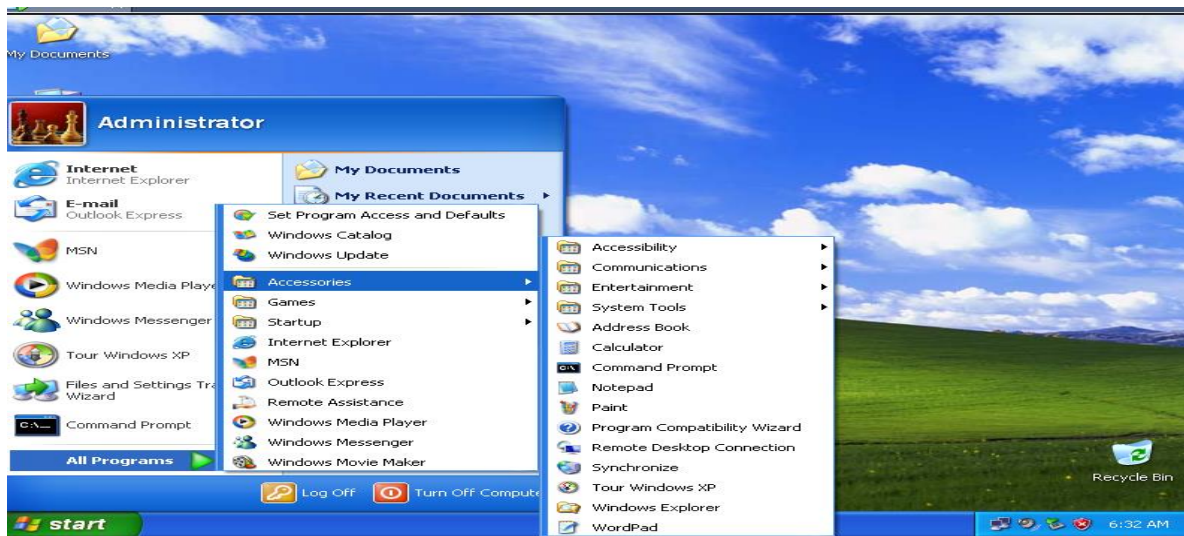


حال به ویندوز دوم نیز IP اختصاص داده شد.  
عمل Ping را انجام می دهیم تا دو سیستم یکدیگر را بشناسند.

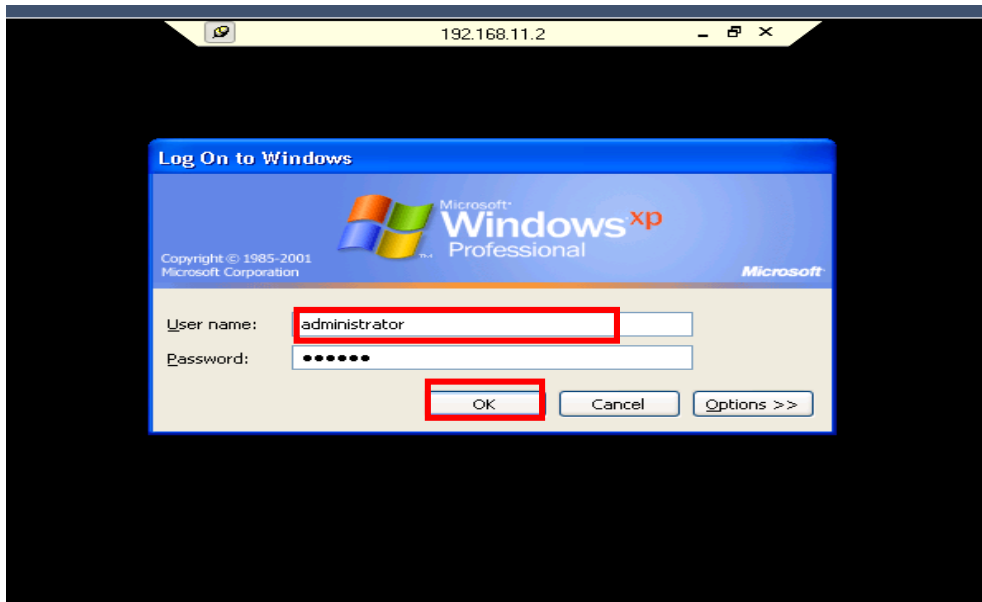
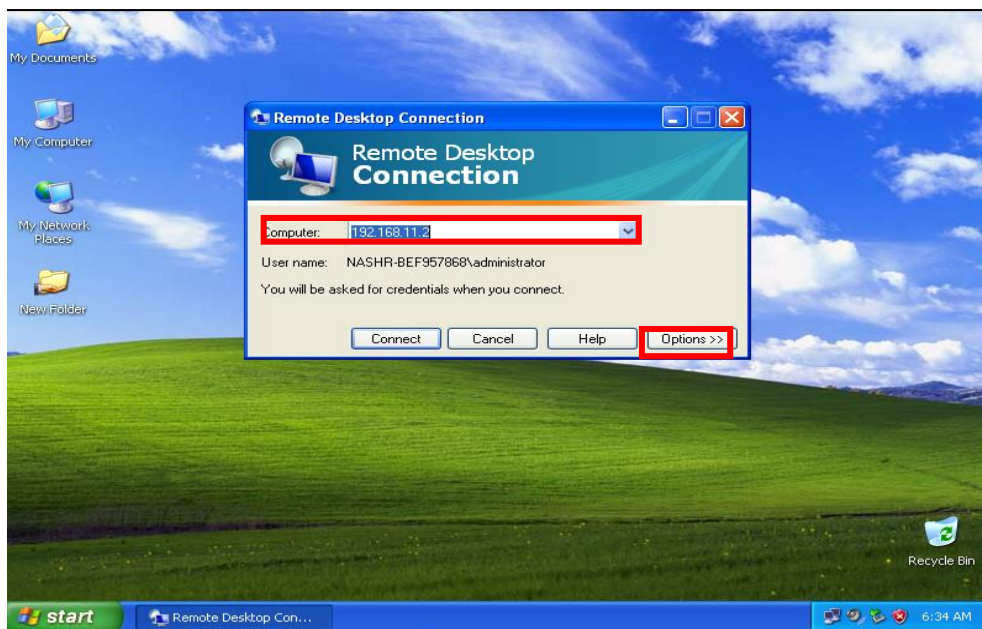






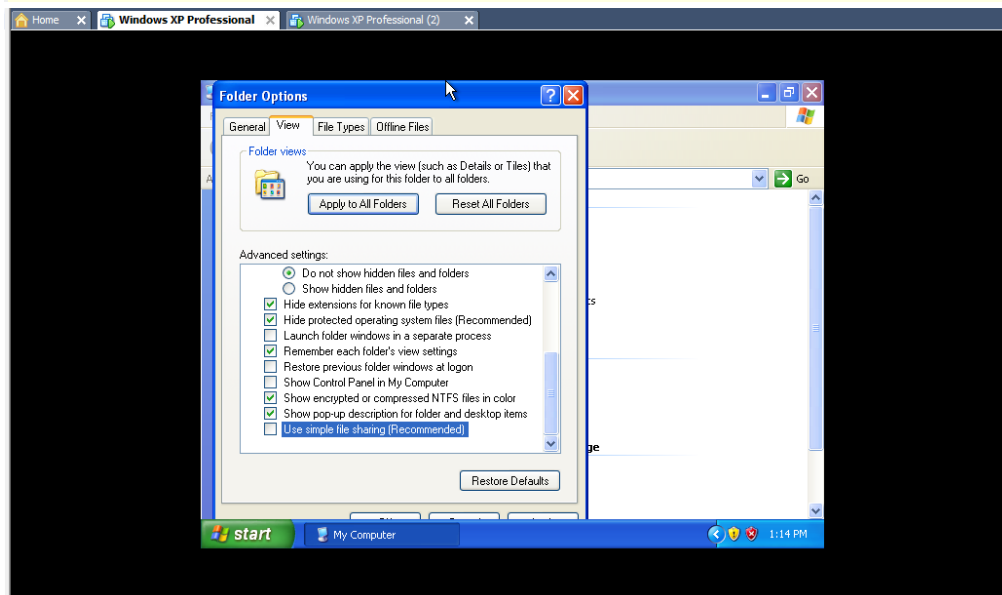
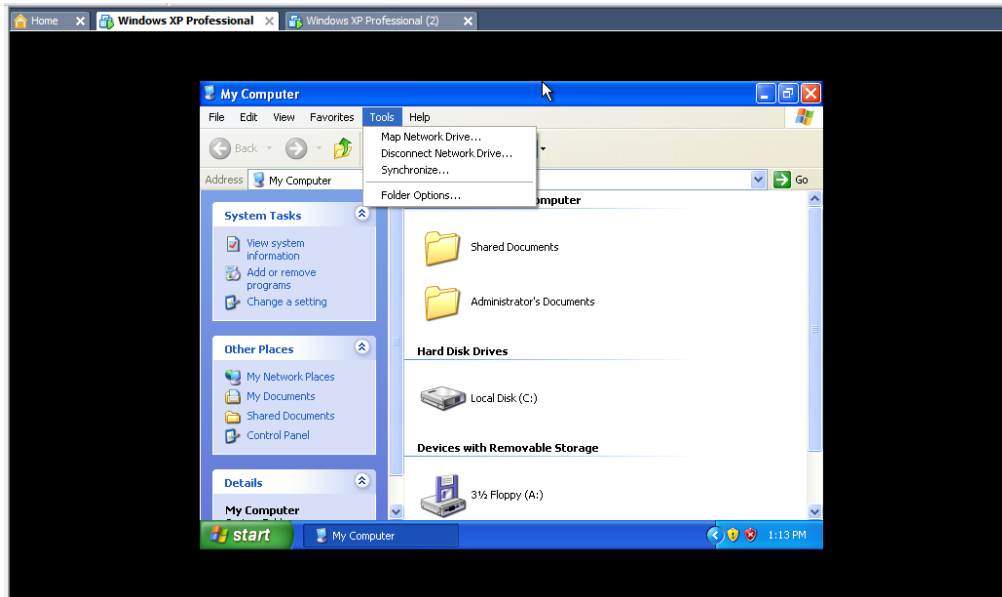


اگر با عمل ping ویندوزها همدیگر را نشناختند آن‌ها را به هم دیگر طبق مسیر زیر ریموت کنید و سپس دوباره عمل ping انجام دهید تا همدیگر را بشناسند.  
در پنجره باز شده IP ویندوز دوم را وارد می‌کنیم.

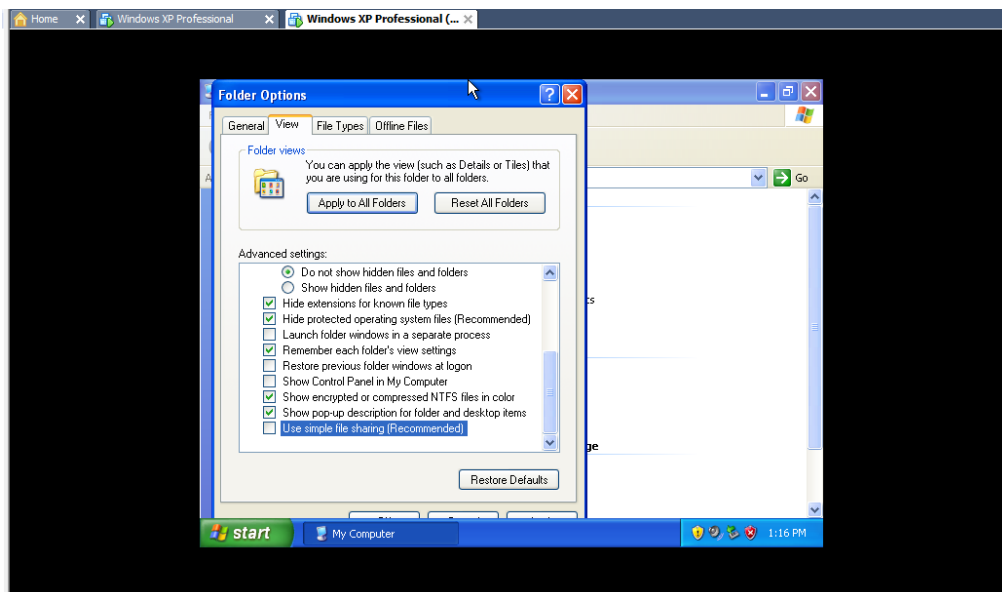
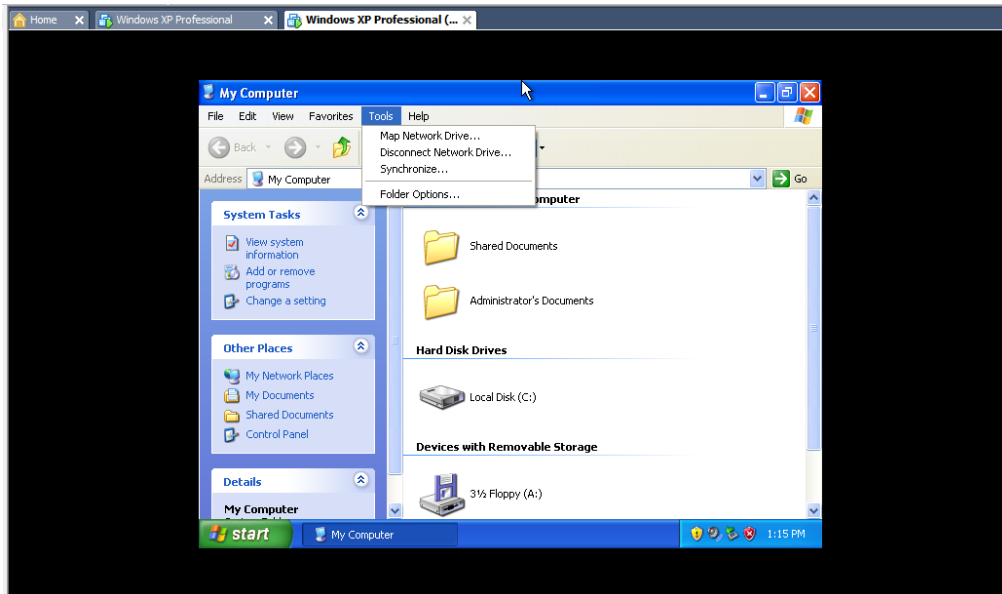


برای اشتراک فایل بین دو ویندوز

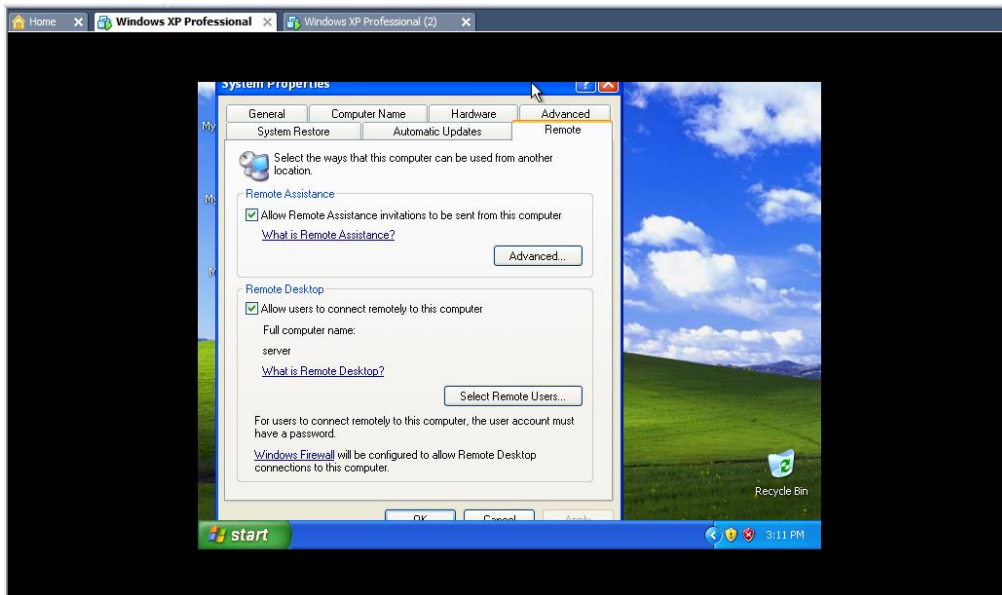
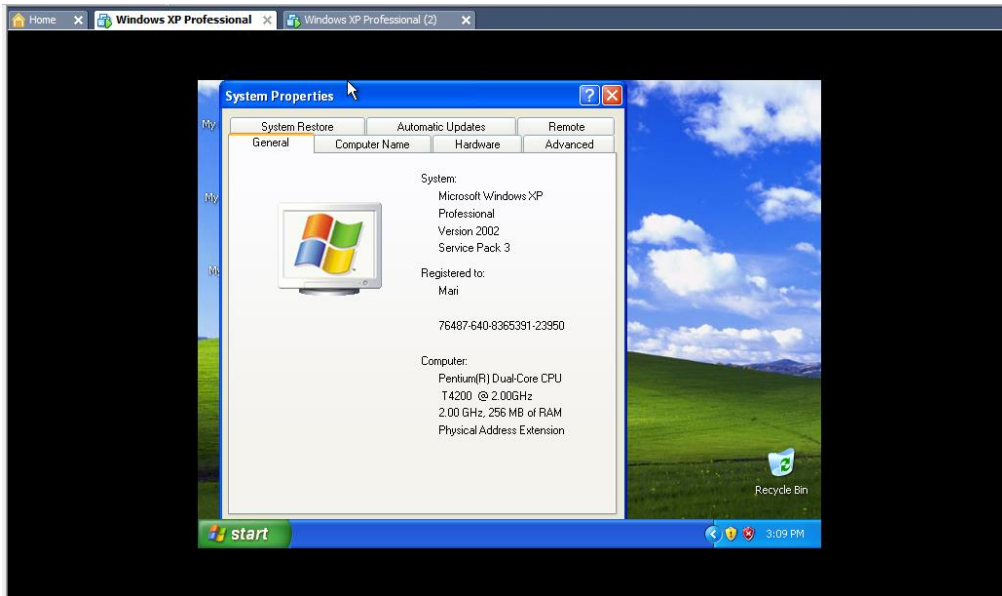
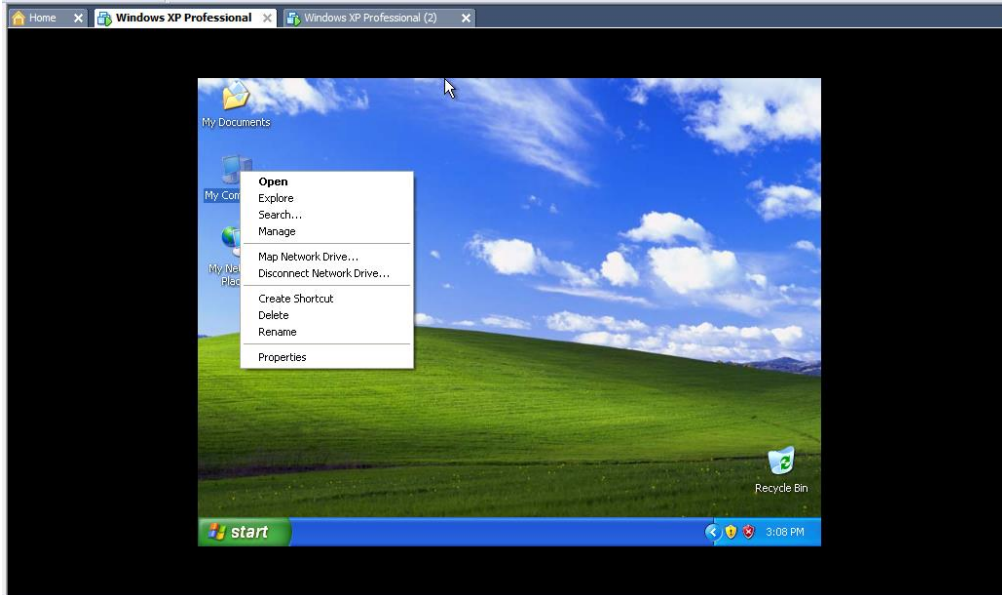
در هر دو ویندوز وارد Folder Option می شویم و تیک گزینه آخر در تب view را برمی داریم. این کار باعث می شود برای به اشتراک گذاشتن هر قسمتی که مدنظر ماست برای هر کاربر حق دسترسی قرار دهیم. برای هر دو ویندوز این کار را انجام می دهیم.

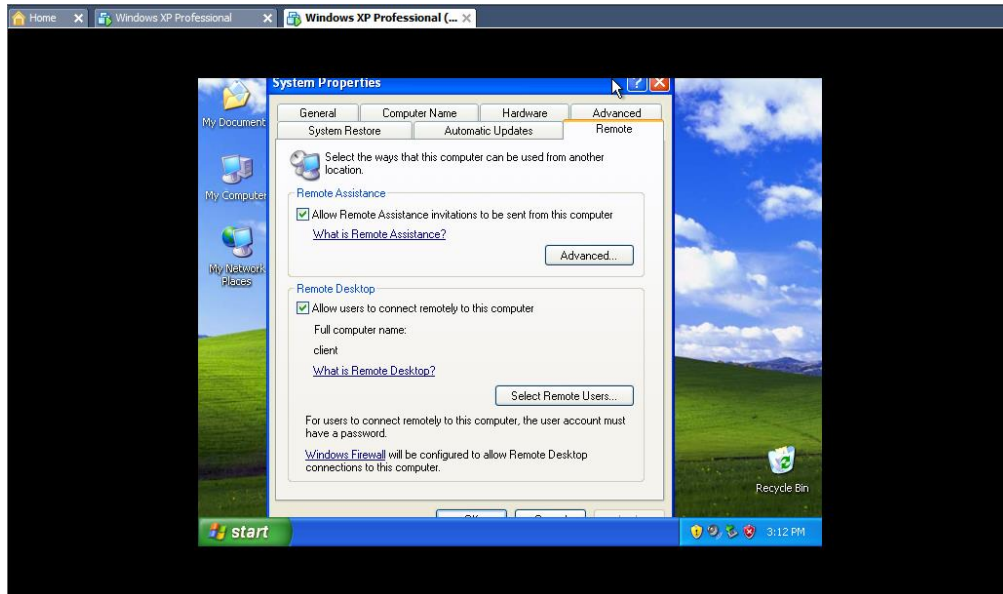




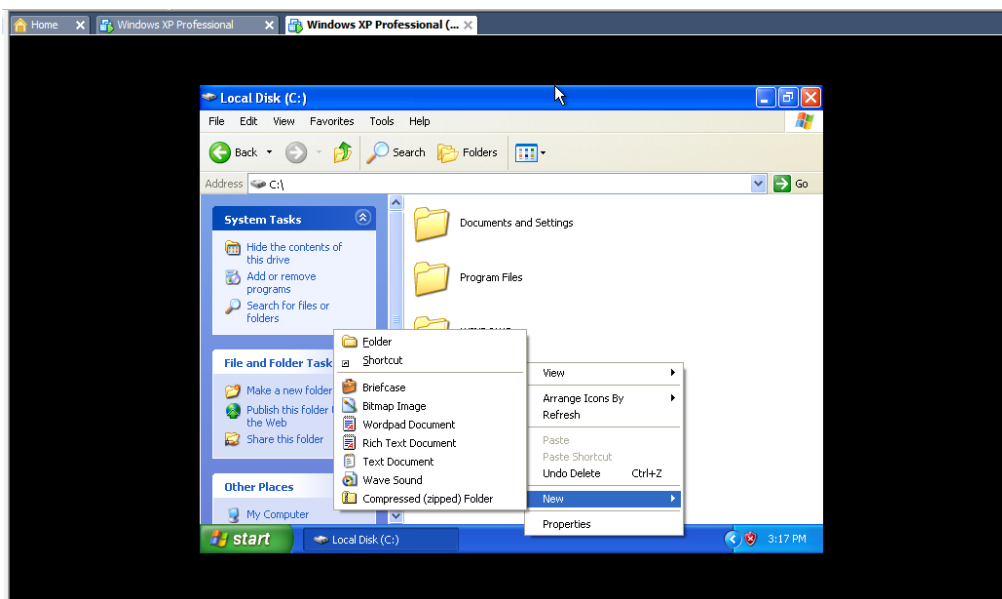


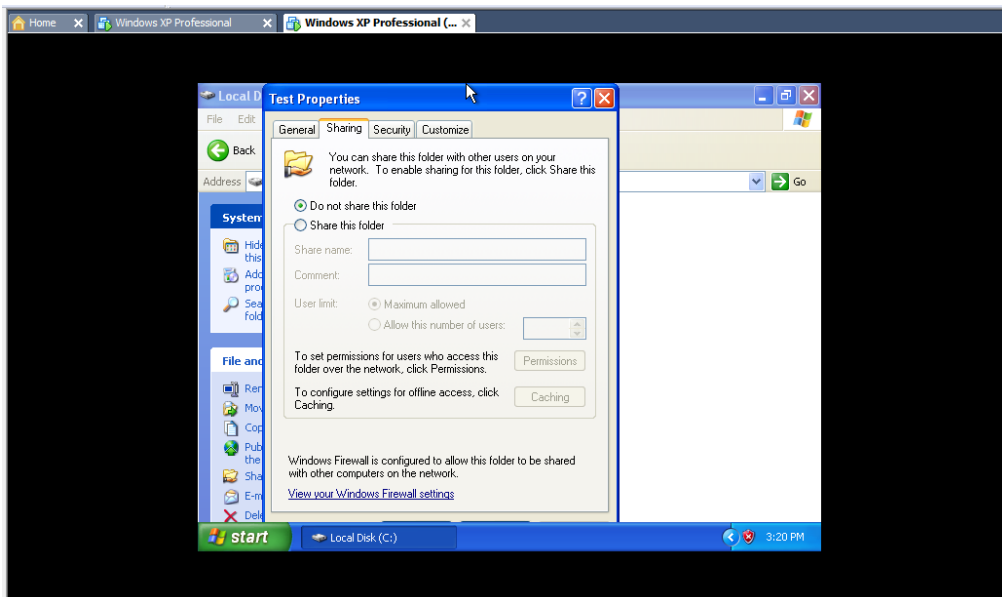
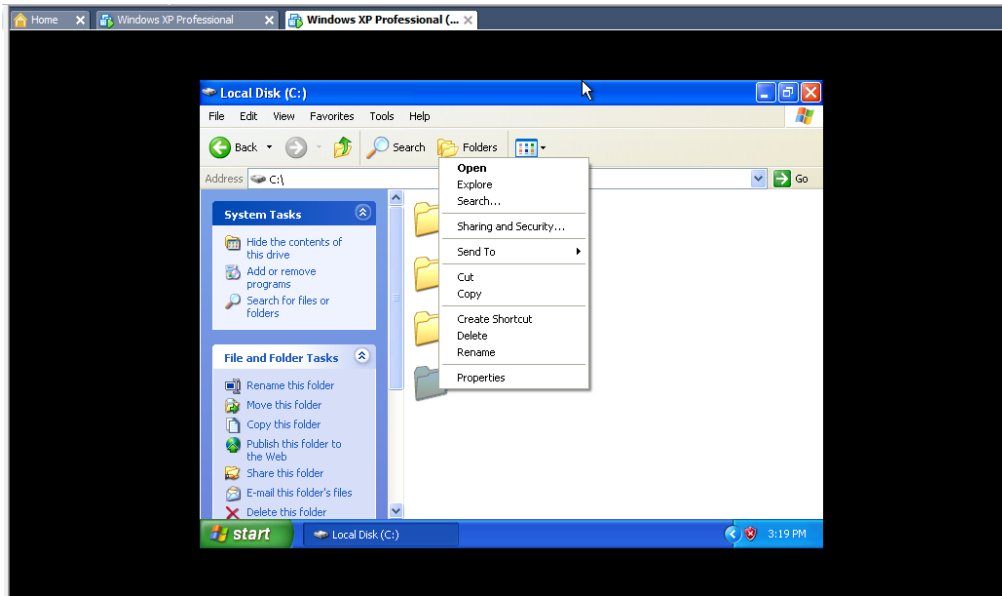
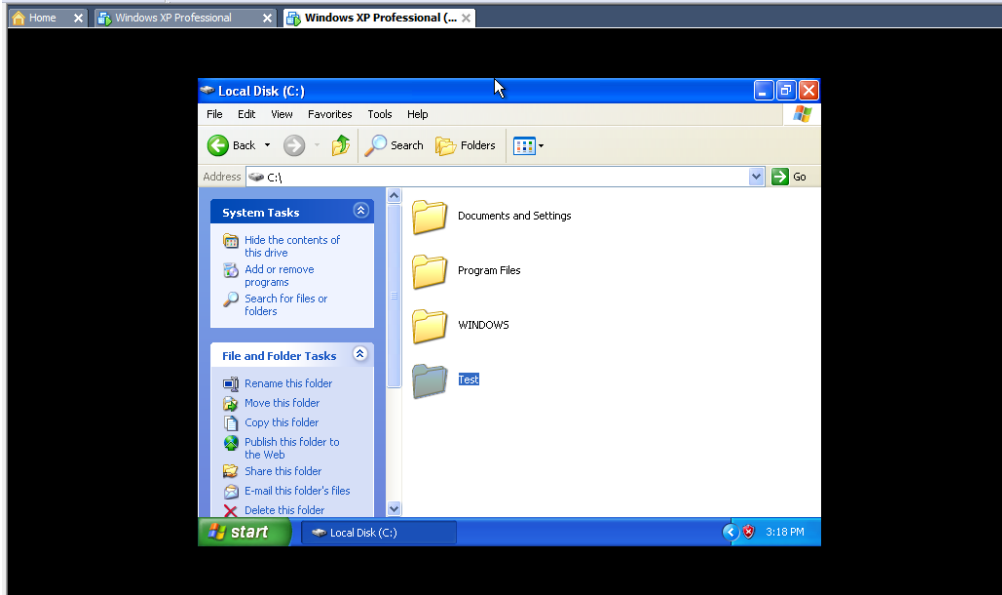
روی My computer راست کلیک کرده و گزینه Properties را انتخاب کنید سپس وارد تب Remote شده و تیک Remote Desktop را انتخاب کنید. با این کار می‌توانید هر User را هر زمان که نیاز باشد روی سیستم خود داشته باشید.

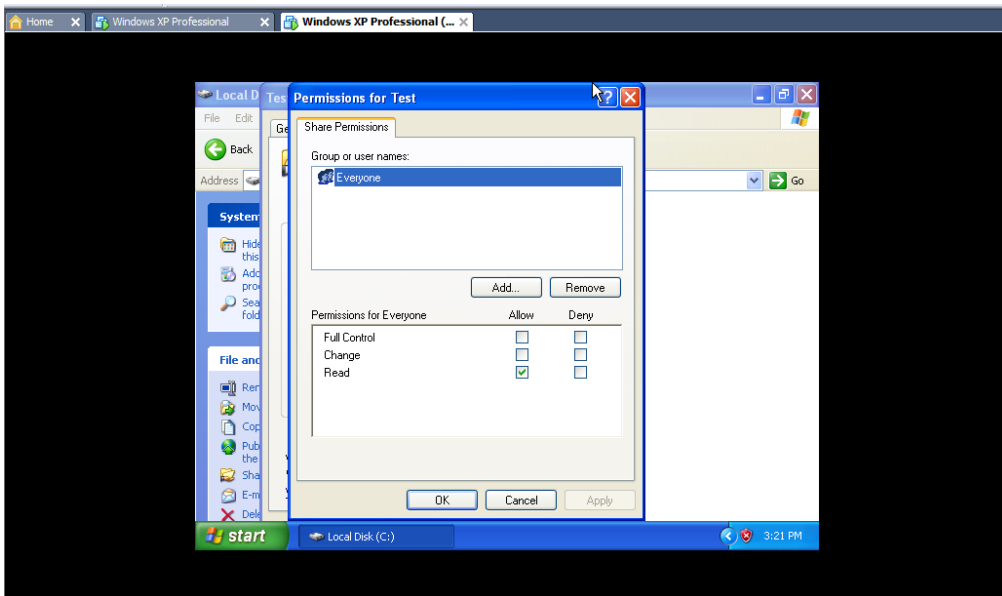
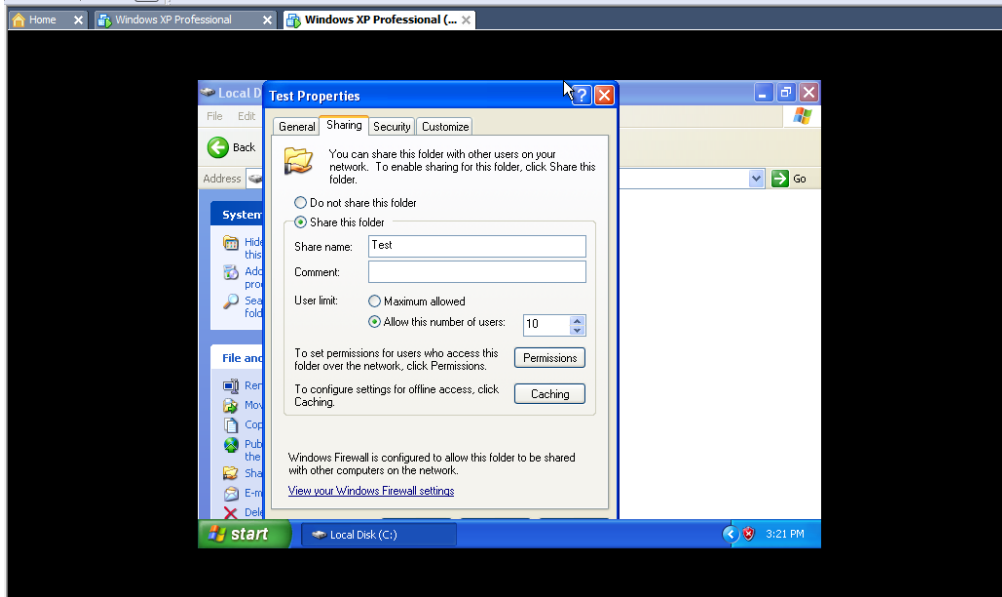




در درایو C یک New Folder ساخته و روی آن عمل share را انجام می‌دهیم. روی آن فولدر کلیک راست کرده و گزینه sharing and security را انتخاب کنید. پنجره موردنظر که باز شد تیک دوم را فعال نمایید و دکمه permission را بزنید. در پنجره باز شده every one را بسته به نیاز و دلخواه Admin حق دسترسی متفاوت دهید.









## فصل سوم (لینوکس)

در سال ۱۹۹۱ در حالی که جنگ سرد رو به پایان می‌رفت و صلح در افق‌ها هویدا می‌شد در دنیای کامپیوتر آینده بسیار روشنی دیده می‌شد. با وجود قدرت سخت‌افزارهای جدید محدودیت‌های کامپیوترها رو به پایان می‌رفت ولی هنوز چیزی بود و این چیزی نبود جز فقدان عمیق در حیطه سیستم‌های عامل داس، امپراطوی کامپیوترهای شخصی را در دست داشت. سیستم‌عاملی استخوانی که با قیمت ۵۰۰۰۰ دلار از یک هکر سیاتلی توسط بیل گیتز (Bill Gates) خریداری شده بود و با یک استراتژی تجاری هوشمند به تمام گوشه‌های جهان رخنه کرده بود.

کاربران PC انتخاب دیگری نداشتند. کامپیوترهای اپل مکینتاش بهتر بودند ولی قیمت‌های نجومی آن‌ها را از دسترس اکثر افراد خارج می‌ساخت.

خیمه‌گاه دیگری دنیای کامپیوترها دنیای یونیکس بود. ولی یونیکس به خودی خود بسیار گران‌قیمت بود. آن قدر گران‌قیمت کاربران کامپیوترهای شخصی جرات نزدیک شدن به آن را نداشتند. کد منبع یونیکس که توسط آزمایشگاه‌های بل بین دانشگاه‌ها توزیع شده بود محتاطانه محافظت می‌شد تا برای عموم فاش نشود. برای حل شدن این مسئله هیچ‌یک از تولیدکنندگان نرم‌افزار راه حلی ارائه ندادند.

به نظر می‌رسد این راه حل به صورت سیستم‌عامل MINIX ارائه شد. این سیستم‌عامل که از ابتدا توسط اندرو اس تانباوم (Andrew S tanenbaum) پروفیسور هلندی نوشته شده بود به منظور تدریس عملیات داخلی یک سیستم‌عامل واقعی بود. این سیستم‌عامل برای اجرای روی پردازنده‌های ۸۰۸۶ اینتل طراحی شده بود و به زودی بازار را اشباع کرد. به عنوان یک سیستم‌عامل MINIX خیلی خوب نبود. ولی مزیت اصلی آن در دسترس بودن کد منبع آن بود. هر کس که کتاب سیستم‌عامل تانباوم را تهیه می‌کرد به ۱۲۰۰۰ خط کد نوشته شده به زبان C و اسمبلی نیز دسترسی پیدا می‌کرد.

برای نخستین بار یک برنامه‌نویس یا هکر مشتاق می‌توانست کد منبع سیستم‌عامل را مطالعه کند.

چیزی که سازندگان نرم‌افزارها آن را محدود کرده بودند. یک نویسنده بسیار خوب یعنی تانباوم باعث فعالیت مغزهای متفکر علوم کامپیوتری در زمینه بحث و گفتگو برای ایجاد سیستم‌عامل شد. دانشجویان کامپیوتر در سرتاسر دنیا با خواندن کتاب و کدهای منبع سیستمی را که در کامپیوترشان در حال اجرا بود درک کردند و یکی از آن‌ها لینوس توروالدز (Linus Torvalds) نام داشت.

در سال ۱۹۹۱ الینوس بندیک توروالدز دانشجوی سال دوم علوم کامپیوتر دانشگاه هلسینکی فنلاند و یک هکر خودآموخته بود. این فنلاندی ۲۱ ساله عاشق وصله‌پینه کردن محدودیت‌هایی بود که سیستم را تحت فشار قرار می‌دادند. ولی مهم‌ترین چیزی که وجود نداشت یک سیستم عامل بود که بتواند نیازهای حرفه‌ای‌ها را برآورده نماید.

**Minix** خوب بود ولی فقط یک سیستم عامل مخصوص دانش آموزان بود و بیشتر به عنوان یک ابزار آموزشی بود تا ابزاری قدرتمند برای به کارگیری در امور جدی در این زمان برنامه نویسان سرتاسر دنیا توسط پروژه گنو (GNU) که توسط ریچارد استالمن آغاز شده بود تحریک شده بودند. هدف این پروژه ایجاد حرکتی برای فراهم نمودن نرم افزارهای رایگان و درعین حال باکیفیت بود.

استالمن خط‌مشی خود را از آزمایشگاه معروف هوش مصنوعی دانشگاه MIT با ایجاد برنامه ویرایشگر **emacs** در اواسط و اواخر دهه ۷۰ آغاز نمود. تا اوایل دهه ۸۰ بیشتر برنامه نویسان نخبه آزمایشگاه‌های هوش مصنوعی MIT جذب شرکت‌های نرم افزارهای تجاری شده بودند و با آن‌ها قراردادهای حفظ اسرار امضا شده بود. ولی استالمن دیدگاه متفاوتی داشت. ولی عقیده داشت برخلاف سایر تولیدات نرم افزار باید از محدودیت‌های کپی و ایجاد تغییرات در آن آزاد باشد تا بتواند روز به روز نرم افزارهای بهتر و کارآمدتری تولید نمود.

با اعلامیه معروف خود در سال ۱۹۸۳ پروژه **GNU** را آغاز کرد. وی حرکتی را آغاز کرد تا با فلسفه خودش به تولید و ارائه نرم افزار بپردازد نام **gnu is not unix** مخفف **gnu is not unix** است. ولی برای رسیدن به رؤیای خود برای ایجاد یک سیستم عامل رایگان وی ابتدا نیاز داشت تا ابزارهای لازم برای این کار را ایجاد نماید. بنابراین در سال ۱۹۸۴ وی شروع به نوشتن و ایجاد کامپایلر زبان **C** گنو موسوم به **gcc** نمود.

ابزاری مبهوت کننده برای برنامه نویسان مستقل. با جادوئی افسانه‌ای خود به تنهایی ابزاری را ایجاد نمود که برتر از تمام ابزارهایی که تمام گروه‌های برنامه نویسان تجاری ایجاد کرده بودند قرار گرفت.

**GCC** یکی از کارآمدترین و قوی‌ترین کامپایلرهای است که تاکنون ایجاد شده‌اند. تا سال ۱۹۹۱ پروژه **GNU** تعداد زیادی ایجاد کرده بود ولی هنوز سیستم عامل رایگانی وجود نداشت. حتی **MINIX** هم لایسنس شده بود.

کار بر روی هسته سیستم عامل گنو موسوم به **HURD** ادامه داشت ولی به نظر نمی‌رسید که تا چند سال آینده قابل استفاده باشد.



این زمان برای توروالدوز بیش از حد طولانی بود.. در ۲۵ آگوست ۱۹۹۱ این نامه تاریخی به گروه خبری MINIX از طرف توروالدز ارسال شد.

از: لینوکس بندیکت توروالدز

به: گروه خبری MINMX

موضوع: بیشتر چه چیزی را می خواهید در MINIX ببینید؟

خلاصه نظرخواهی کوچک در مورد سیستم عامل جدید من، با سلام به تمام استفاده کنندگان از MINIX من در حال تهیه یک سیستم عامل رایگان فقط به عنوان سرگرمی و نه به بزرگی و حرفه ای GNU برای دستگاه های ۳۸۶ و ۴۸۶ هستم. این کار از آوریل شروع شده و در حال آماده شدن است من مایلم تا نظرات کاربران را در مورد چیزهایی که در MINIX دوست دارند یا ندارند جمع آوری کنم. زیرا سیستم عامل من حدوداً شبیه آن است. مانند ساختار سیستم فایل مشابه و چیزهای دیگر من اکنون bash نسخه ۱،۰۸ و GCC نسخه ۱،۴۰ را به آن منتقل کرده ام و به نظر می رسد که کار می کند. من در عرض چند ماه آزمایشی درست کرده ام و مایلم بدانم که کاربران بیشتر به چه قابلیت هایی نیاز دارند؟ من از هر پیشنهادی استقبال می کنم ولی قول نمی دهم همه آنها را اجرا کنم لینوکس همان طور که در این نامه پیداست خود توروالدز هم باور نمی کرد که مخلوقش آن قدر بزرگ شود که چنین تحویلی در دنیا ایجاد کند. لینوکس نسخه ۰،۰۱ در اواسط سپتامبر ۱۹۹۱ منتشر شد و روی اینترنت قرار گرفت. شور و اشتیاقی فراوان حول مخلوق توروالدز شکل گرفت کدها دانلود شده آزمایش شدند و پس از بهینه سازی به توروالدز بازگردانده شدند لینوکس نسخه ۰،۰۲ در پنجم اکتبر به همراه اعلامیه معروف بوروالدز آمده شد لینوکس نسخه ۰،۰۳ پس از چند هفته آماده شد و تا دسامبر لینوکس به نسخه ۰،۱۰ رسید. هنوز لینوکس فقط چیز کمی بیشتر از یک فرم اسکلت بود این سیستم عامل فقط دیسک های سخت AT را پشتیبانی می کرد و ورود به سیستم نداشت و مستقیماً به خط فرمان بوت می شد.

نسخه ۰،۱۱ خیلی بهتر شد. این نسخه از صفحه کلیدهای چندزبانه پشتیبانی می کرد. دیسک های فلاپی و کارت های گرافیکی VGA و EGA و ... نیز پشتیبانی می شدند. شماره نسخه ها از ۰،۱۲ به ۰،۹۵ و ۰،۹۶ افزایش پیدا کرده و ادامه یافت. به زودی که آن به وسیله سرویس دهنده های FTP در فنلاند و مناطق دیگر در سرتاسر جهان منتشر شد. به زودی صدها نفر به اردوگاه لینوکس پیوستند. سپس هزاران نفر و سپس صدها هزار نفر لینوکس دیگر اسباب بازی هکرها نبودند با پشتیبانی نرم افزارهای پروژه GNU لینوکس آماده یک نمایش واقعی بود. لینوکس تحت مجوز GPL قرار داده شد. با این مجوز همه می توانستند کدهای منبع لینوکس را به رایگان داشته باشند بر روی آنها مطالعه کرده و آنها را تغییر دهند. دانشجویان و برنامه نویسان آن را

قاپیدند و خیلی زود تولیدکنندگان تجاری وارد شدند. لینوکس به خودی خود رایگان بود و هست. کاری که این تولیدکنندگان انجام دادن کامپایل کردن بخش‌ها و نرم‌افزارهای مختلف و ارائه آن به صورت یک فرمت قابل توزیع همانند سایر سیستم‌عامل‌ها بود تا مردم عادی نیز بتوانند از آن استفاده کنند اکنون توزیع‌هایی مانند ردهت و دیبان و سوزی دارای بیشترین سهم کاربران در سرتاسر جهان هستند با رابط‌های گرافیکی کاربر جدید مانند KDE و GNOME توزیع‌های لینوکس در بین مردم بسیار گسترش یافتند.

بهترین موردی که امروزه برای لینوکس وجود دارد طرفداران متعصب آن هستند هنگامی که یک قطعه سخت‌افزاری جدید ارائه می‌شود هسته لینوکس برای استفاده از آن تغییر داده می‌شود. برای مثال هنگام ارائه پردازنده ۶۴بیتی شرکت ADM هسته به سرعت چند هفته برای کار با آن آماده شد. اکنون لینوکس بر روی تمام انواع خانواده‌های سخت‌افزاری موجود اعم از PC و MAC و ALPHA و انواع سخت‌افزارهای درونی قابل اجراست که آن را برای استفاده در ماشین‌آلات صنعتی و آلات و ادواتی که نیاز به پردازش کامپیوتری دارند بسیار مناسب نموده است. لینوکس با همان فلسفه و هدفی که در سال ۱۹۹۱ ایجاد شد وارد هزاره جدید شده است.

توروالدز هنوز یک انسان ساده است برخلاف بیل گیتز او یک میلیاردی نیست پس از اتمام مطالعاتش وی به آمریکا رفت تا با شرکت Transmeta همکاری کند. پس از انجام یک پروژه فوق سری که توروالدز یکی از اعضای فعال آن بود، پردازنده Cruose را به بازار ارائه کرد.

توروالدز هنوز پرفرودارترین و مشهورترین برنامه‌نویس جهان است. در حال حاضر توروالدز ترانسمتا را ترک نموده و با حمایت شرکت‌های بزرگ به طور تمام وقت بر روی لینوکس کار می‌کند.

### پس از یک دهه:

امروز لینوکس بیش از یک دهه توسعه را پشت سر گذاشته است و یکی از سریع‌ترین توسعه‌ترین سیستم‌های عامل به شمار می‌رود.

از چند کاربر انگشت‌شمار در سال‌های ۱۹۹۱ و ۱۹۹۲ امروزه میلیون‌ها کاربر از لینوکس استفاده می‌کنند

IBM که زمانی بزرگ‌ترین دشمن جماعت Open Source به شمار می‌رفت اکنون سرمایه‌گذاری عظیمی در زمینه توسعه راه‌حل‌های Open Source تحت لینوکس نموده است.

در حال حاضر تعداد توسعه‌دهندگانی که برای افزایش قابلیت‌های لینوکس تلاش می‌کنند روزبه‌روز افزایش می‌یابد.

امروزه تعداد زیادی از شرکت‌ها و مؤسسات حرفه‌ای تجاری پشتیبانی از محصولات مبتنی بر لینوکس را بر عهده گرفته‌اند.

اکنون دیگر استفاده از لینوکس در محیط‌های اداری پذیرفتن ریسک نیست. از نظر قابلیت اطمینان و پایداری و همچنین حفاظت در برابر انواع ویروس‌ها چیزی بهتر از لینوکس وجود ندارد. با تلاش شرکت‌های بزرگی مانند ردهت استفاده از لینوکس در محیط‌های تجاری توسعه فراوان یافته و اکنون تعداد زیادی از شرکت‌های کوچک و بزرگ در حال استفاده از سرویس‌دهنده‌ها و ایستگاه‌های کاری مبتنی بر لینوکس هستند.

## طلوع لینوکس روی میزی (Desktop Linux)

بزرگ‌ترین ایرادی که از لینوکس گرفته می‌شد چه بود؟

قبلاً محیط تمام متنی لینوکس بسیاری از کاربران را از استفاده کردن از آن بر حذر می‌داشت باینکه در استفاده از محیط متنی کنترل کامل سیستم در اختیار شماست. ولی این محیط اصلاً برای کاربران عادی سیستم‌های کامپیوتری مناسب نیست. محیط‌های گرافیکی که بر پایه X-Window وجود داشتند نیز پاسخ‌گوی امکاناتی که سیستم‌عامل‌های گرافیکی مانند ویندوز برای کاربران خود ارائه می‌کردند نبود. ولی از چند سال گذشته این وضعیت در حال تغییر بوده است. اکنون محیط‌های گرافیکی حرفه‌ای مانند KDE (k desktop environment) و GNOME (Network object model environment) تصویر لینوکس را کامل کرده‌اند. این محیط‌های گرافیکی اکنون بسیار کاربرپسند و قدرتمند شده‌اند و وجود این سیستم‌هاست که امروزه کاربران عادی نیز می‌توانند از لینوکس استفاده کنند.

## لینوکس در جهان سوم

ورود لینوکس به کشورهای جهان سوم تحولی ایجاد نموده است. قبل از وجود لینوکس کشورهای جهان‌سومی در زمینه کامپیوتر در سطح بسیار پایین‌تری قرار داشتند. هزینه سخت‌افزار بسیار پایین آمده بود ولی هزینه نرم‌افزار برای این‌گونه کشورها همچنان کمرشکن بود. این امر باعث شد تا در بسیاری از این کشورها کپی غیرمجاز نرم‌افزارهای گسترش پیدا کند که باعث میلیاردها دلار خسارت سالیانه می‌شود. یکی از عمده‌ترین دلایل این کار پایین بودن درآمد سرانه در این کشورهاست. هنگامی که مجموع درآمد سرانه سالیانه بیش از ۲۰۰ تا ۳۰۰ دلار نیست هیچ‌گاه امکان خرید یک سیستم‌عامل ۱۰۰ دلاری وجود نخواهد داشت. طلوع لینوکس و سایر تولیدات باز متن این وضعیت را تغییر داده است. این امکان وجود دارد تا بتوان لینوکس را در کامپیوترهای قدیمی ۴۸۶ و پنتیوم که اکنون در کشورهای توسعه‌یافته به تاریخ پیوسته‌اند ولی هنوز در کشورهای در حال توسعه از آن‌ها استفاده می‌شود اجرا نمود. همچنین استفاده از نرم‌افزارهای رایگان باز متن گسترش یافته تا جلوی هزینه‌های سرسام‌آور نرم‌افزاری این کشورها را بگیرد. امروزه در کشورهای آسیایی، آفریقایی و آمریکای لاتین استفاده از لینوکس و نرم‌افزارهای باز

متن گسترش فراوانی یافته با استفاده از خصلت ذاتی تغییرپذیری لینوکس برای استفاده از زبان‌های ملی این کشورها سفارشی شده است. امروزه مستندات لینوکس به اکثر زبان‌های زنده جهان ترجمه شده‌اند.

## از میز کار تا ابر کامپیوترها

هنگامی که توروالدز لینوکس را ایجاد نمود این مخلوق جدید فقط یک اسباب‌بازی تازه برای هکرها بود. ولی از زمان دستگاه‌های ۳۸۶ نخستین هسته لینوکس بر روی آن‌ها اجرا می‌شد. لینوکس راه درازی را طی نموده است. یکی از مهم‌ترین استفاده‌های امروزی لینوکس استفاده از آن در پردازش‌های سنگین موازی در ابر کامپیوترهاست. امروزه اکثر ابر کامپیوترهایی که در جهان ساخته می‌شوند از لینوکس به عنوان سیستم عمل خود استفاده می‌کنند.

## داستان ادامه دارد...

حرکت لینوکس از یک پروژه هکری تا جهانی شدن یک انقلاب شگفت‌انگیز است. پروژه GNU که در اوایل دهه ۱۹۸۰ توسط ریچارد استالمن شروع شد. توسعه نرم‌افزارهای باز متن را رهبری نمود پروفیسور اندرو تانباوم سیستم عامل MINIX او مطالعه سیستم عامل‌ها را از حالت تئوری به عملی تبدیل نمود و در نهایت همت و تلاش توروالدز منجر به تولد لینوکس شد. امروزه لینوکس دیگر یک پروژه هکری به شما نمی‌رود بلکه یک حرکت جهانی است که توسط میلیون‌ها نفر برنامه‌نویس باز متن و شرکت‌های بزرگی مانند IBM حمایت می‌شود. لینوکس در تاریخ کامپیوتر به عنوان یکی از شگفت‌انگیزترین محصولات تلاش بشری باقی خواهد ماند.

## نشان لینوکس

نشان لینوکس یک پنگوئن است. برخلاف سایر سیستم‌عامل‌های تجاری، این نشان زیاد جدی نیست! نشانگر وضعیت بدون نگرانی حرکت لینوکس است. این نشان تاریخچه بسیار جالبی دارد. لینوکس در ابتدا فاقد هرگونه نشانی بود، هنگامی که توروالدز برای تعطیلات به استرالیا رفته بود. در دیداری که از یک باغ وحش داشت، هنگامی که می‌خواست با یک پنگوئن بازی کند، پنگوئن دست وی را گاز گرفت و همین ایده‌ای شد تا از پنگوئن به عنوان نشان لینوکس استفاده شود.

## Free Ware یا برنامه آزاد یعنی چه؟

نرم‌افزاری را آزاد می‌گویند که دارای شرایط زیر باشد:

۱- کاربر این نوع برنامه می‌تواند برنامه را برای هر هدفی اجرا کند.

۲- کاربر این نوع برنامه می‌تواند برنامه را بنا به نیازهای خود تغییر دهد برای این کار کاربر باید کد برنامه را در اختیار داشته باشد.

۳- کاربر این نوع برنامه می‌تواند برنامه را تکثیر کند و آن رایگان و یا در مقابل پول در اختیار دیگران قرار دهد اما در صورت باید کد برنامه نیز منتشر شود.

۴- کاربر آزاد است که نسخه‌های تصحیح‌شده برنامه را منتشر کند.

برای تضمین کردن شرایطی که یک **Free Ware** باید داشته باشد به نام **Copy Left** تهیه شده است که دقیقاً در مقابل **Copr Right** قرار دارد و اجازه‌ی تملیکی شدن نرم‌افزارها را نمی‌دهند.

در سال ۱۹۹۸، عده‌ای که عقیده داشتند واژه نرم‌افزار آزاد مناسب نمی‌باشد، شروع به استفاده از واژه باز متن ( **Open Source**) کردند. تفاوت این گروه با طرفداران پروژه گنو و نرم‌افزار آزاد در این است که به عقیده این گروه نرم‌افزار انحصاری اشکالی ندارد، اما نرم‌افزار متن‌باز صرفاً بهتر است، حال آنکه به عقیده طرفداران نرم‌افزار آزاد، نرم‌افزار انحصاری غلط است و باعث ماندن جامعه می‌گردد. گروهی نیز به‌تازگی برای آنکه خود را به هیچ‌یک از این کمپ‌ها نسبت ندهند، از عبارت **Free/Libre Open Source Software** یا **Floss** استفاده می‌کنند که مجموعه‌ای از همه واژه‌های توصیف‌کننده نرم‌افزار آزاد می‌باشد.

### کدام توزیع گنو/ لینوکس را انتخاب کنیم؟

یکی از سؤالاتی که توسط کاربرانی که مایل به انتقال به سیستم عامل گنو/ لینوکس هستند مطرح می‌شود، انتخاب توزیع است. چرا انواع مختلفی از گنو/لینوکس وجود دارد؟ کدام یک مناسب‌تر است؟ در ادامه به معرفی اصول و مفاهیم توزیع‌های گنو/لینوکس و معرفی برخی از آن‌ها می‌پردازیم و به مقایسه توزیع‌های مختلف خواهیم پرداخت.



### توزیع چیست ؟

سیستم عامل گنو/لینوکس از بخش‌های بسیار زیادی تشکیل شده که هر بخش آن توسط عده‌ای خاص توسعه می‌یابد که هر یک در یک سمت جهان قرار دارند. معروف است که می‌گویند گنو/لینوکس مانند هواپیمایی است که هر قسمت آن را در یک کشور ساخته‌اند. البته این نکته نقطه قوت آن به شمار می‌رود. در صورتی که شما به‌عنوان یک کاربر بخواهید یک گنو/لینوکس داشته باشید، باید تمام این قطعات را جداگانه‌ای جمع‌آوری کرده و پس از کامپایل استفاده نمایید. درصد کمی از مردم این

امکان و توانایی را دارند. بنابراین افراد و شرکت‌هایی اقدام به جمع‌آوری این قطعات مجزا و قرار دادن آن‌ها کنار هم کرده‌اند و علاوه بر آن برای این مجموعه برنامه‌های نصب و مدیریت نوشته‌اند تا کار نصب و مدیریت سیستم را برای کاربران آسان کنند. به این مجموعه‌ها که توسط افراد و شرکت‌ها گردآوری شده است، توزیع یا **Distribution** گنو/لینوکس می‌گویند.

## علت تنوع توزیع‌ها چیست؟

هر یک از توزیع‌های گنو/لینوکس دارای ویژگی‌های خاصی است که آن را از توزیع دیگر متمایز می‌کند. مثلاً ممکن است برنامه‌های نصب آن‌ها باهم تفاوت داشته باشند (البته اصول نصب همه گنو/لینوکس‌ها یکسان است) و یا ابزارهای مدیریتی گرافیکی تهیه‌شده باهم متفاوت باشند و یا نسخه برنامه‌هایی که با یک توزیع خاص ارائه می‌شوند جدیدتر یا قدیمی‌تر باشند، محل فایل‌های پیکربندی آن‌ها متفاوت باشد، و یا ممکن است توزیع‌هایی مخصوص امور خاص طراحی شده باشند. مثلاً مخصوص سرویس‌دهنده‌ها. مخصوص ایستگاه‌های کاری. مخصوص کامپیوترهای قدیمی. مخصوص مدیریت شبکه. مخصوص چندرسانه‌ای. مخصوص بازی و به همین ترتیب توزیع‌ها به دو صورت تجاری و رایگان ارائه می‌شوند.

## انتخاب توزیع:

همان‌طوری که در بالا اشاره شد هر یک از توزیع‌ها دارای ویژگی‌های خاص خود هستند. مثلاً برنامه نصب یک توزیع بسیار راحت است و یک توزیع دیگر ممکن است از نظر پایداری و امنیت مطرح باشد انتخاب توزیع بستگی به شرایط زیر دارد

۱- سطح علمی کاربر

۲- مورد استفاده از گنو/لینوکس

۳- ویژگی‌های توزیع

۴- بازار

الف: سطح علمی کاربر: کاربرانی که دارای آشنایی کمتری با گنو/لینوکس هستند. جذب توزیع‌هایی می‌شوند که دارای ابزارهای پیکربندی گرافیکی است که آن‌ها را قادر می‌سازد راحت‌تر سیستمشان را اداره و نصب نمایند. همچنین دارای نرم‌افزارهای جدیدتری باشد که به آن‌ها حداکثر قابلیت‌ها را ارائه نماید. از توزیع‌هایی که برای کاربران تازه‌کار مناسب‌تر هستند

می‌توان اوبونتو **Ubuntu** و سوزی **SuSE** و فدورا **Fedora** و مندریوا **Mandriva** و لیندوز **Lindows** و لیکوریس **Lycoris** و مپیس **Mepis** و **XandarOS** را نام برد. کاربرانی که پیشرفته‌تر هستند و ابزارهای پیکربندی گرافیکی برایشان مهم نبوده. کیفیت و سرعت سیستم برایشان مهم‌تر است جذب توزیع‌های حرفه‌ای‌تر مانند دیبان **Debian** جنتو **Gentoo** و اسلکور **Slackware** می‌شوند. ممکن است نصب و راه‌اندازی این توزیع‌ها برای کاربران تازه‌کار دشوار باشد. ولی در عوض هر سه آن‌ها بسیار باکیفیت و پایدار هستند.

ب: مورد استفاده از گنو/لینوکس: برخی از توزیع‌های مخصوص نیازهای خاصی طراحی شده‌اند. مثلاً امروزه از اصلی‌ترین نیازها می‌توان به سرویس‌دهنده‌ها و ایستگاه‌های کاری اشاره نمود. البته برخی از توزیع‌ها این امکان را به شما می‌دهند که هنگام نصب نوع مصرف آن‌ها را تعیین کنید و با توجه به انتخاب شما نرم‌افزارهای مربوط به آن مصرف خاص نصب خواهند شد. مثلاً اوبونتو

و فدورا و دبیان این امکان را دارا هستند. برخی از توزیع‌ها تنها مخصوص یک نیاز طراحی شده‌اند و دارای ابزارهای مربوط به آن نیاز می‌باشند مثلاً گنو/لینوکس ناپیکس که یک توزیع روی میزی است تنها دارای ابزارهایی است که برای کاربران روی میزی کاربرد دارد.

ج: ویژگی‌های توزیع: برخی اوقات یک توزیع دارای ویژگی‌های خاصی است که آن را برای استفاده قابل انتخاب می‌سازد. مثلاً گنو/لینوکس اورالوکس (Oralux) دارای امکانات مخصوص نابینایان می‌باشد. مانند شناسایی صفحه‌نمایش‌های بریل و یا مرور صوتی وب و پست الکترونیک و یا یک گنو/لینوکس ممکن است سخت‌افزارهای خاصی را به خوبی پشتیبانی نمی‌کند و یا ممکن است سرعت و کیفیت یک توزیع یا آسانی استفاده از آن ملاک انتخاب قرار گیرد.

د: بازار: ممکن است موجود بودن یک توزیع در بازار و یا نبود آن ملاک انتخاب باشد. مثلاً در ایران فراوان‌ترین توزیع گنو/لینوکس توزیع فدورا و ردهت است. اصلاً برخی افراد و مؤسسات گنو/لینوکس را به نام ردهت می‌شناسند. به دلیل اینکه Linux بر گفته و تجربه یک ربع فراوان کار روی UNIX است. پس بهتر است ابتدا با ویژگی‌ها و ساختار یونیکس آشنا شویم.

### ویژگی‌های سیستم عامل یونیکس (UNIX)

۱- Multi programming: چند برنامه در آن واحد روی حافظه اصلی resident شده‌اند و پردازنده‌ها بین آن‌ها سوئیچ می‌کنند که باعث افزایش بازدهی سیستم می‌شود.

۲- Time Sharing: در این روش پردازنده مرکزی جسمی است که توسط تمام کاربران و پردازنده‌ها استفاده می‌شود و به هر یک کسری از زمان CPU تعلق می‌گیرد.

۳- Multi User: چند کاربره بودن.

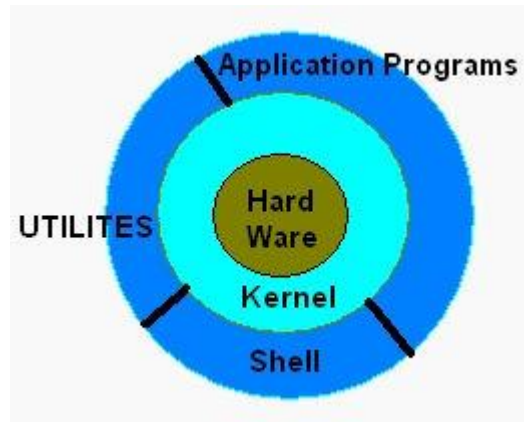
۴- Multi Tasking: امکان اجرای برنامه به صورت Background, Foreground.

۵- File System: دارا بودن File System به صورت سلسله‌مراتب و تأمین امنیت برای داده‌ای سیستم.

۶- generality: بدین معنی که یک روش واحد بتواند اهداف چندگانه‌ای را برآورد سازد.

۷- transportability: سیستم عامل UNIX به آسانی قابل حمل است. برای یک سیستم جدید با دست‌کاری اندک بر روی کدهای سیستم عامل UNIX می‌توان آن را جهت نصب آماده نمود. توانایی حمل سیستم عامل UNIX از یک نوع کامپیوتر به نوع دیگر دلیل اصل موفقیت آن می‌باشد.

## معماری سیستم عامل لینوکس (ARCHITECTURE OF LINUX SYSTEM)



### KERNEL

هسته اصلی سیستم عامل لینوکس kernel می باشد. kernel قسمتی از سیستم که حافظه فایل و وسایل جانبی را مدیریت می کند و زمان و تاریخ و را حفظ می کند. برنامه های کاربردی را آغاز می کند و منابع سیستمی را تخصیص می دهد kernel مستقیماً با سخت افزاری در ارتباط است.

### SHELL

Shell نقش رابط بین کاربر و kernel را بر عهده دارد. shell یک برنامه مفید می باشد که دستورات و فرمانها را از کاربر دریافت می کند و بعد از ترجمه آن را جهت اجرا به kernel تحویل می دهد.

### UTILITES LINUX

لینوکس utilites یا فرمانها یک مجموعه در حدود ۲۰۰ برنامه که فرآیند لازم را بر عهده دارند. این برنامهها از طریق shell درخواست می شوند

### APPLICATION SOFTWARE

برنامه هایی نظیر نرم افزار حسابداری و سیستم های مدیریتی و data base می باشند

### ساختار سیستم عامل یونیکس (UNIX)

سیستم عامل UNIX شامل چهار بخش اصلی زیر است

۱- Kernel

۲- File System

۳- Shell

۴- Commands



## ساختار سیستم فایل (File System) در یونیکس (UNIX)

فایل سیستم ساختمانی برای ذخیره و بازیابی اطلاعات ارائه می‌دهد. هر فایل سیستم دیسک را به چهار منطقه تقسیم می‌کند که عبارت‌اند از:

۱- **boot block**: اولین بلاک حافظه جانبی که برای سیستم رزرو می‌شود و حاوی اطلاعات لازم جهت راه‌اندازی سیستم می‌باشد.

۲- **super block**: بلاک شماره ۱ را سوپر بلاک می‌نامند. این بلاک شامل مجموعه اطلاعات است که وضعیت فایل سیستم را مشخص می‌کند که عبارت است از:

سایز فایل سیستم

تعداد **Inode** های موجود

آدرس اولین بلاک حاوی اطلاعات

تعداد بلاک‌های آزاد

تعداد **Inode** های آزاد

زمان آخرین بروز رسانی فایل سیستم

اینکه آیا یک فایل سیستم در ست **Close** شده است یا نه

ویرایش و نوع فایل سیستم

سایز هر بلاک

۳- **list**: یک ناحیه شامل یک لیست پیوندی از **inode** ها می‌باشد. **inode** ساختاری به طول ۶۴ بایت می‌باشد و شماره آن از ۱ آغاز می‌شود و حاوی اطلاعات زیر می‌باشد:

شمار کاربر فایل **UID**

شماره گروه فایل **GID**

آدرس فیزیکی محتوای فایل روی دیسک

سایز فایل

زمان ایجاد فایل

نوع فایل

زمان آخرین دستیابی به فایل

زمان آخرین تغییر روی فایل

مجوزهای فایل

تعداد لینک‌های فایل

نکته: **inode** شماره ۱ برای سیستم رزرو شده است و **inode** شماره ۲ مربوط به فهرست ریشه می‌باشد.

۴-Date: فضای آزاد باقی مانده روی دیسک به صورت یک لیست پیوندی از بلاک‌های در دسترس دیسک نگهداری می‌شود که برای ذخیره داده‌ها مورد استفاده قرار می‌گیرد.

نکته: سیستم عامل UNIX حداقل یک فایل سیستم روی دیسک سخت اولیه‌اش دارد. این فایل سیستم **root** نام دارد و با علامت / مشخص می‌شود. فایل سیستم **root** شامل برنامه‌های و دایرکتوری‌هایی است که توسط سیستم عامل ایجاد می‌شود. نکته: نگهداری اطلاعات در فایل سیستم از وظایف سیستم عامل است از دست دادن اطلاعات اتفاق نادری است زیرا فایل سیستم در برابر تخریب اطلاعات بسیار مقاوم است سیستم عامل UNIX از برنامه **FSCK** برای تعمیر فایل سیستم خراب شده استفاده می‌کند.

**FSCK** در هنگام بوت به طور خودکار فایل سیستم **root** را چک می‌کند هنگامی که سیستم به طور غیرمعمول **terminate** شده باشد **fsck** برای تمیز کردن فایل سیستم از مدیر سؤال می‌کند و در صورت تأیید او فایل سیستم تمیز می‌گردد.

### مفهوم سیستم فایل یونیکس (UNIX) و لینوکس (Linux)

اگر بخواهیم یک توصیف ساده در مورد سیستم‌های یونیکس و مشابه آن یعنی لینوکس ارائه کنیم باید بگویید در یک سیستم مبتنی بر یونیکس هر چیزی یک فایل است یا یک فایل نماینده خود را دارد و اگر چیزی یک فایل نباشد حتماً یک روند یا **process** است.

این توصیف به راستی حقیقت دارد چراکه در یونیکس فایل‌های مخصوص وجود دارند که وظایفی بیشتر از یک فایل بر عهده دارند (برای مثال **pipe** ها و **socket** ها).

در یک سیستم گنو/لینوکس درست مثل یونیکس هیچ تفاوتی بین فایل و دایرکتوری وجود ندارد به این معنی که یک دایرکتوری هم خود یک فایل است که حاوی اسامی تعدادی فایل یا دایرکتوری دیگر است.

برنامه‌ها، سرویس‌ها، متن‌ها و تصاویر و سایر موارد مشابه و همچنین دستگاه‌ها ورودی و خروجی و عموماً همه ابزارهای سیستمی همگی با یک فایل به سیستم معرفی می‌شوند.

حال با این تفکر اگر بخواهیم همگی این فایل‌های متنوع را در یک ساختار منطقی مرتب کرده و نگهداری کنیم باید به آن‌ها به صورت یک ساختار درختی نگاه کنیم که روی هارد دیسک استقرار یافته است (مثل سیستم عامل **DOS**).

این ساختار درختی از یک ریشه اصلی یا **ROOT** تشکیل شده و شاخه‌های بزرگ‌تر منشعب شده از آن دارای زیرشاخه و شاخه‌های انتهائی دارای برگ‌هایی هستند که همان فایل‌های ما می‌باشند.

هارد دیسک به پارتیشن تقسیم می‌شود و دیوارک‌ها برای ذخیره اطلاعات توسط کاربران با سیستم فایل‌ها فرمت می‌شوند.

سیستم فایل پیش فرض برای لینوکس **THIRD EXTENDED LINUX ILE SYSTEMEXT3** می‌باشد.

دیگر سیستم فایل‌ها **msdos** , **ext2** که برای فلاپی‌ها مورد استفاده فرار می‌گیرند هستند همچنین **iso 9660** برای **cd** مورد استفاده قرار می‌گیرد.

## انواع فایل‌ها در لینوکس

۱- فایل‌های معمولی **regular files**: بیشتر فایل‌ها فقط معمولی هستند که حاوی اطلاعات معمولی مثل متن و کدهای اجرایی و برنامه‌ها و خروجی و ورودی برنامه‌های کاربردی دیگر می‌باشد.

۲- فایل‌های اختصاصی

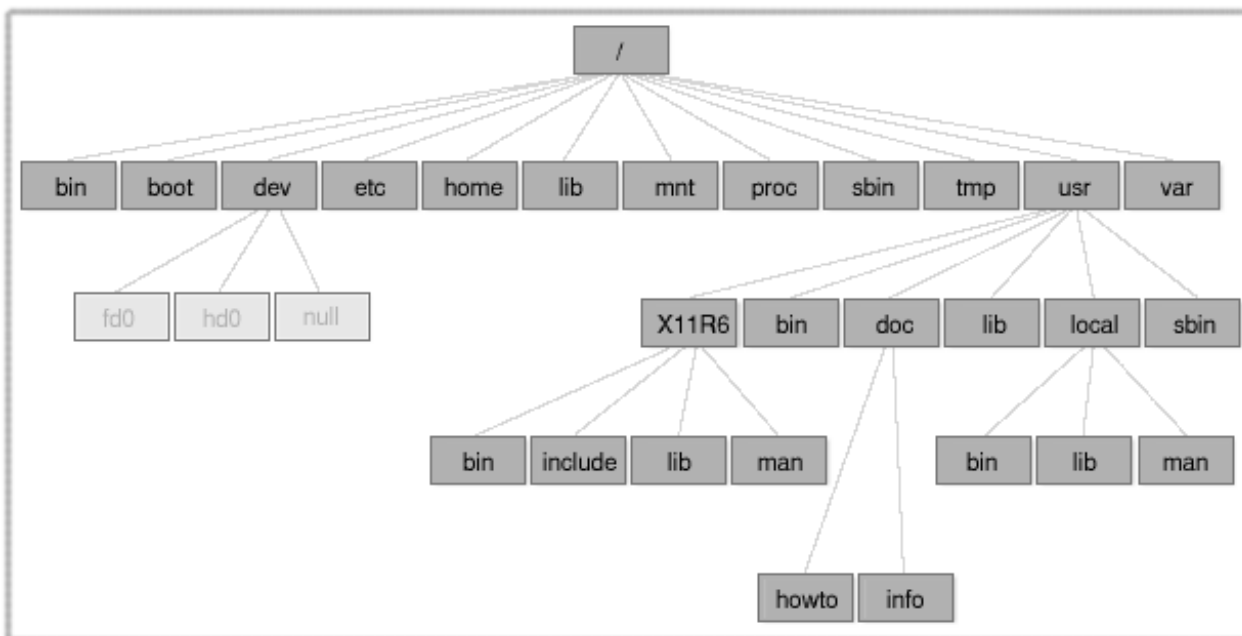
دایرکتوری: فایل‌های حاوی اسامی دیگر فایل‌ها

فایل‌های ویژه: حاوی مکانیزم‌های چگونگی ورود و خروج اطلاعات به کامپیوتر (همه فایل‌های موجود در مسیر **/dev** از این دسته هستند).

لینک‌ها سیستمی را فراهم می‌کنند که یک فایل یا دایرکتوری در نقاط مختلف ساختار درختی بدون نیاز به تکرار محتوای قابل مشاهده و در دسترس باشند.

**DOMAIN** یا **Sockets**: نوع بخصوصی از فایل‌ها که همانند سوکت‌های **tcp/ip** روندهائی را که برای کنترل دسترسی به فایل سیستم در شبکه به صورت کنترل شده فراهم می‌کنند.

**Named piped**: کم‌وبیش عملی همانند سوکت‌ها انجام می‌دهند ولی متفاوت با روالی که سوکت‌ها در پیش می‌گیرند سیستم فایل لینوکس ساختاری است که اطلاعات شما را در کامپیوتر ذخیره می‌کند. فایل‌های در یک ساختار درختی از دایرکتوری‌ها ذخیره می‌شوند. هر دایرکتوری می‌تواند حاوی فایل‌ها و دایرکتوری‌های دیگری باشد در صورتی که بخواهید ساختار سیستم فایل لینوکس را دقیق‌تر توصیف کنید آن بیشتر شبیه یک درخت وارونه است آن بالاترین نقطه دایرکتوری ریشه قرار دارد که به وسیله یک اسلش تنها نشان داده می‌شود. در زیر آن دایرکتوری‌های عمومی و سیستمی سیستم عامل لینوکس قرار می‌گیرند مانند **bin, dev, home, tmp** هرکدام از این دایرکتوری‌ها محتوی دایرکتوری‌های دیگری هستند. تصویر زیر ساختار درختی سیستم فایل لینوکس را نشان می‌دهد.



برخی از دایرکتوری‌های مهم سیستم فایل لینوکس در زیر توضیح داده شده‌اند.

## معرفی دایرکتوری‌های سطح بالای سیستم فایل‌های رایج در لینوکس:

**/:** این دایرکتوری **root** یا ریشه مبنای سیستم فایل لینوکس است. از نظر منطقی کلیه دایرکتوری‌ها و فایل‌های لینوکس صرف‌نظر از محل فیزیکی آنان در این دایرکتوری قرار دارند.

**/bin:** این دایرکتوری شامل برنامه‌های اجرایی است که جزئی از سیستم‌عامل لینوکس هستند. بسیار از فرمان‌های لینوکس مانند **tar, more, ls, cp, cat** در این دایرکتوری قرار دارند.

**/boot:** این دایرکتوری حاوی هسته لینوکس و دیگر فایل‌هایی است که موردنیاز برنامه **GRUB boot Manager** است. (این هسته و فایل‌های دیگر می‌توانند در هر دایرکتوری نیز قرارداد ولی به‌طور مرسوم آن‌ها را در این دایرکتوری قرار می‌دهند)

**/dev:** این دایرکتوری شامل فایل‌های ویژه جهت نمایش وسایل متصل شده به سیستم هستند.

**/etc:** این دایرکتوری حاوی بیشترین پرونده‌های پیکربندی سیستم و اسکریپت‌های **iniliations** است.

**/home:** محل قراردادی دایرکتوری خانگی کلیه کاربران مانند دایرکتوری خانگی **/home/Roohollah** برای کاربری به نام **Roohollah**

**/lib:** دایرکتوری حاوی فایل‌های کتابخانه برای کلیه برنامه‌های ذخیره‌شده در دایرکتوری‌های **/bin** و **/sbin** از جمله ماژول‌های قابل بارگذاری درایو که موردنیاز راه‌اندازی لینوکس است

**/media:** یک دایرکتوری برای سوار کردن سیستم‌های فایل روی رسانه‌های قابل حمل مانند درایوهای **CD-ROM** دیسکت های فلاپی و درایوهای **ZIP** این دایرکتوری شامل دایرکتوری **/media/floppy** برای سوار کردن دیسکت های فلاپی و دایرکتوری **/media/cdrom** برای سوار کردن سی‌دی‌رام است.

اگر یک ضبط کننده **CD** داشته باشید بجای دایرکتوری آن دایرکتوری **/media/cdrecorder** در اختیار شما گذاشته می‌شود.

**/mnt:** یک دایرکتوری برای سیستم‌های فایل سوار شده موقتی.

**/opt:** این دایرکتوری یک ناحیه ذخیره‌سازی برای بسته‌های نرم‌افزار برنامه‌های بزرگ کاربردی ارائه می‌دهد. مانند برنامه‌های کاربردی **KDE** و **Gnome** نصب‌شده در این دایرکتوری.

**/proc:** یک دایرکتوری خاص ساکن در حافظه حاوی اطلاعات مختلف درباره کارهای در حال اجرا در سیستم لینوکس.

**/root:** دایرکتوری خانگی برای کاربر ریشه.

**/sbin:** حاوی فایل‌های اجرایی معرف فرمان‌هایی که به‌طور نوعی برای کارهای سرپرستی استفاده می‌شود و توسط کاربر **Root** بکار می‌رود. فرمان‌هایی مانند **Halt** و **shutdown** در دایرکتوری **/sbin** قرار می‌گیرند.

**/srv:** این دایرکتوری شامل اطلاعاتی برای خدماتی مانند **web** و **FTP** ارائه‌شده در این سیستم است

**/SYS:** یک دایرکتوری خاص حاوی اطلاعات درباره وسایل کامپیوتر که هسته لینوکس با آن‌ها سروکار دارد

**/tmp:** یک دایرکتوری موقت قابل‌استفاده برای هر کاربر لینوکس به‌عنوان یک دایرکتوری **scratch** به این معنا که محتویات آن حائز اهمیت نبوده و هر بار با راه‌اندازی سیستم محتویات آن پاک می‌شود.

/usr: دایرکتوری حاوی زیر دایرکتوری‌ها برای بسیاری از برنامه‌های مهم مانند **x window system** که در دایرکتوری **user/x11R6** و کتابچه راهنمای **ONLINE**

**/var**: این دایرکتوری حاوی فایل‌های مختلف سیستم مانند فایل‌های گزارش‌های سیستمی و نیز دایرکتوری‌هایی برای نگهداری سایر اطلاعات مانند فایل‌هایی برای چاپ و پیام‌های الکتریکی است.

### **LILO و GRUB و فرآیند راه‌اندازی لینوکس**

در لینوکس دو مدیر رایج وجود دارد. لیلو (**lilo=linux loader**) مدیر بوت سنتی لینوکس و گراب (**grub=grand unified unix bootloader**) مدیر بوتی جدیدتر می‌باشد. هر کدام از این برنامه‌ها ابتدا مقداری اطلاعات پیکربندی را دریافت کرده و سپس هسته لینوکس یا سیستم‌عامل دیگری را بارگزاری می‌کند و ادامه فرایند بوت را به آن می‌سپارد.

بعد از اینکه بایاس کنترل را به بوت لودر سپرد. بوت لودر هم کنترل را در نهایت به سیستم‌عامل شما مثلاً به لینوکس می‌سپارد. بدیهی است که بوت لودرهای لینوکس به راحتی قابل پیکربندی و اختصاصی شدن هستند و اطلاعات نخستین سیستم‌عامل قابل بوت و زمان وقفه و پارتیشن هارد دیسک محل استقرار

**MBR (Master Boot Record)** یا سیستم‌عامل‌ها با ویرایش فایل‌های **/etc/lili.conf** یا **/boot/grub/menu.list** قابل تغییر است.

اولین چیزی که لینوکس پس از آغاز عملیات بوت انجام می‌دهد تغییر وضعیت سیستم یا سوئیچ کردن به وضعیت **protected mode** یا همان سیستم امن می‌باشد.

البته باید دانست که سیستم‌های عامل قدیمی همچون داس بدون تغییر وضعیت سیستم از لحاظ امنیتی به وضعیت بدون حفاظ **real mode** می‌شوند.

همان‌طوری که می‌بینید سیستم‌های عامل پیشرفته مثل لینوکس با انجام این کار کنترل سیستم را کاملاً به دست گرفته و از اختیار بایاس کاملاً خارج می‌سازند.

در مرحله بعدی لینوکس به جستجو و شناسایی سخت‌افزارهای موجود بر روی سیستم می‌پردازد و این کار را هر بار که شما سیستم خود را خاموش و سپس روشن نمایید انجام خواهد داد چراکه اطلاعات مربوطه به سخت‌افزار سیستم مثل نوع مادربرد، هارد دیسک، چیپست کارت گرافیک، ماوس و ابزارهای شبکه و ... در هر بار بوت شدن لازم و ضروری‌اند و لینوکس نمی‌تواند و نباید این اطلاعات را به خاطر بسپارد چون هر کس ممکن است در طول در دو مرحله بوت کامپیوتر خود نوعی از سخت‌افزار را به آن افزوده از آن بکاهد.

### **محیط x windows چیست؟**

معمولاً برای راحتی تلفظ به این محیط **X** نیز گفته می‌شود که بر پایه **GUI** بنا نهاده شده و یک محیط گرافیکی قوی برای راحتی کار در **LINUX** است. در این محیط شما می‌توانید چندین پنجره **Terminal** را به‌طور هم‌زمان در یک صفحه داشته باشید. به‌طوری‌که در هر پنجره یک کاربر **login** کرده باشد. معمولاً در محیط **X** و هر محیط گرافیکی دیگر می‌توان از ماوس برای راحتی بیشتر استفاده کرد. خیلی از برنامه‌ها مانند بازی‌ها و نرم‌افزارهای گرافیکی و کاربردی برای محیط **X** نوشته شده

است **linux** از دو محیط متنی و گرافیکی تشکیل شده است. معمولاً محیط متنی به دلیل کاربرپسند کاربران مبتدی قرار نمی‌گیرد. ولی این محیط برای کاربران حرفه‌ای‌تر از لینوکس برای مصارف شبکه استفاده می‌کنند. رضایت‌بخش است زیرا دوام سرویس‌های نصب‌شده و در حال استفاده در محیط متنی بسیار بیشتر از دوام سرویس‌ها در محیط گرافیکی است. همراه با لینوکس دو محیط گرافیکی بانام‌های **GNOME** و **KDE** ارائه می‌شود. همراه با این محیط‌ها ابزارهایی برای تنظیم گزینه‌هایی مانند رنگ‌ها، شکل پنجره‌ها، ماوس، تصویر پس‌زمینه، منوها و ... وجود دارد. به طوری که هر کاربر می‌تواند تنظیمات جداگانه خود را از سایر کاربران داشته باشد هنگامی که فرایند ورود به سیستم خاتمه یافت محیط گرافیک به طور خودکار اجرا می‌شود.

یکی از محیط‌های گرافیکی لینوکس که محیطی راحت و دوست‌داشتنی را باری کاربران خود فراهم می‌کند **KDE** نام دارد. این محیط بسیار شبیه دسکتاپ در ویندوز است. البته برخلاف ویندوز لینوکس دارای چندین دسکتاپ در یک محیط گرافیکی است به این نکته باید توجه کرد که سرعت محیط‌های گرافیکی لینوکس در حد ویندوز یا حتی کمتر از آن است زیرا **GUI** در ویندوز ب صورت ب همراه سیستم عامل است ولی در لینوکس محیط‌های گرافیکی به عنوان برنامه‌هایی جدا از سیستم عامل هستند و گاهی بر روی لینوکس باز اضافی ایجاد می‌کنند.

## پروژه **KDE** و **GNOME**

پروژه **GNOME** (**GNU Network Object Model Environment**) دو امکان را فراهم می‌کند: محیط میز کار **GNOME** یک دسکتاپ ویزوال و جالب برای کاربران مختلف و دوم زیرساخت‌های برنامه‌سازی به معنی چارچوبی گسترده برای ساخت کاربردهایی که با سایر برنامه‌های دسکتاپ یکپارچه می‌شوند. برخی از خصوصیات **GNOME** عبارت‌اند از:

قابل استفاده: قابل استفاده بودن در واقع به مفهوم ایجاد نرم‌افزاری است که برای همگان چه کاربران و چه توسعه‌دهندگان نرم‌افزار قابل استفاده باشد این نکته همواره مورد توجه خاص جامعه **GNOME** می‌باشد.

دسترسی آسان و بین‌المللی: **GNOME** به زبان‌های مختلف توسعه یافته. مستندسازی شده و مرود استفاده قرار می‌گیرد. گروه **GNOME** در تلاش است این اطمینان را بدهد که تمامی بخش‌های نرم‌افزار را بتواند به کلیه زبان‌ها ترجمه کند.

موردپسند برنامه نویسان (**developer friendly**): توسعه‌دهندگان نرم‌افزار برای استفاده از **GNOME** تنها از یک زبان استفاده نمی‌کنند شما می‌توانید از **C, C++, Python, Perl, Java** و یا حتی **c#** برای ایجاد کاربرهای سطح بالایی که با سایر برنامه‌های دسکتاپ یونیکس و لینوکس یکپارچه می‌شوند استفاده کنید.

سازمان‌یافتگی: **GNOME** تلاش می‌کند که یک انجمن سازمان‌یافته با ساختاری متشکل از صدها عضو و تشکیلات منظم باشد. توزیع‌ها **GNOME** توسط تیم مخصوص این کار و طبق برنامه هر ۶ ماه یکبار می‌شود.

پشتیبانی: **GNOME** توسط شرکت‌ها بانفوذ در لینوکس و یونیکس پشتیبانی می‌شود برخی از این شرکت‌ها عبارت‌اند از **Sun, hp, mandrakesoft, novell, redhat, canonical** و شرکت **Sun**.

پروژه (K Desktop Environment) KDE در اواخر سال ۱۹۹۶ به وجود آمد. اهداف KDE عبارت بودند از آماده‌سازی بستری مناسب برای نوشتن نرم‌افزار برای یونیکس و گنو/لینوکس و فراهم کردن محیط گرافیکی جذاب برای ایستگاه‌های کاری یونیکس و گنو/لینوکس

KDE در ابتدا یک پروژه عمدتاً آلمانی بود که به مرور زمان گسترش یافته و امروزه بدل به شبکه‌ای از مهندسان نرم‌افزار معتقد به نرم‌افزار آزاد در سراسر جهان شده است. KDE که مخفف K Desktop Environment می‌باشد با پیشرفت خیره‌کننده‌اش طی سال‌های اخیر تبدیل به تاج طلایی پروژه‌های نرم‌افزار آزاد گردیده است به گونه‌ای که یکی از جدیدترین نسخه‌های آن یعنی نسخه ۱/۳ هم از نظر زیبایی و چشم‌نوازی و هم از نظر قابلیت و کارایی با محیط‌های ویندوز و مکیتاش رقابت می‌کند و به اعتقاد بسیاری حتی از آن‌ها پیشی گرفته است.

محیط میز کار KDE شامل‌ها و نرم‌افزارهای گوناگون و متنوعی می‌باشد نرم‌افزارهای شبکه برنامه FTP و پست الکترونیکی و گفتگوی مستقیم (Kopete) نرم‌افزارهای گرافیکی و ویرایش تصویر (Kontour) پخش موسیقی (noatun) پخش فیلم (aktion) و کار با دوربین‌ها دیجیتال (kamera) و پویشگر (kooka) و چاپگر (kdeprint) نرم‌افزارهای مدیریت سیستم همانند مرکز کنترل مرورگر صفحات وب (konquerur) نرم‌افزارهای نمابر (kfax) طراحی صفحات وب (quanta) محیط برنامه‌نویسی (kdevelop) تعدادی بازی (kdgames) نرم‌افزارهای آموزشی (kdeedu) و یک مجموعه کامل نرم‌افزارهای اداری (koffice) شامل واژه‌پرداز (kword) صفحه گسترده (kspread) نمایش (kpresenter) و بسیاری نرم‌افزارهای دیگر محیطی کامل را برای کاربران فراهم کرده‌اند.

با توجه به سرعت پیشرفتی که KDE از خود نشان داده است پیش‌بینی آینده آن کاری بس دشوار است. به‌تازگی دولت آلمان طی پروژه‌ای به نام Kroupware اقدام به اضافه کردن پاره‌ای امکانات به KDE کرده است. سایر دولت اروپایی نیز همگی مشغول بررسی KDE می‌باشند

شرکت Apple برای ساخت مرورگر جدید خو به نام safari از مرورگر Konqueror استفاده کرده و پیشرفت‌های خود به این مرورگر را در اختیار پروژه kde قرار داده است. همگی نشانه‌ها بیانگر آن است که آینده درخشانی در انتظار این پروژه می‌اشد و سناریوی یونیکس سخت است دیگر صادق نیست. ترکیب KDE / لینوکس آماده فتح رایانه‌های رومیزی می‌باشد.

## آشنایی با بخش‌های مختلف محیط‌های گرافیکی

سیستم عامل لینوکس از دو بخش گرافیکی

Interface GUI (Graphical user Interface) و محیط متنی

CLI (Command Line Interface) تشکیل شده است

برای رفتن به محیط GUI با فشار دادن دکمه Ctrl+Alt+F1 و یا Ctrl+Alt+F2...F6 می‌توانید به آن دستیابی پیدا کنیم و با فشار دادن Ctrl+Alt+F7 می‌توانیم به محیط گرافیکی وارد شویم

## محیط گرافیکی GNOME

هنگامی که شما از XWINDOWS در GNOME استفاده می‌کنید چندین آیتم می‌بینید از جمله:

## محیط کاری (Desktop)

فضای اصلی کار شماست که بیشترین فضای صفحه را اشغال کرده است. کلید پنجره‌ها روی این محیط قرار می‌گیرند. مدرک پوشه میانبرها و دیگر داده‌ها نیز روی این محیط قرار می‌گیرند.

### پنل (panel)

میله بلندی است که سرتاسر پایین محیط کاری را پر کرده است. تعدادی از میانبرها و اپلت‌ها روی آن قرار دارد.

Applet از کلمه **mini application** به معنی درخواست‌های کوچک گرفته شده است.

پنل قابل تغییر و پیکربندی است شما می‌توانید ابزارهای و اپلت‌هایی را روی پنل بگذارید و از آن‌ها استفاده کنید در ضمن می‌توانید با کلیک کردن روی علامت‌های فلش مانند در دو طرف پنل در سمت چپ یا راست پنهان کنید.

### دکمه منوی اصلی

معمولاً اولین دکمه از سمت چپ روی پنل است که به شکل یک جای (آرام GNOME) این منو شامل زیر منوها اپلت‌ها و ابزارهای کاربردی است. بیشتر برنامه لینوکس را نیز می‌توان از آنجا یافت. البته می‌توان چیزهایی نیز به آن افزود.

### میانبرها (Launchers)

آیکون‌هایی هستند که روی محیط‌های کاری می‌آیند و شما با یک کلیک روی آن‌ها می‌توانید به برنامه یا محیط مربوطه دسترسی پیدا کنید مثلاً اگر روی آیکون Netscape کلیک کنید به این مرورگر وب دسترسی پیدا خواهید کرد.

### پوشه home

پوشه دسترسی شما را به فهرست‌هایی مخصوص کاربران که در `/home/youruser` قرار دارد امکان‌پذیر می‌کند.

اضافه کردن اپلت‌ها:

روی پنل کلیک راست کنید و **Applet** را انتخاب کنید در پنجره جدید اگر روی ابزار موردنظر خود کلیک کنید آن ابزار به پنل اضافه خواهد شد.

### استفاده از راهنما (help)

در روی پنل دکمه‌ای وجود دارد که یک علامت سؤال روی آن دیده می‌شود با کلیک روی آن می‌توانید به راهنمای GNOME دسترسی پیدا کنید. این راهنما یکی از مفیدترین ابزار است که بسیار به شما کمک می‌کند.

## محیط کاری چندگانه

اگر صفحه کاری شما کمی شلوغ شد به طوری که ما را گیج می‌کرد یعنی مثلاً اگر در هنگام کار مجبور شدید چندین پنجره را باهم باز کنید می‌توانید کارتان را در چندین صفحه کاری تقسیم کنید و در روی هیئت‌رئیس در سمت راست یک مربع وجود دارد که به چهار قسمت تبدیل شده است. با کلیک روی هر کدام از این قسمت‌ها به صفحه مربوطه می‌روید. در حقیقت شما به طور پیش فرض چهار صفحه کاری دارید که می‌توانید این تعداد را کم و زیاد کنید.



سویچ کردن بین محیط‌ها کاری

در لینوکس ابزاری به نام **switchdesk** وجود دارد که این امکان را برای شما ایجاد می‌کند که در هنگام کار در یکی از محیط‌های گرافیکی به یکی از محیط‌های دیگر بروید. مثلاً شما در **GNOME** کار می‌کنید و می‌خواهید به **KDE** بروید باید با استفاده از این ابزار رابط گرافیکی را به **KDE** تغییر داده و از سیستم خارج شوید. حال اگر دوباره وارد سیستم شوید خود را در رابط گرافیکی **KDE** خواهید یافت. برای اجرای این برنامه از منوی اصلی **GNOME** گزینه **Programs** سپس گزینه **system** و سپس گزینه **desktop switching tool** را انتخاب کنید

پایان کار در **GNOME**

در منوی اصلاً **GNOME** دکمه‌ای به نام **log out** وجود دارد که شما با استفاده از این دکمه می‌توانید عمل **shutdown** یا **reboot** کردن را انجام دهید. همچنین می‌توانید از محیط گرافیکی که در آن هستید خارج شوید.

### محیط گرافیکی **KDE**

شکل ظاهری آن تقریباً شبیه **GNOME** است. هنگام ورود به این محیط **panel,desktop** و ... را می‌بینید. مفاهیم **panel,desktop**. محیط کاری چندگانه. پوشه‌ها و ... در قسمت قبل توضیح داده شد. این مفاهیم دقیقاً در مورد **KDE** هم صادق است.

پس در اینجا چند ویژگی متفاوت **KDE** را می‌گوییم.

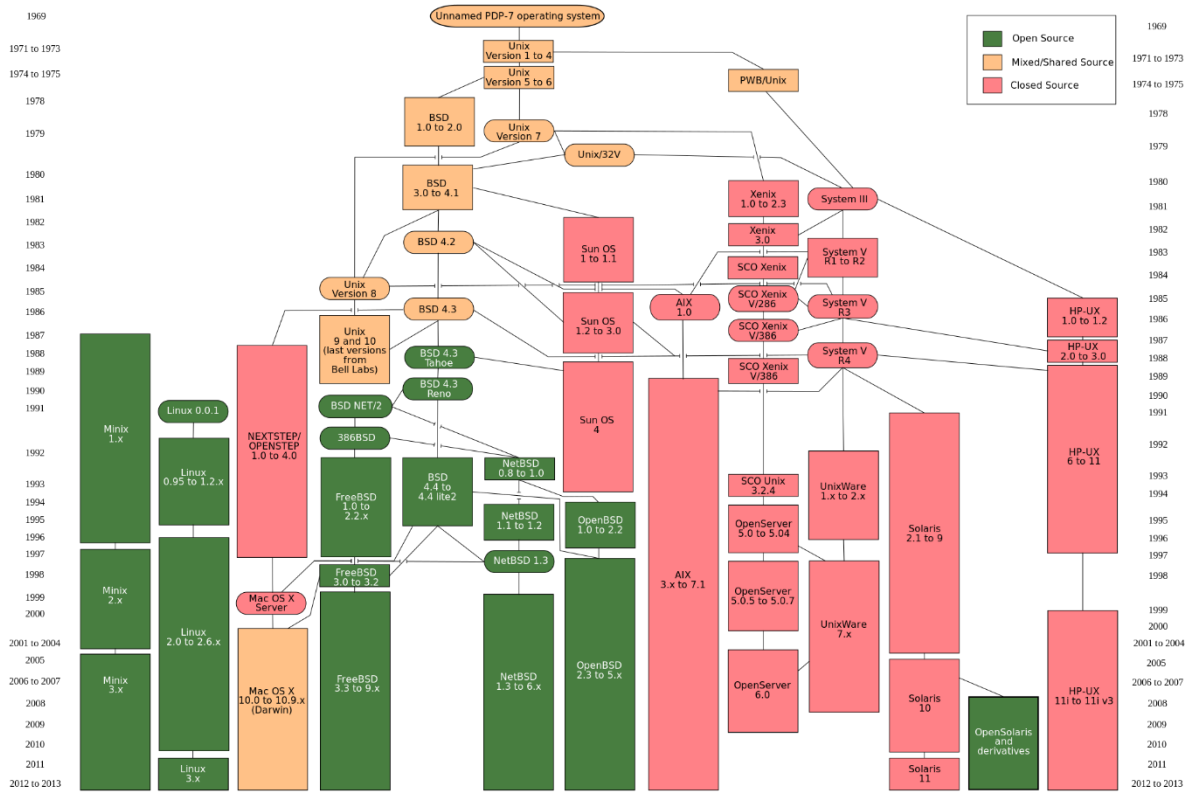
### **CD-Rom, Trash, Floppy**

آیکون‌هایی هستند که بر روی صفحه کاری دیده می‌شوند. آیکون **Floppy CD-Rom**, دستیابی شما را به فلاپی درایو و سی دی آسان می‌سازند. یعنی با یک کلیک روی آن‌ها می‌توانید محتویات **cd** یا **floppy** را مشاهده کنید. آیکون **Trash** هم همان سطل آشغال **KDE** است که آن را در ویندوز با نام **Recycle Bin** می‌شناسید برای حذف یک پوشه یا فایل می‌توانید آن را با ماوس گرفته و به درون سطل آشغال بیندازید.

آیکون منوی اصلی **k** آیکونی روی پنل شما وجود دارد که یک حرف بزرگ **k** روی آن دیده می‌شود. در این منو می‌توانید انواع ابزارها و برنامه‌ها و زیر منوهای موجود در **KDE** را بیابید. این برنامه و زیر منوها قابل تغییرند یعنی قابل حذف و اضافه هستند


### راهنمای **KDE**

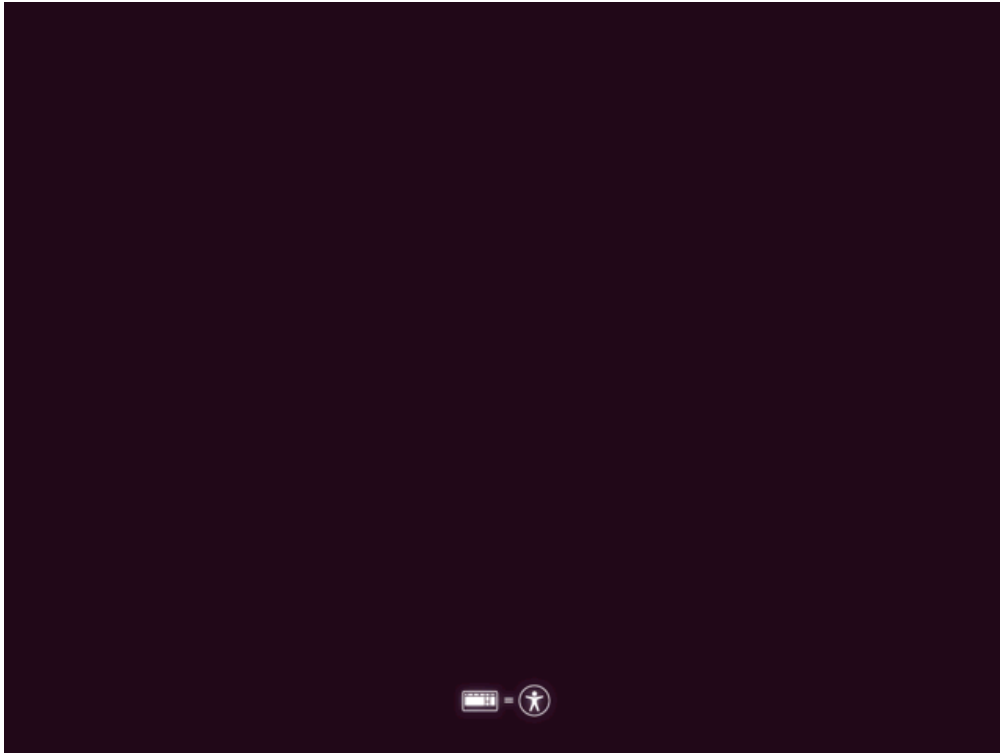
همانند **GNOME** و **KDE** هم یک راهنمای مفید و پرکاربرد دارد که می‌تواند کمک خوب برای شما باشد. این راهنما شامل راهنمای اقسام مختلف **KDE** راهنمای پوشش سیستم. سفارش کردن محیط کاری و دیگر کارهای سیستم شماست.



## طریقه ی نصب Ubuntu 16.04 از طریق CD: Boot from CD

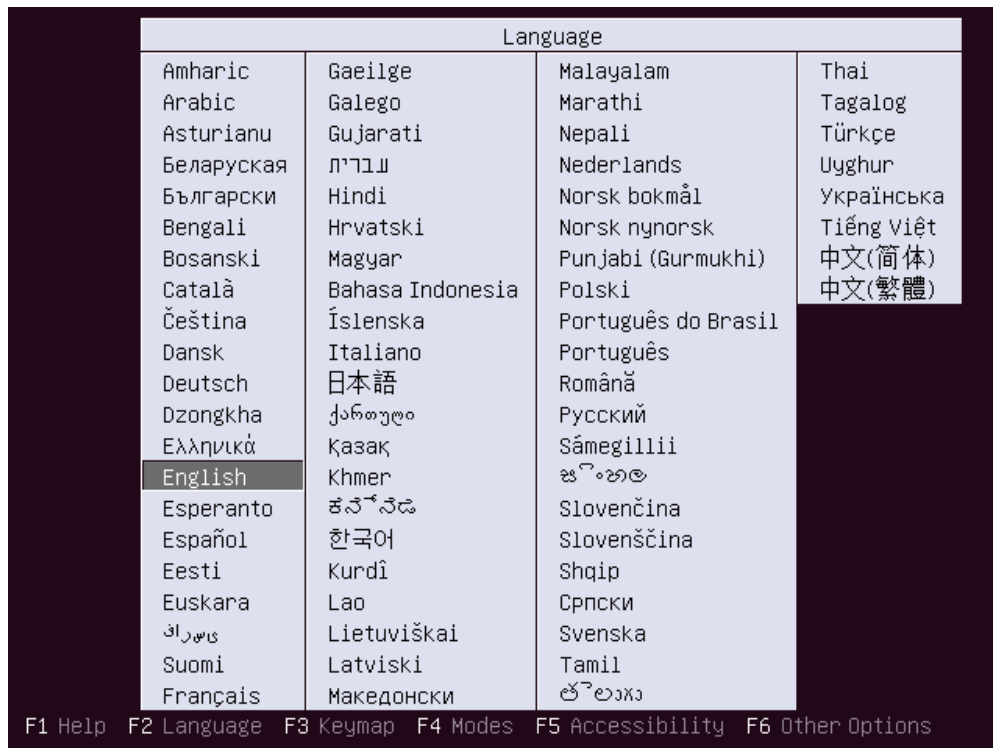
۱. اولین صفحه بعد از بوت از سی دی:

در این قسمت کلید  را بزنید

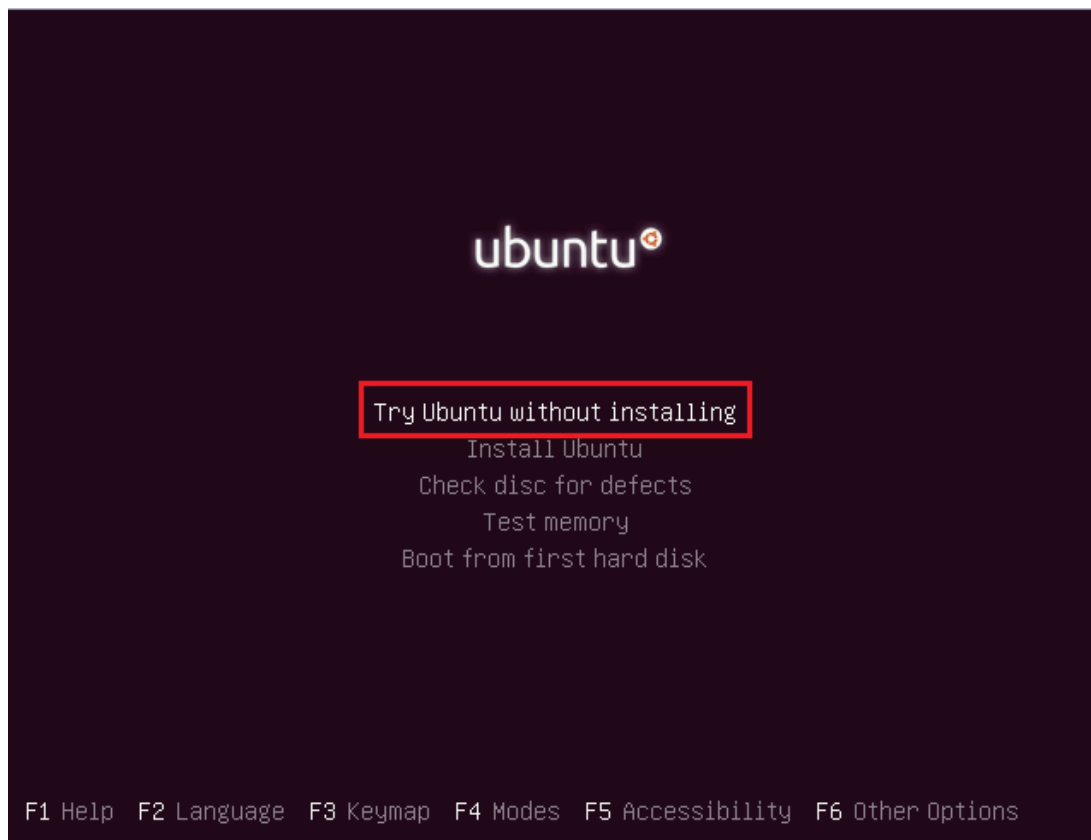


۲. انتخاب زبان:

کلید Enter را بزنید:

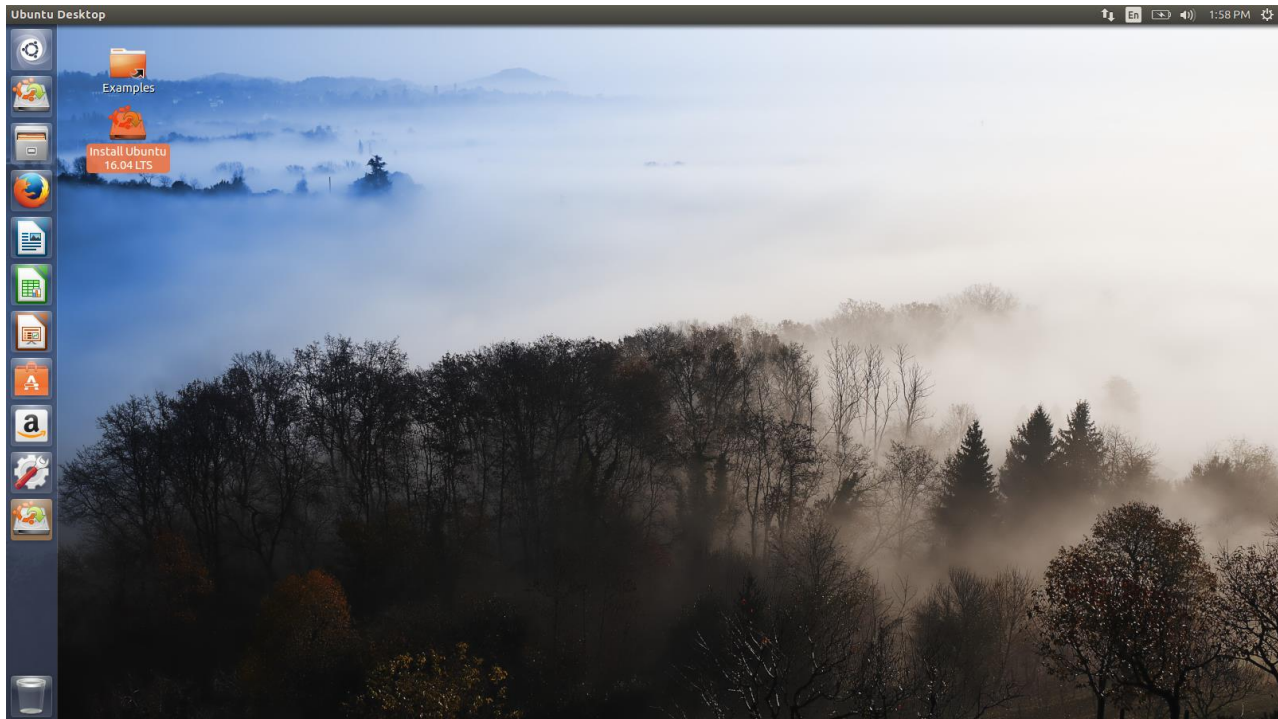


۳. در این مرحله برای تست سیستم عامل Ubuntu میتوانیم گزینه Try Ubuntu without installing را انتخاب کنیم و در صورتی که بخواهیم مستقیماً به نصب برویم گزینه Install Ubuntu را می‌زنیم، در این آموزش من از گزینه Try Ubuntu without installing را انتخاب کرده‌ام:

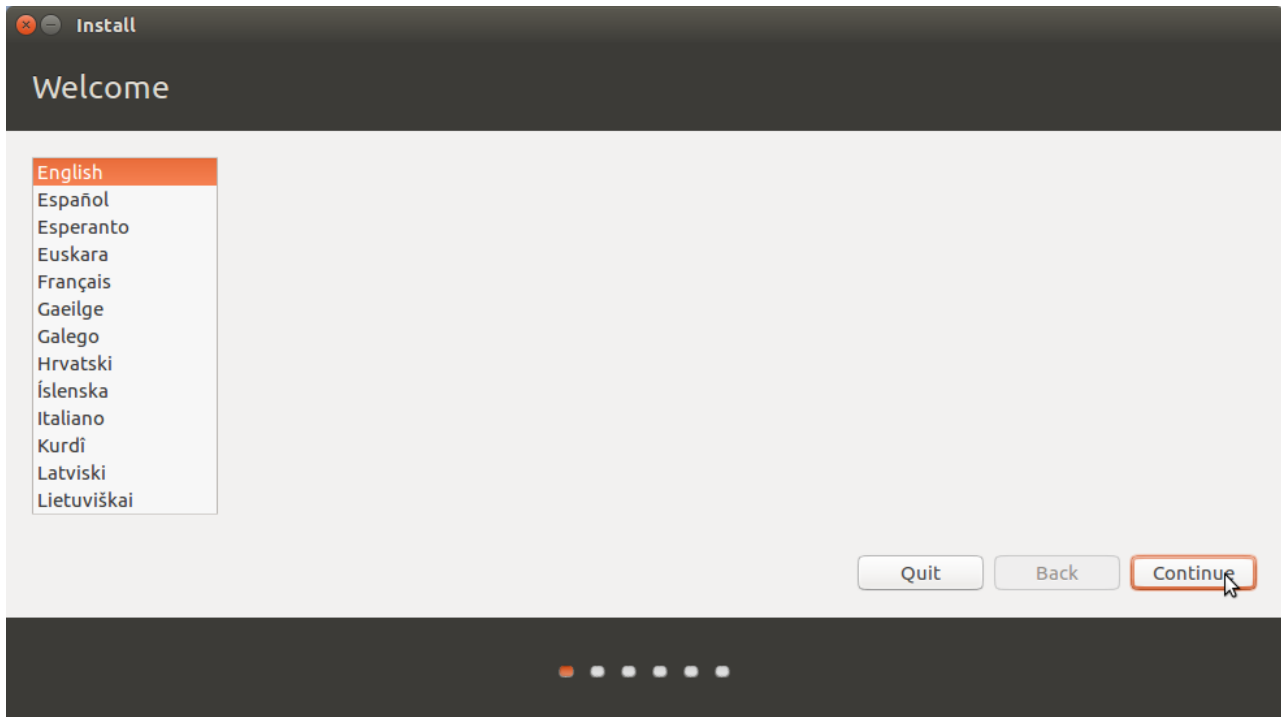


۴. صفحه دستکتاب Ubuntu پس از لود فایل ها به شکل زیر میباشد با انتخاب گزینه Install Ubuntu وارد ستاپ

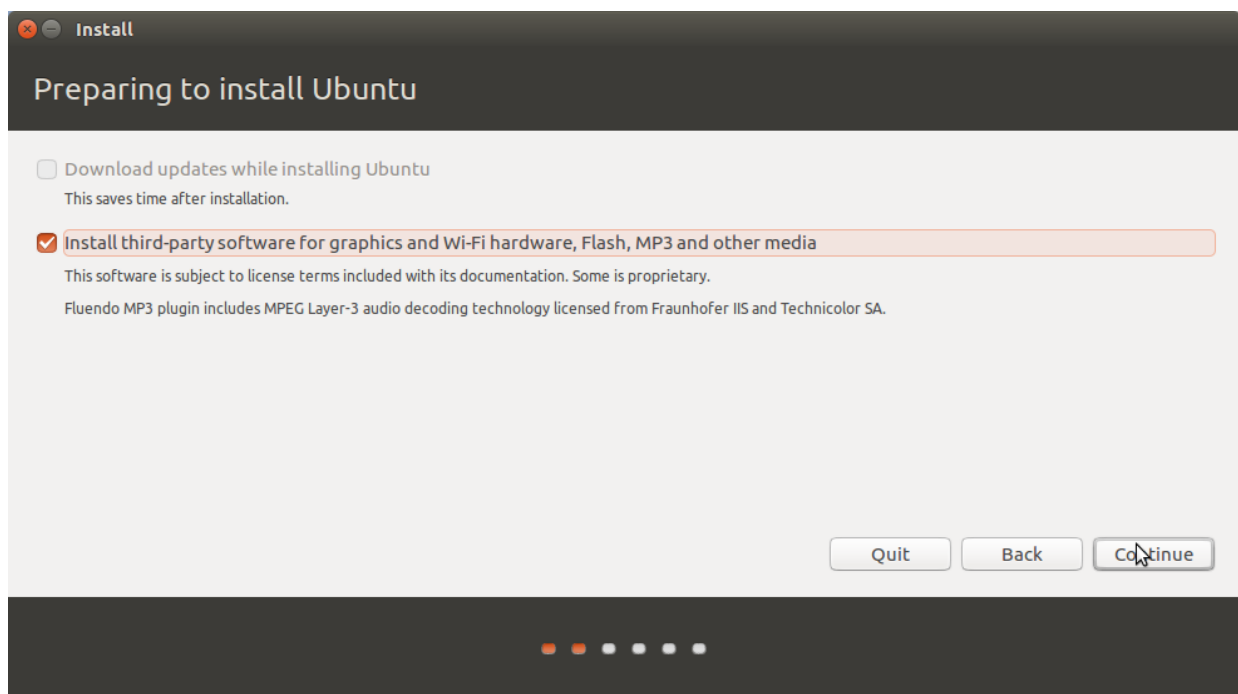
نصب می شویم:



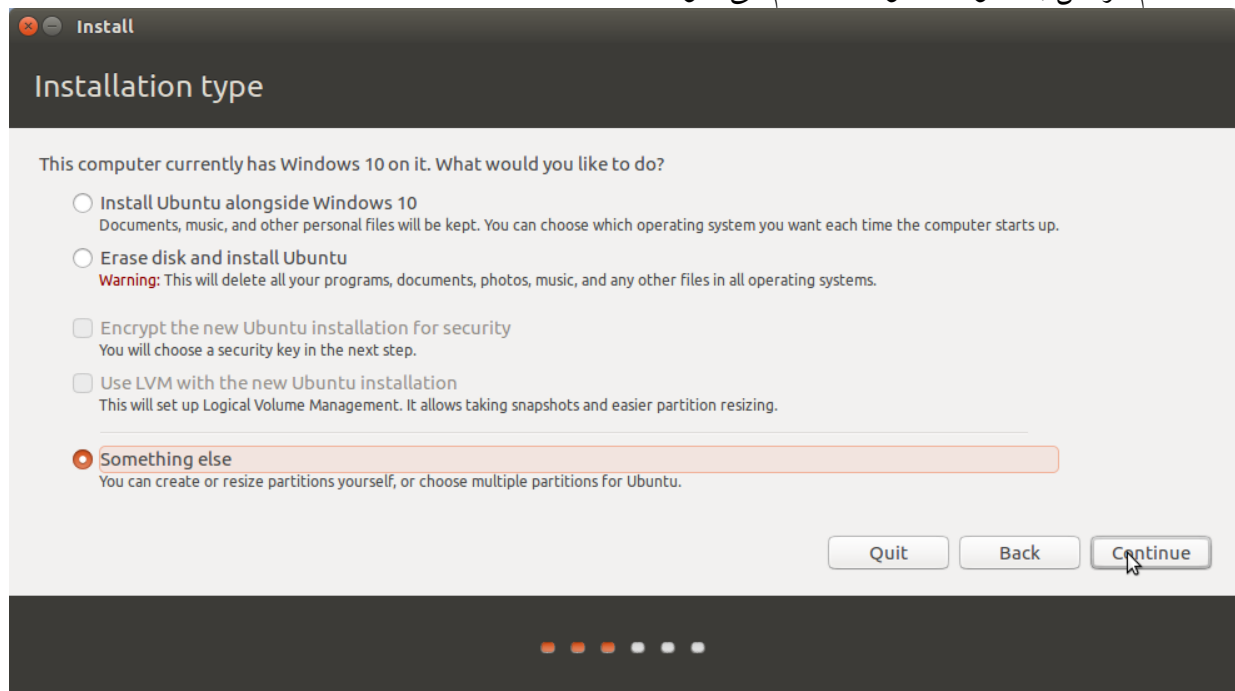
۵. زبان را انتخاب کرده و بر روی گزینه Continue کلیک می کنیم:



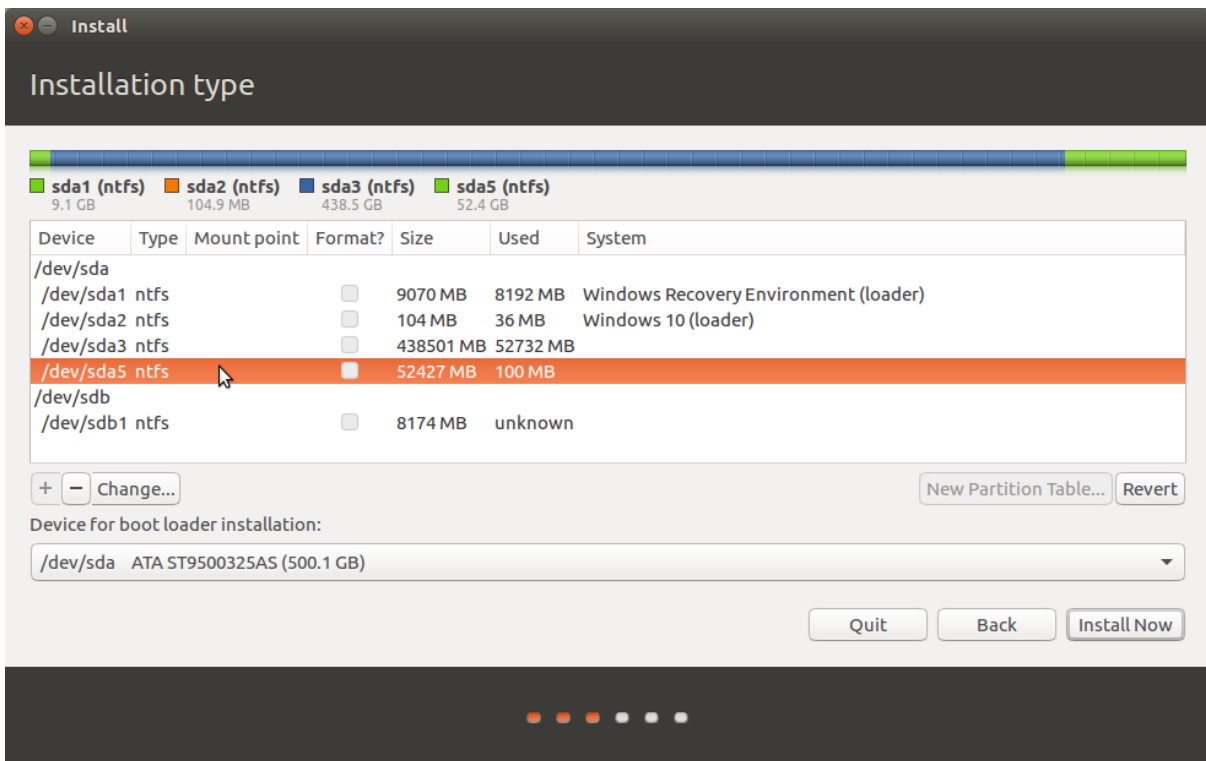
۶. در این مرحله برای نصب آپدیت ها تیک گزینه اول و برای نصب قابلیت پخش mp3، نصب درایور گرافیک ، نصب Wi-Fi و... تیک گزینه دوم را می زنیم ، لازم به ذکر است که در صورت نزدن تیک هیچ یک امکان انجام آنها در خود سیستم عامل وجود دارد.



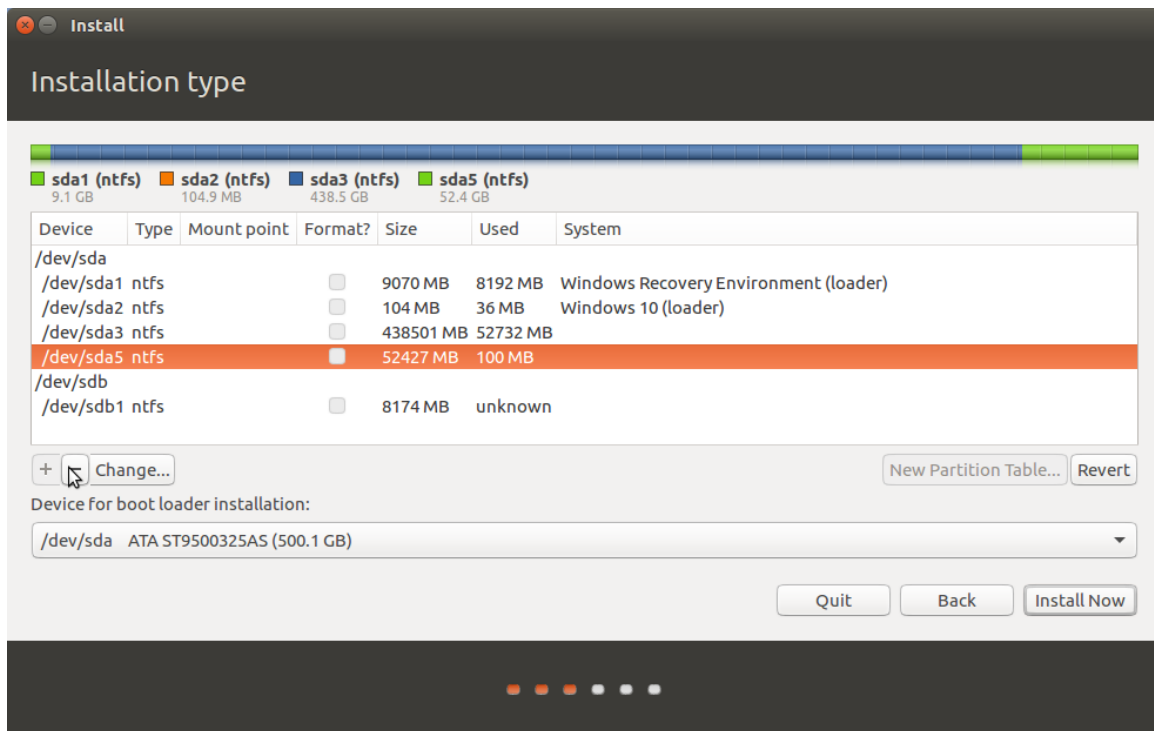
۷. در این مرحله گزینه **Something else** را انتخاب میکنیم تا به صورت دستی برای نصب **Ubuntu** ، پارتیشن های مورد نیاز برای نصب را به وجود آوریم ، لازم به ذکر است که در صورت نیاز و عدم آگاهی از طریقه به وجود آوردن پارتیشن های سیستمی می توان گزینه اول را برای نصب بدون انجام دستی عمل پارتیشن بندی انتخاب کرد ، در این حالت تمام مراحل به صورت خودکار انجام می شود.



۸. در این مرحله من از قبل حدود ۵۰ گیگا بایت از هارد خود را خالی کردم که مشاهده میکنید به صورت زیر نمایش داده می شود ، می توانید برای راحتی کار در داخل ویندوز خود ، از طریق **Shrink** در داخل **Disk management** یک پارتیشن جدید و جدا کردن آن از پارتیشن های دیگر با سایز دلخواه بوجود آورید و سپس **Ubuntu** را بر روی آن نصب کنید:

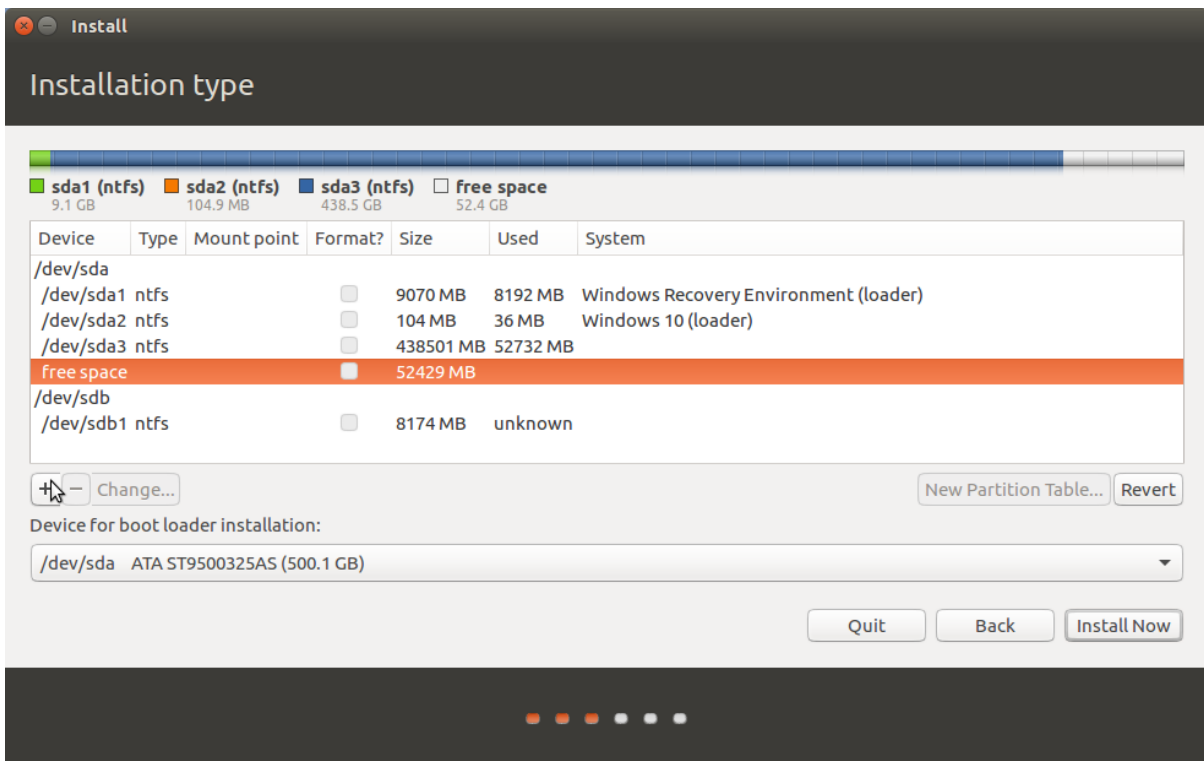


۹. بر روی گزینه منفی برای دیلیت کردن پارتیشن کلیک کنید:





۱۰. پس از زدن گزینه منفی، پارتیشن دیلیت شده و به صورت Free Space یا فضای خالی در می آید سپس بر روی گزینه + کلیک کرده تا صفحه پارتیشن بندی باز شود که حال به نکات زیر توجه نمایید:



### در داخل صفحه ای که باز می شود با گزینه های زیر روبرو می شوید:

۱. در گزینه اول باید سایز را انتخاب کنید که بسته به پارتیشنی که بوجود می آوریم سایز را باید انتخاب کنیم.
  ۲. در قسمت بعد می توانید نوع پارتیشن را انتخاب کنید:
- الف: Primary Partition توجه داشته باشید که تنها می توانید ۴ پارتیشن پرایمری بر روی هارد دیسک خود داشته باشید (MBR) و یک Extended که در داخل آن می توانید logical پارتیشن ها را قرار دهید، در واقع پرایمری پارتیشن محل قرار گیری ویندوز یا اوبونتو میباشد که میتواند به صورت اکتیو باشد ولی فقط یک پارتیشن پرایمری می تواند اکتیو باشد
- ب. می توانید برای سادگی تنها از گزینه logical استفاده کنید!
۳. در این قسمت محلی که میخواهید پارتیشن جدید شما شروع شود را انتخاب میکنید
  ۴. انتخاب نوع فایل سیستمی که شامل: btrfs, Ext2, Ext3, Ext4 و غیره می باشد

۵. در این مرحله باید **mount point** خود یا به عبارت دیگر به سیستم بگویید که چگونه میخواهید از این پارتیشن استفاده کنید را انتخاب نمایید که شرح آن به صورت زیر می باشد:

- **Root** : / یا به عبارت دیگر همان **C:** ویندوز می باشد ، این قسمتی است که فایل سیستمی ابونتو بر روی آن نصب می شود ، می توانیم ابونتو را از همین جا بوت کنیم یا جداگانه یک پارتیشن **/boot** به وجود بیاوریم.

- **/home**: دایرکتوری ، پارتیشنی جدا برای ذخیره فایل های شخصی و در واقع **user directory**.

- **/tmp**: یا همان **Temporary Directory** از **mount point** برای ذخیره محتوای وب سرور ها و ذخیره فایل های موقت سیستم استفاده می شود.

- **/usr**: این قسمت برای ذخیره فایل های باینری ، کتابخانه ی کامپایلرها و نرم افزارها استفاده می شود ، در این قسمت نرم افزارهایی قرار می گیرند که شامل **Dependency** یا وابستگی می باشند قرار میگیرند

- **Swap**: یا **Temporary virtual memory** در واقع رم مجازی برای بوت ابونتو می باشد ، در صورتی که رم زیاد دارید می توانید از **Swap** استفاده نکنید ولی استفاده از آن توصیه می شود بخصوص در سرور ها ، همچنین در لپ تاپ ها اگر نیاز به **Hibernate** دارید باید میزان **Swap** را تقریباً دو برابر مقدار رم خود قرار دهید!

- **/var**: در این قسمت **log** و فایل های ادمین ذخیره میشود که میتواند حجم آن زیاد باشد.

- **/srv**: شامل دیتاهای مربوط به سایت ها میشود همچنین میتوان از آن برای **share** در **samba** استفاده کرد

- **/opt**: در این قسمت نرم افزارهای **third party** نصب می شوند که به هیچ نرم افزار دیگری خارج از پیمانۀ آن پکیج وابستگی ندارند.

- **/usr/local**: در این قسمت نرم افزارهای بوجود آمده توسط ادمین که به صورت محلی نصب می شوند قرار میگیرد ، که معمولاً با دستور **make** بوجود می آیند. از این قسمت برای جلوگیری از درگیری فایل های به وجود آمده توسط ادمین با فایل های سیستم عامل استفاده می شود.

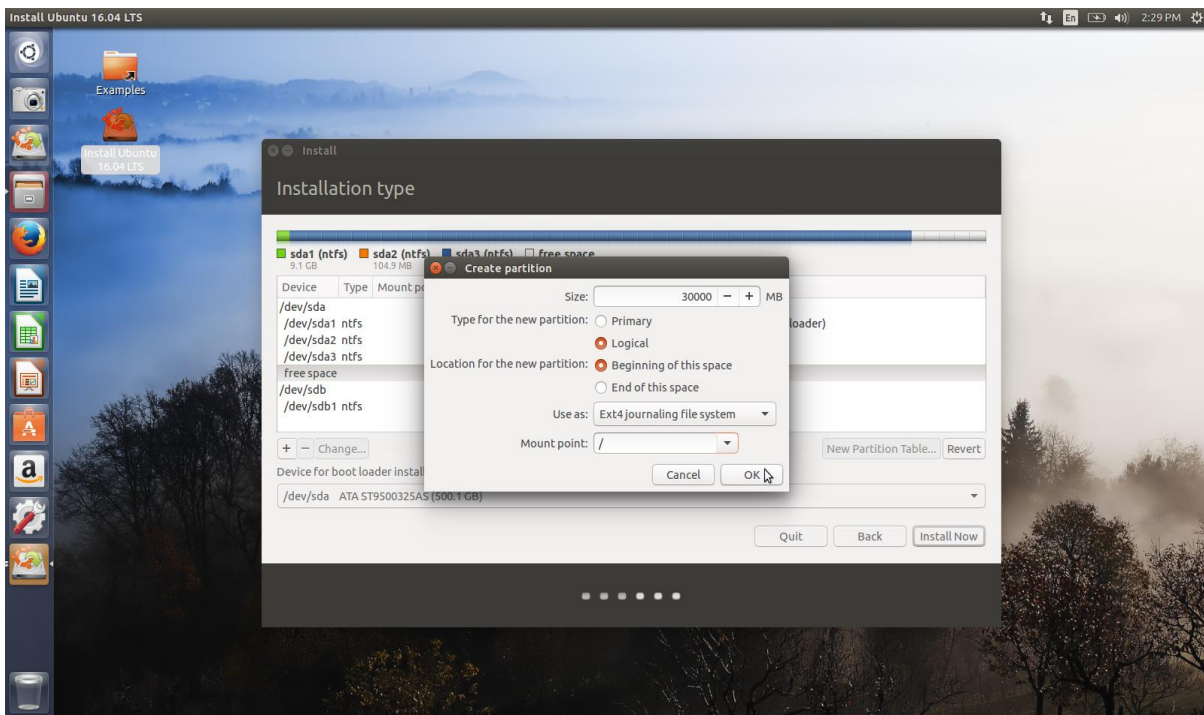
برای نصب ساده لینوکس شما به ۳ **mount point** زیر احتیاج دارید:

- / یا روت اجباری!

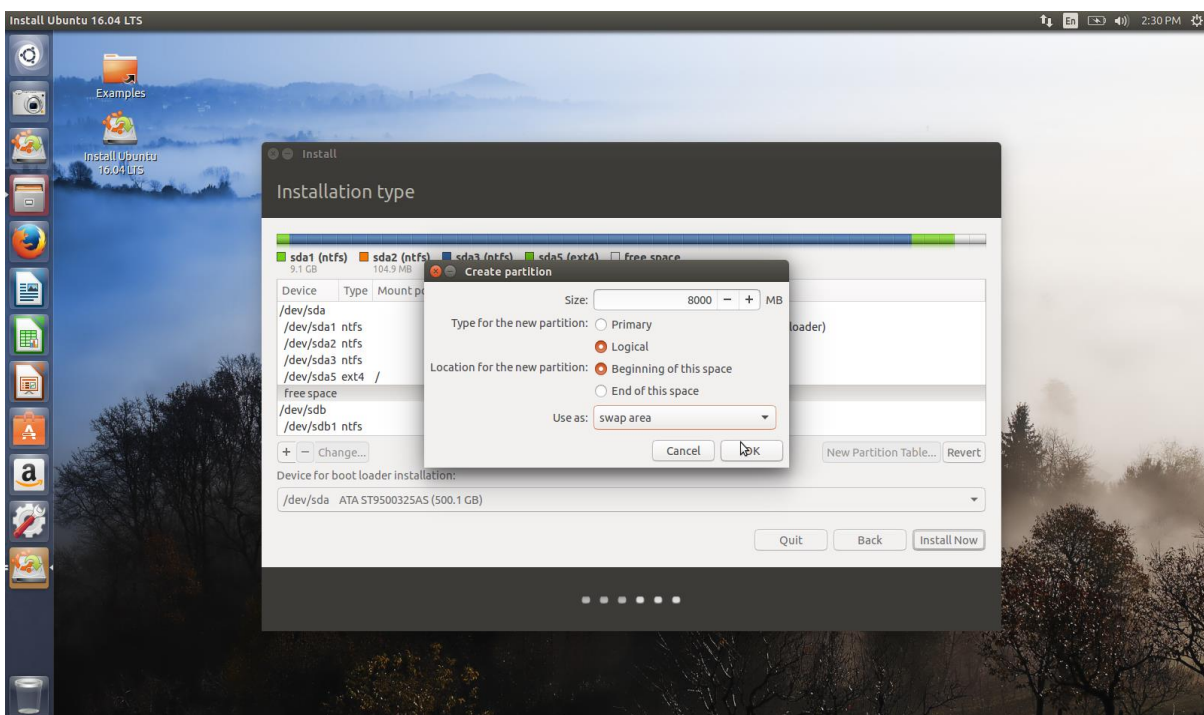
- **Swap** به صورت اختیاری!

• /home به صورت اختیاری!

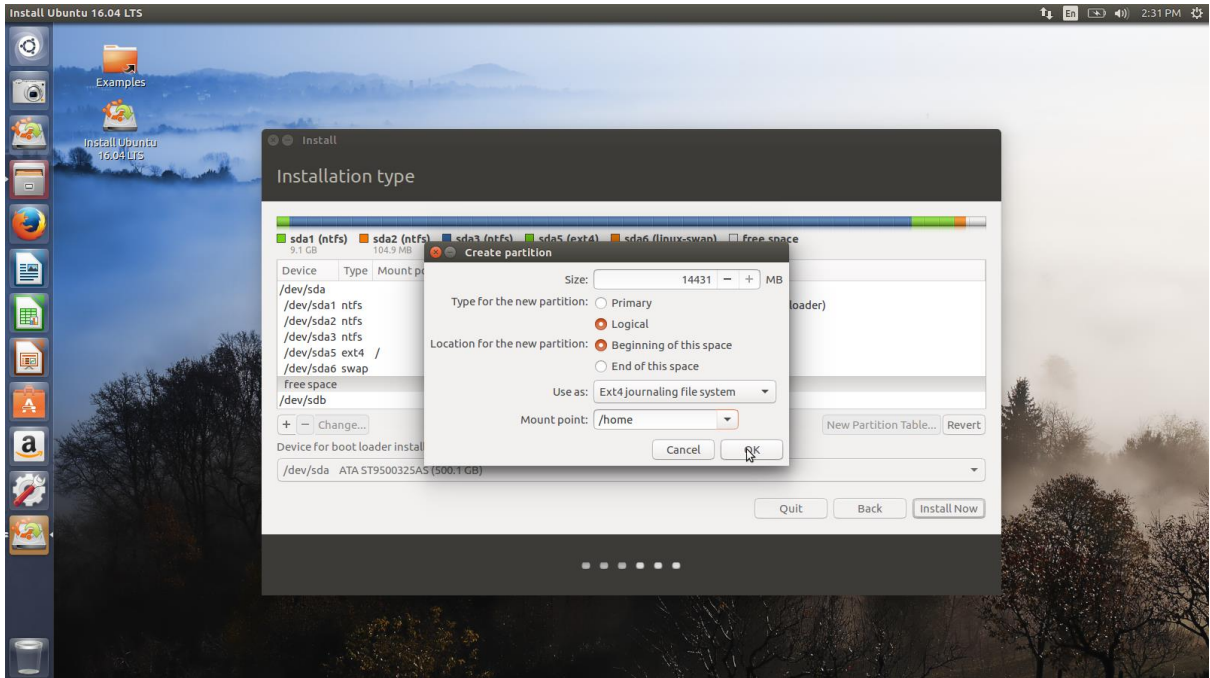
۱۱. در این مرحله پارتیشن روت را به وجود می آوریم و Ext4 را به عنوان فایل سیستمی در نظر میگیریم:



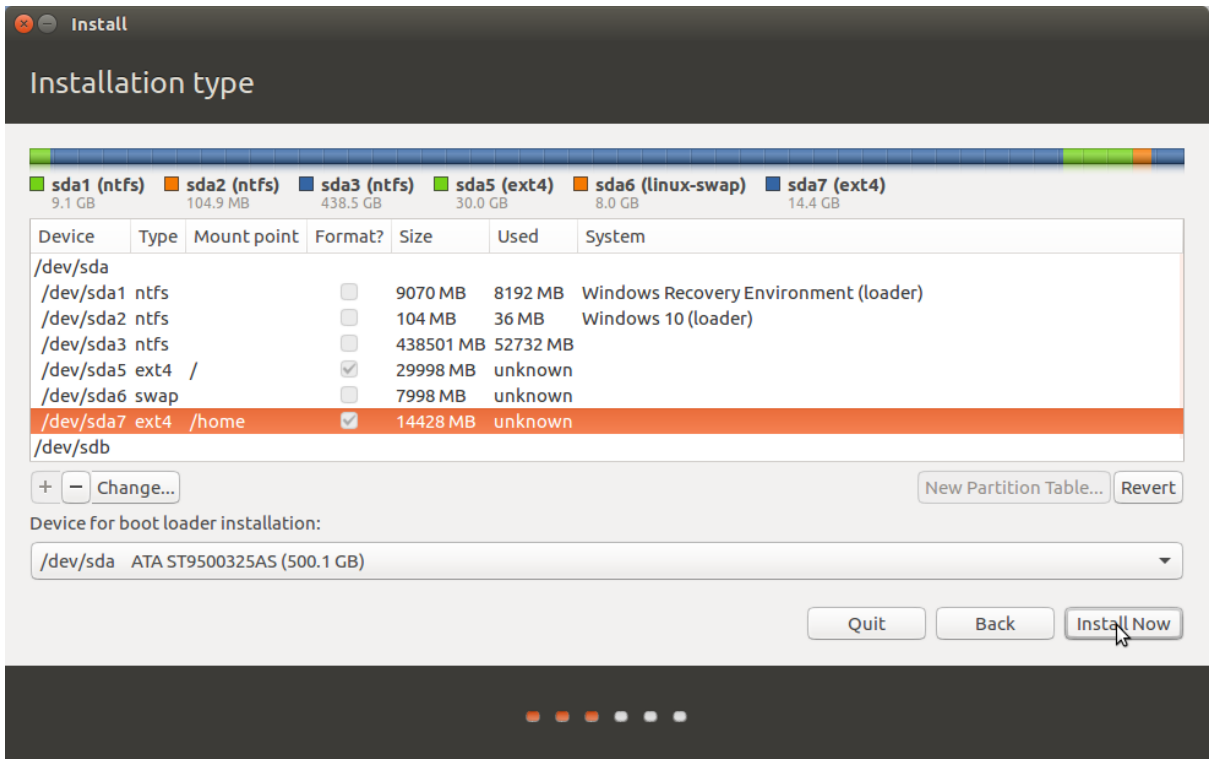
۱۲. در این مرحله Swap را بوجود می آوریم:



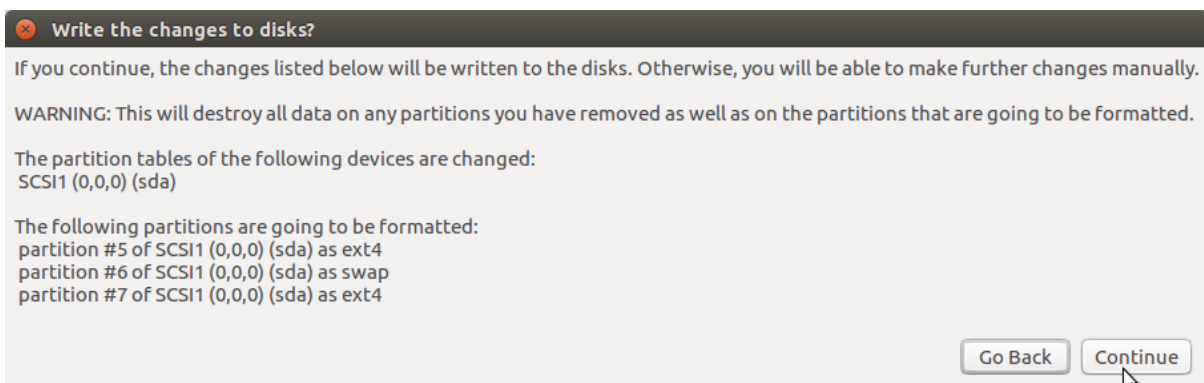
۱۳. در این قسمت /home را با فایل سیستمی Ext4 بوجود می آوریم:



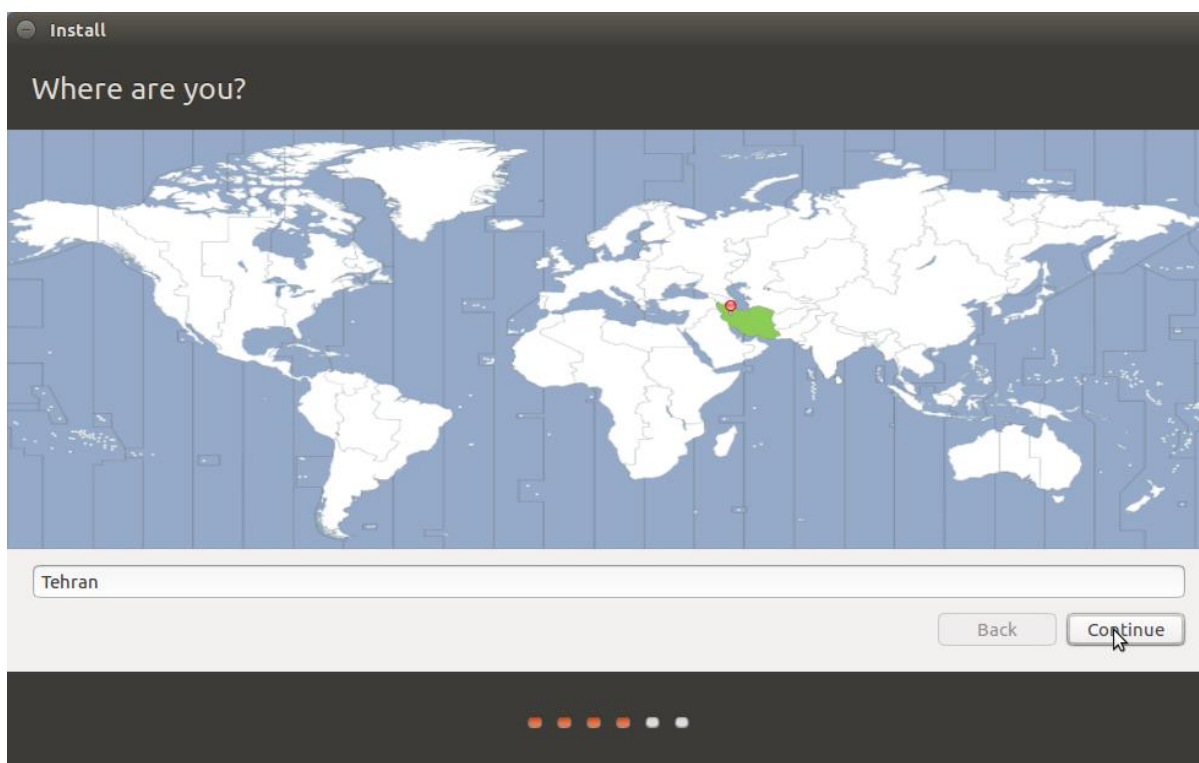
۱۴. حال بروی گزینه Install now کلیک می کنیم:



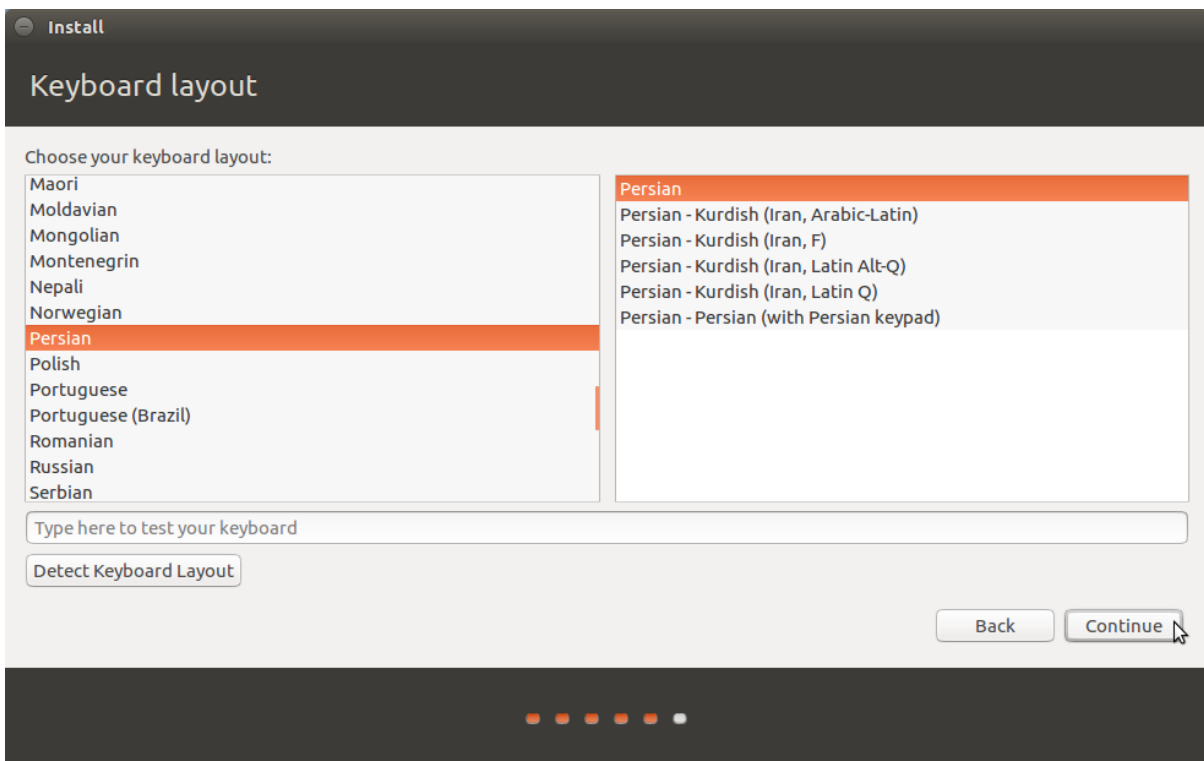
۱۵. باید صفحه زیر را پس از زدن **Install now** مشاهده کنید که جزئیات پارتیشن های بوجود آمده را نشان میدهد و سپس بر روی **Continue** کلیک کنید:



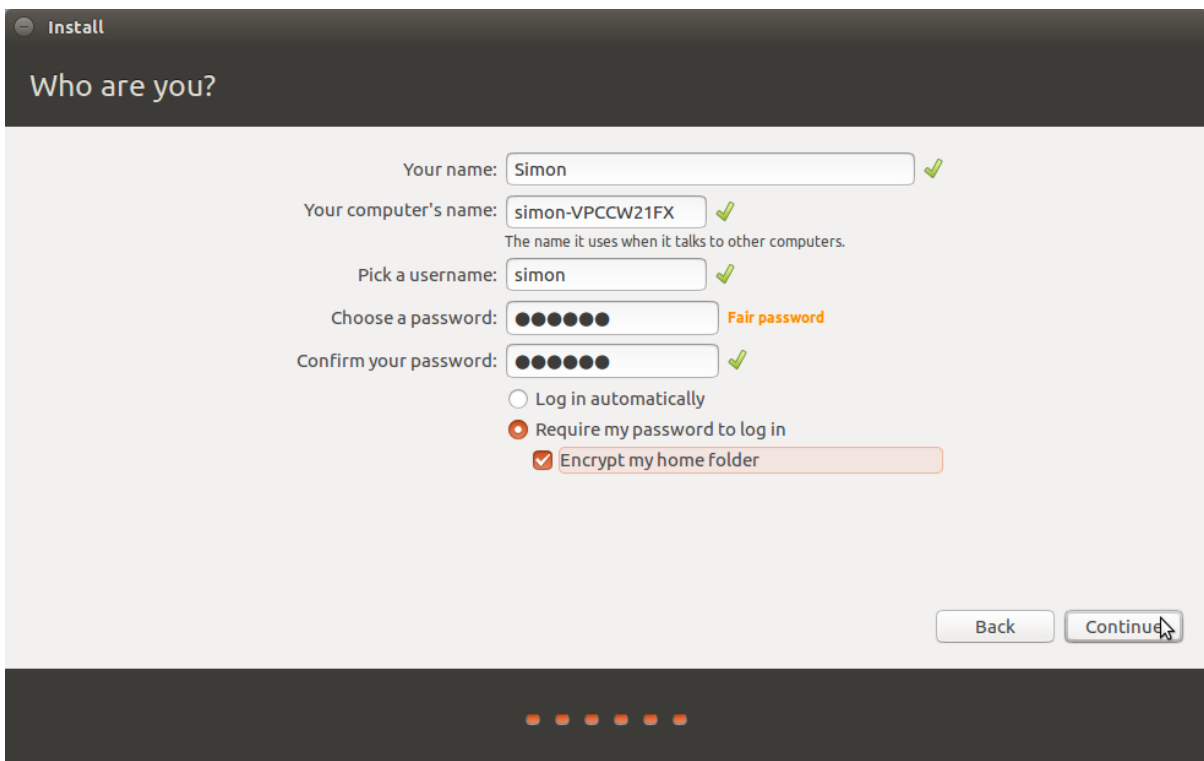
۱۶. انتخاب کشور:



۱۷. انتخاب کیبورد فارسی برای ابونتو و سپس گزینه Continue را کلیک کنید:



۱۸. ایجاد User account و دادن رمز به ابونتو، در این قسمت می توانید انتخاب کنید که Home folder شما به صورت Encrypted در بیاید:





رفع اشکال صفحه مشکی بعد از اولین بوت:

در صورتی که پس از اولین بوت ابونتو با صفحه مشکی روبرو شدید مراحل زیر را انجام دهید:

۱. کامپیوتر خود را روشن نموده و کلید شیفت سمت راست را بزنید ، در صورتی که dual boot هستید وقتی صفحه ی Grub را دیدید با کلید بالا و پایین Ubuntu را انتخاب کرده و کلید e را فشار دهید:

```
GNU GRUB version 1.99-21ubuntu3.1

Ubuntu, with Linux 3.2.0-26-generic
Ubuntu, with Linux 3.2.0-26-generic (recovery mode)
Previous Linux versions
Memory test (memtest86+)
Memory test (memtest86+, serial console 115200)

Use the ↑ and ↓ keys to select which entry is highlighted.
Press enter to boot the selected OS, 'e' to edit the commands
before booting or 'c' for a command-line.
```

۲. پس از زدن کلید e صفحه زیر را مشاهده خواهید کرد:

```
GNU GRUB version 1.99-21ubuntu3.1

setparams 'Ubuntu, with Linux 3.2.0-26-generic'

recordfail
gfxmode $linux_gfx_mode
insmod gzio
insmod part_msdos
insmod ext2
set root='(hd0,msdos1)'
search --no-floppy --fs-uuid --set=root 837ff500-baec-47e9-beb8-9e43\
5193d3b7
linux /boot/vmlinuz-3.2.0-26-generic root=UUID=837ff500-baec-47e9-be\
b8-9e435193d3b7 ro quiet splash $vt_handoff
initrd /boot/initrd.img-3.2.0-26-generic

Minimum Emacs-like screen editing is supported. TAB lists
completions. Press Ctrl-x or F10 to boot, Ctrl-c or F2 for
a command-line or ESC to discard edits and return to the GRUB
menu.
```



۳. با زدن دکمه های بالا، پایین، چپ و راست به قسمت کادر زرد در بالا رسیده و عبارت `nomodeset` را طبق شکل زیر وارد نمایید:

```
GNU GRUB version 1.99-21ubuntu3.1

setparams 'Ubuntu, with Linux 3.2.0-26-generic'

recordfail
gfxmode $linux_gfx_mode
insmod gzio
insmod part_msdos
insmod ext2
set root='(hd0,msdos1)'
search --no-floppy --fs-uuid --set=root 837ff500-baec-47e9-beb8-9e43\
5193d3b7
linux /boot/vmlinuz-3.2.0-26-generic root=UUID=837ff500-baec-47e9-be\
b8-9e435193d3b7 ro nomodeset $vt_handoff
initrd /boot/initrd.img-3.2.0-26-generic

Minimum Emacs-like screen editing is supported. TAB lists
completions. Press Ctrl-x or F10 to boot, Ctrl-c or F2 for
a command-line or ESC to discard edits and return to the GRUB
menu.
```

حال کلید `ctrl+x` یا `F10` را بزنید تا سیستم به محیطی برود که به راحتی بتوانید درایور گرافیک را نصب کنید

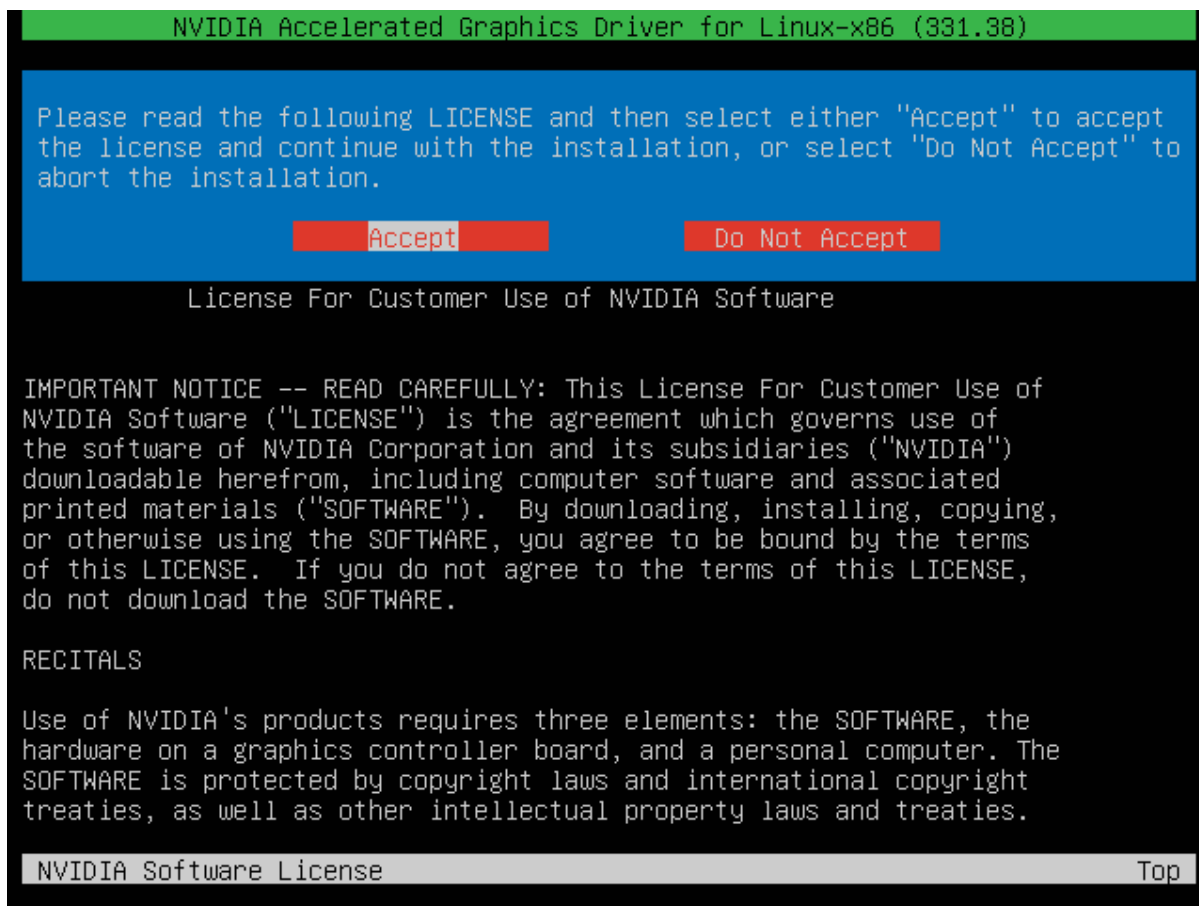
## نصب درایو گرافیک: (NVIDIA Driver)

برای نصب درایور گرافیک مراحل زیر را دنبال کنید:

- ابتدا کلید **CTRL+ALT+F1** را بزنید.
- وارد سیستم بشوید (توجه داشته باشید که به جای **login** باید نام کاربری سیستم و برای رمز، رمز عبور آن نام کاربری را وارد کنید، در صورت استفاده از اعداد از کلید های بالای حروف استفاده کنید)
- با استفاده از دستور **CD** به محلی بروید که فایل **.run** شما ذخیره شده است (مثال: **cd /home/Richard/Downloads**) ، در صورتی که در فولدر **home** است نیازی به این کار نیست
- دستورات زیر را وارد کنید:

```
۱. sudo su
۲. sudo service lightdm stop
۳. chmod +x NVIDIA-Linux-x86.run
۴. sudo sh NVIDIA-Linux-x86.run
```

توجه داشته باشید که به جای **NVIDIA-Linux-x86** باید نام فایل دانلود شده خود را بزنید!



The distribution-provided pre-install script failed! Continue installation anyway?

Yes

No

Building NVIDIA kernel module:

-----  
43%

NVIDIA Accelerated Graphics Driver for Linux-x86 (331.38)

Building Unified Memory kernel module:

-----  
100%

NVIDIA Software Installer for Unix/Linux

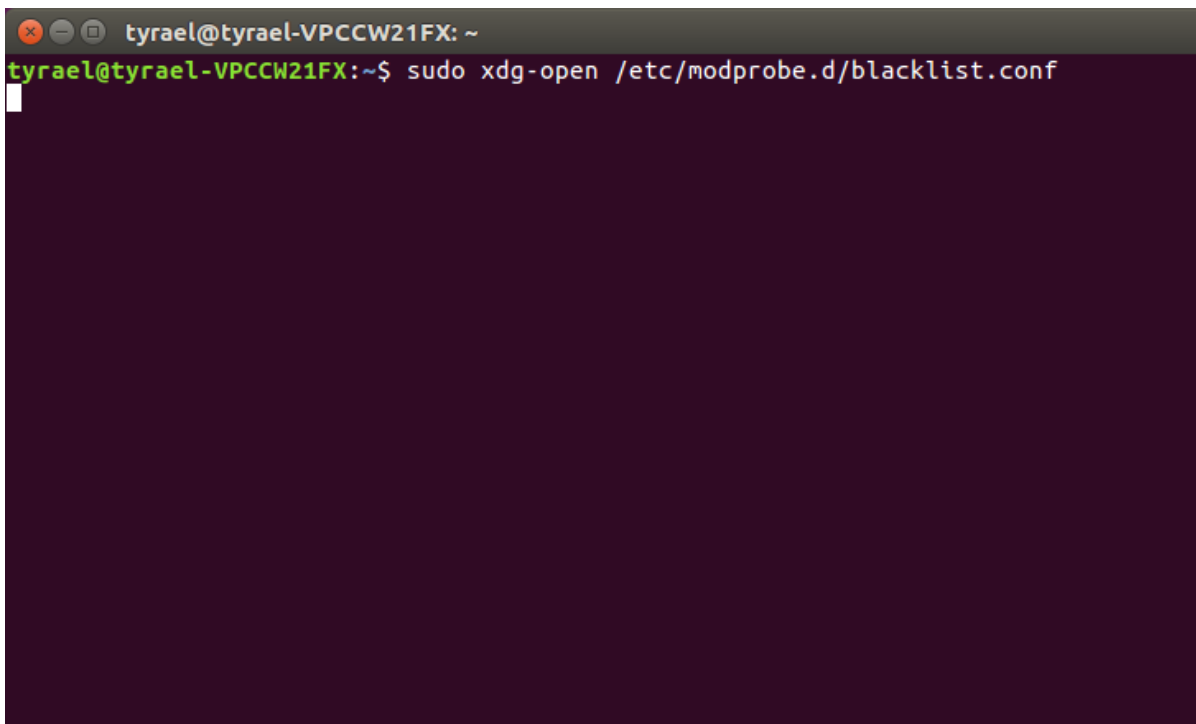
[www.nvidia.com](http://www.nvidia.com)

در پایان ستاپ از شما سوال خواهد کرد که آیا میخواهید تنظیمات در فایل X-Configuration ذخیره شده و ایدیت شود ، گزینه **yes** را انتخاب کرده و منتظر بمانید تا کامل شود **Ok** کنید و سپس عبارات زیر را در **TTY1** وارد کنید:

```
sudo service lightdm start
```

سیستم خود را ریستارت کنید.

توجه داشته باشید که اگر ستاپ از شما درخواست کرد تا **Nouveau** را غیر فعال کنید مراحل زیر را انجام دهید:



```
tyrael@tyrael-VPCCW21FX: ~  
tyrael@tyrael-VPCCW21FX:~$ sudo xdg-open /etc/modprobe.d/blacklist.conf
```

عبارت blacklist nouveau را در پایان فایل اضافه کرده و گزینه سیو را بزنید

```
Open [icon] Save

# snd_intel8x0m can interfere with snd_intel8x0, doesn't seem to support much
# hardware on its own (Ubuntu bug #2011, #6810)
blacklist snd_intel8x0m

# Conflicts with dvb driver (which is better for handling this device)
blacklist snd_aw2

# causes failure to suspend on HP compaq nc6000 (Ubuntu: #10306)
blacklist i2c_l801

# replaced by p54pci
blacklist prism54

# replaced by b43 and ssb.
blacklist bcm43xx

# most apps now use garmin usb driver directly (Ubuntu: #114565)
blacklist garmin_gps

# replaced by asus-laptop (Ubuntu: #184721)
blacklist asus_acpi

# low-quality, just noise when being used for sound playback, causes
# hangs at desktop session start (Ubuntu: #246969)
blacklist snd_pcsp

# ugly and loud noise, getting on everyone's nerves; this should be done by a
# nice pulseaudio bing (Ubuntu: #77010)
blacklist pcspr

# EDAC driver for amd76x clashes with the agp driver preventing the aperture
# from being initialised (Ubuntu: #297750). Blacklist so that the driver
# continues to build and is installable for the few cases where its
# really needed.
blacklist amd76x_edac

#Disable Nouveau
blacklist nouveau
blacklist amd76x_edac

Plain Text Tab Width: 8 Ln 58, Col 18 INS
```

دستور زیر را در ترمینال وارد کنید:

**sudo update-initramfs -u**

## نصب پلاگین پخش MP3 و دیگر امکانات نظیر پخش ویدیو:

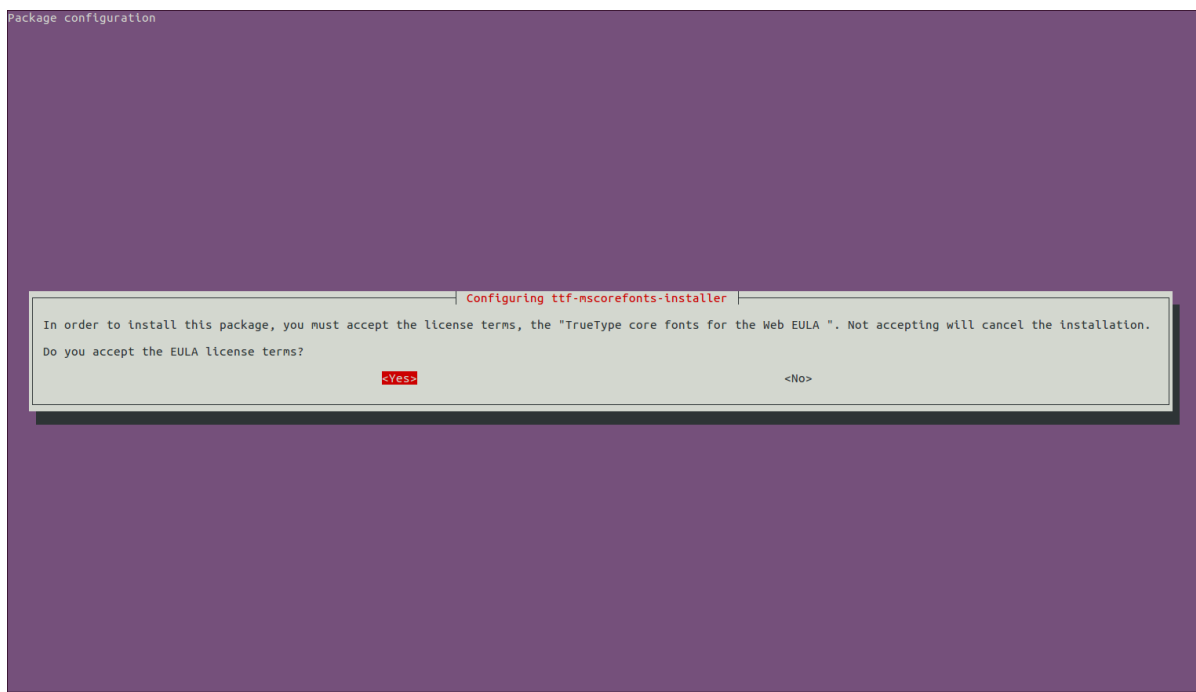
۱. دستور زیر را در ترمینال وارد کنید:

```
tyrael@tyrael-VPCCW21FX: ~  
tyrael@tyrael-VPCCW21FX:~$ sudo apt-get install ubuntu-restricted-extras
```

۲. با استفاده از کلید Tab گزینه OK را انتخاب کرده و Enter را فشار دهید

```
Package configuration  
Configuring ttf-mscorefonts-installer  
TrueType core fonts for the Web EULA  
END-USER LICENSE AGREEMENT FOR MICROSOFT SOFTWARE  
IMPORTANT-READ CAREFULLY: This Microsoft End-User License Agreement ("EULA") is a legal agreement between you (either an individual or a single entity) and Microsoft Corporation for the Microsoft software accompanying this EULA, which includes computer software and may include associated media, printed materials, and "on-line" or electronic documentation ("SOFTWARE PRODUCT" or "SOFTWARE"). By exercising your rights to make and use copies of the SOFTWARE PRODUCT, you agree to be bound by the terms of this EULA. If you do not agree to the terms of this EULA, you may not use the SOFTWARE PRODUCT.  
SOFTWARE PRODUCT LICENSE The SOFTWARE PRODUCT is protected by copyright laws and international copyright treaties, as well as other intellectual property laws and treaties. The SOFTWARE PRODUCT is licensed, not sold.  
1. GRANT OF LICENSE. This EULA grants you the following rights:  
• Installation and Use. You may install and use an unlimited number of copies of the SOFTWARE PRODUCT.  
• Reproduction and Distribution. You may reproduce and distribute an unlimited number of copies of the SOFTWARE PRODUCT; provided that each copy shall be a true and complete copy, including all copyright and trademark notices, and shall be accompanied by a copy of this EULA. Copies of the SOFTWARE PRODUCT may not be distributed for profit either on a standalone basis or included as part of your own product.  
2. DESCRIPTION OF OTHER RIGHTS AND LIMITATIONS.  
• Limitations on Reverse Engineering, Decompilation, and Disassembly. You may not reverse engineer, decompile, or disassemble the SOFTWARE PRODUCT, except and only to the extent that such activity is expressly permitted by applicable law notwithstanding this limitation.  
• Restrictions on Alteration. You may not rename, edit or create any derivative works from the SOFTWARE PRODUCT, other than subsetting when embedding them in documents.  
• Software Transfer. You may permanently transfer all of your rights under this EULA, provided the recipient agrees to the terms of this EULA.  
• Termination. Without prejudice to any other rights, Microsoft may terminate this EULA if you fail to comply with the terms and conditions of this EULA. In such event, you must destroy all  
  
<ok>
```

۳. با استفاده از کلید Tab گزینه Yes را انتخاب کرده و کلید Enter را فشار دهید.



منتظر بمانید تا نصب کامل شود.



## استفاده از دستورات در خط فرمان

شما می‌توانید هم در محیط گرافیک لینوکس (X-windows) و هم در محیط متنی کار کنید سیستم شما در محیط گرافیکی فعال است لیکن اغلب دستورات را در محیط متنی به نام پنجره (terminal windows) تایپ و اجرا می‌کنیم. بازکردن پنجره ترمینال (ورود به محیط Consol)

K-menu → search

cal نمایش تقویم	logname برای نمایش اسم
پوسته چیست؟ یک برنامه <b>unix</b> است که دستورات وارد شده از طریق صفحه‌کلید را اجرا می‌کند. یونیکس پوسته‌های متعددی دارد مثل <b>korn</b> ، <b>bourne</b> ، <b>C</b> ، ... لینوکس از پوسته مجانی دیگری به نام <b>bash</b> استفاده می‌کند که بهترین خصوصیات در پوسته‌های <b>bourne</b> ، <b>korn</b> را دارا می‌باشد. انواع پوسته‌های موجود:	logout برای خروج از حساب کاربری
بورن - کورن - <b>Cshell</b> - <b>bash</b> - <b>tcsh</b> - <b>zsh</b> با وارد کردن نام یک پوسته می‌توانیم از یک پوسته به پوسته دیگر وارد شویم	reboot یا <b>init6</b> برای راه‌اندازی مجدد سیستم
دستور دیدن نام پوشه شما <b>echo \$SHELL</b>	shutdown یا <b>init0</b> برای خاموش کردن سیستم
اگر پوشه شما <b>bash</b> باشد ← <b>/bin/bash</b>	hostname برای نشان دادن نام کامپیوترتان
نام جدید <b>hostname</b> برای تغییر نام کامپیوترتان	uname این دستور اطلاعاتی مانند - سیستم عامل شما ( <b>linux</b> ) - نام کامپیوتر - شماره نسخه ( <b>linux</b> ) - معماری پردازنده را نشان می‌دهد.
استفاده از چند دستور هم‌زمان برای جدا کردن چند دستور که می‌خواهیم در یک خط بنویسیم از ۱۲۹ جداکننده ؛ استفاده می‌کنیم.	uname -a برای شناسایی شبکه در محیط‌های اشتراکی مانند شبکه استفاده می‌شود.

	cal ; date
	clear پاک کردن صفحه
date دستور تاریخ	پس از اینکه دستور را تایپ کردید دستور به طور کامل در تاریخچه
date -U زمان به وقت گرینویچ	پوسته فرمان ذخیره می شود برای نمایش محتویات تاریخچه پوسته
	history دستور
	تا ۵ دستور قبل را نشان می دهد history5
	پاک کردن تاریخچه history -c

دو نوع کاربرد (دو سطح دسترسی) در لینوکس داریم:

۱) مدیر سیستم (root) که مثل Administrator در محیط windows می باشد و تمام اختیارات را دارد. همچنین

توانایی انجام هرگونه تنظیمی مثل: اضافه کردن، حذف کاربر، دادن permission به یک فولدر و یا فایل خاص ....

۲) کاربران عادی

[~root@hostname]# سطر فرمان (اعلان) برای root

Hostname: نام کامپیوتری که میزبان به آن متصل شده

[~username@hostname]\$ برای کاربران عادی

-بررسی نشست ورود (به) و خروج (از) سیستم:

هنگامی که وارد سیستم linux می شوید برای سیستم دارای یک هویت خاص هستید این هویت شامل نام کاربری شما، نام گروه

کاربری شما، شماره گروه شما و اطلاعات زمان ورود به سیستم مدت حضور، مدت بیکاری و محل ورود شما به سیستم را

نگهداری می کند.

برای به دست آوردن اطلاعات راجع به هویت کاری خودتان

id دستور

مثال:

Uid = 500 (Alan)

شماره کاربری

gid = 500 (Alan)

نام کاربر شماره گروه

groups = 500 (Alan)

عضو گروه

arch برای نمایش معماری سیستم می‌باشد.

دستورات کمک (info , help , whatis , man)

man مخفف manua نام دستور --help مخفف man

info دستور info مخفف information

man دستور

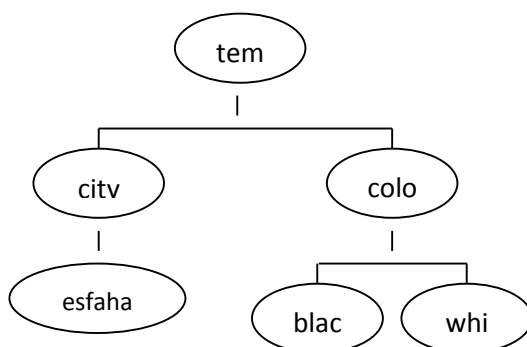
whatis نام دستور (اطلاعات مختصری در مورد دستور می‌دهد)

passwd عوض کردن پسورد

useradd جدید user اضافه کردن

userdel حذف کردن کاربر

کار کردن با فایل‌ها و فهرست‌ها و سطوح دسترسی اساسی‌ترین واحد فایل سیستم نام دارد. در unix حتی با درایوها DVD/rom یا CD/rw هم مانند فایل رفتار می‌شود.



برای ساختن پوشه ← نام پوشه فاصله mkdir

برای ایجاد ساختار درختی بالا که با دو سوئیچ p و m به کار می‌رود سوئیچ‌ها در linux با دش (-) مشخص می‌شوند.

مثال: این فرمان سبب ایجاد پوشه Esfahan داخل city می‌شود (ساخت پوشه‌های متداخل)

mkdir -p city/Esfahan

mkdir black white → این دستور باعث ایجاد پوشه‌ها همزاد می‌شود:

برای اعمال سطح دسترسی در linux

Permission یا همان مجوز در محیط‌های شبکه‌ای یا چندکاربره به وضعیتی گفته می‌شود که کاربر می‌تواند از طریق

account خاص خود به یک منبع خاصی دسترسی داشته باشد. این مجوزها توسط مدیر سیستم یا صاحب فایل اعطا می‌شود.

سه سطح دسترسی در linux داریم:

- ۱) read → r تنها اجازه خواندن
- ۲) write → w اجازه نوشتن و تغییر در فایل
- ۳) execute → x اجازه اجرای فایل

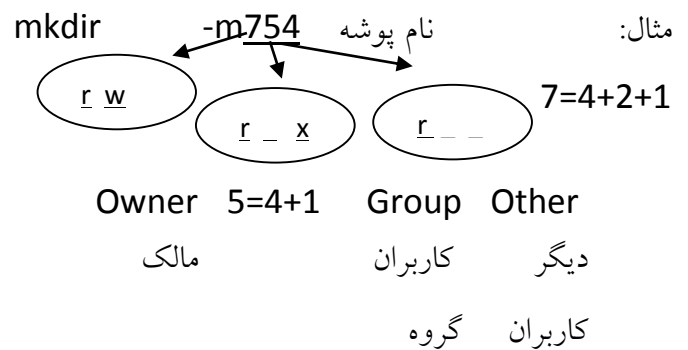
هر سطح دسترسی در linux را می‌توانید با یک عدد نشان دهید.

r → 4      w → 2      x → 1

در linux ، ۳ کلاس داریم برای کاربران که عبارت‌اند از :

owner, group , other

که برای هر کدام می‌توان هر یک از مجوزهای read ، write ، execute را تنظیم کرد.



فرمان **chmod** به معنی **change mode** برای تغییر دادن مجوزها می‌باشد که با علامت - مجوز را می‌گیریم و با علامت + می‌دهیم و با علامت = توأم دادن و گرفتن می‌دهیم.

**chmod [who][+/-/=][permission]** نام فایل

تعیین‌کننده کاربری که می‌خواهیم مجوز را به او بدهیم یا از او بگیریم که باید یکی از حروف

u= برای کاربر مال , g= برای کاربر گروه , o= برای دیگر کاربران

**chmod u+x (file1)**

**chmod ug=1 (file1)**

**chmod a=rx file1**

**chmod 440**

**chmod 347**

a = اگر بخواهیم برای تمامی کاربران به صورت یکجا مجوزها را تغییر دهیم به جای **who** از **a** استفاده می‌کنیم. می‌توانیم

حروف **ugo** را به صورت توأم هم به کاربریم.

دستور ساختن فایل

cat > اسم فایل |

تایپ محتوا  
Ctrl+z ذخیره

حذف فایل با دستور `rm -[fri]` اسم فایل

-f: فایل‌هایی که برای آن‌ها اجازه `w` ندارد را بدون گرفتن تایید (`force`) پاک می‌کند.

-i: برای پاک کردن فایل‌ها از کاربر تایید می‌گیرد.

-r: تمام فایل‌ها و فهرست‌ها یک دایرکتوری به صورت `recursive` پاک می‌کند.

حذف فهرست‌ها: `rmdir` نام فهرست

Rename directory

در صورتی که از سوئیچ `-p` استفاده کنید اگر دایرکتوری خالی هم نباشد آن را پاک می‌کند.

دستور `mv` برای انتقال و تغییر نام فایل‌ها به کار می‌رود.

`mv old new`

تغییر نام دایرکتوری از `old` به `new`

در صورتی که نوع مبدأ و مقصد فرمان `mv` یکسان باشد (یعنی هر دو فایل باشند یا هر دو پوشه باشند) به شرط یکسان بودن مسیر موردنظر در فرمان، کاربرد تغییر نام برقرار می‌شود.

اما اگر مبدأ فایل و مقصد پوشه باشد و یا مسیر به کاررفته در پارامترهای فرمان متفاوت باشد کاربرد آن انتقال است.

دستور `od` برای مشاهده محتویات فایل‌ها به کار می‌رود اما به کارگیری با سوئیچ‌های

`-c` -بیت‌های محتویات فایل را به صورت حروف اسکی نشان می‌دهد.

`-d` -کلمات محتویات فایل را در مبنای 0 نشان می‌دهد.

`-o` -کلمات محتویات فایل را در مبنای هشت (اکتان) نشان می‌دهد.

`X` -کلمات محتویات فایل را در مبنای 16 نشان می‌دهد.

یافتن فایل‌ها `find`

می‌توان موقعیت فایل‌های خاصی را که نام و مشخصات دیگری از آن‌ها را می‌دانیم در صورت عدم ذکر مسیر جستجو، جستجو در کل حافظه سیستم انجام می‌گیرد.

پارامتر `type` برای تعیین نوع فایل یا پوشه‌ای که به دنبال آن می‌گردیم

`find /t/ -name black -type d` فایل ویژه بلاکی `b`:

`find -size 10240 c` فایل پایپ `p`:

f: فایل معمولی

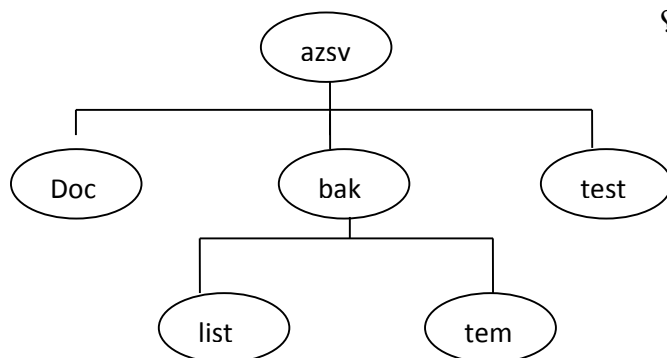
a: فایل لینک find /t/ -size 10

d: پوشه

### دستور کار لینوکس

✓ به کاربر root سوئیچ کنید و سپس با استفاده از دستورات ساختن پوشه (پوشه‌های همزاد و متداخل) ساختار

درختی زیر را ایجاد کنید؟



✓ داخل پوشه doc1 یک فایل متنی ایجاد کنید و آن را در پوشه test کپی کنید؟

✓ روی پوشه list یک permission 754 اعمال کنید (سپس دستورات زیر را اعمال کنید چه اتفاقی می افتد؟)

Chmod 643

Chmod a+=x

✓ تحقیق کنید چگونه در لینوکس یک برنامه نوشته شده با زبان C++ را اجرا می کنید؟

← در ترمینال تایپ می کنیم:

G++ -c اسم فایل

G++ -o اسم فایل اسم دیگر

./اسم

در اوبونتو ۱۱ C++ اجرای برنامه

## فصل چهارم: برنامه نویسی bash script

در این بخش می خواهیم شما را با تعدادی از دستورات این زبان آشنا نماییم.

اسکرپت سنتی سلام دنیا (Hello World)

```
#!/ bin/ bash
```

این اسکریپت فقط از دو خط تشکیل شده است:

اولین خط به سیستم نشان می‌دهد که از چه برنامه‌ای برای اجرای فایل استفاده کند.

دومین خط، تنها عملیاتی است که توسط این اسکریپت انجام می‌شود، که عبارت Hello World را در محیط ترمینال چاپ می‌کند.

اگر با خطاری شبیه به `./hello.sh: command not found` روبرو شدید، احتمالاً خط اول `((#!/bin/bash))` اشتباه است. دستور `which bash` را اجرا کنید تا مقدار درست این آدرس را ببینید.

### اسکریپت پشتیبان‌گیری خیلی ساده

```
#!/bin/bash
tar -czf /var/my -backup.tgz /home/me/
```

در این اسکریپت، به‌جای چاپ کردن یک پیغام در ترمینال، من یک فایل فشرده `tar-ball` از دایرکتوری `home` یک کاربر ایجاد کردم.

### متغیرها

می‌توانید از متغیرها مانند هم‌تاهایشان در تمام زبان‌های برنامه‌نویسی استفاده کنید. در `bash` هیچ‌گونه نوع داده‌ای وجود ندارد. یک متغیر در `bash` می‌تواند شامل یک عدد، یک کاراکتر و یا یک رشته از کاراکترها باشد.

شما نیازی به اعلان متغیرها ندارید. اختصاص یک مقدار به مرجع آن، متغیر مورد نظر را می‌سازد.

### مقایسه متغیرها در دو زبان برنامه‌نویسی

برای درک بهتر موضوع، به دو قطعه کد زیر توجه کنید. این دو قطعه کد ایجاد متغیر در دو زبان برنامه‌نویسی `bash` و `C++` را نمایش می‌دهد.

تعریف دو متغیر (یکی از نوع عددی و دیگری از نوع رشته‌ای)

در C++ :

```
int a = 1;
string sta = "bash";
```

تعریف همان دو متغیر در `bash`:

```
a=1
sta="bash"
```

مثال: استفاده از متغیرها در اسکریپت سلام دنیا!

```
#!/bin/bash
STR="Hello World!"
echo $STR
```

خط دوم یک متغیر به اسم STR می‌سازد و رشته "Hello World!"

را به آن تخصیص می‌دهد. در خط سوم مقدار متغیر با گذاشتن نشان \$ در ابتدای اسمش، فراخوانی می‌شود. لطفاً توجه کنید ( امتحان کنید!) که اگر از علامت \$ استفاده نکرده باشید خروجی برنامه متفاوت خواهد بود و احتمالاً آن چیزی که شما انتظار دارید نخواهد بود.

یک اسکریپت پشتیبان گیری خیلی ساده ( کمی بهتر از قبل)

```
#!/bin/bash
OF=/var/my-backup-$(date +%Y%m%d).tgz
tar -czf $OF /home/me/
```

این اسکریپت چیز دیگری هم معرفی می‌کند. اول از همه، شما باید با ساخت متغیرها و اختصاص دادن مقادیر به آن‌ها آشنا باشید. به این عبارت توجه کنید: «\$(date +%Y%m%d)». با اجرای اسکریپت متوجه می‌شوید که بش دستور میان پرانتز را اجرا کرده و خروجی آن را استفاده کرده است.

توجه کنید که در این اسکریپت، نام فایل خروجی در هرروز متفاوت خواهد بود. دلیلش هم استفاده از سوئیچ (+%Y%m%d) با دستور date است. می‌توانید با عوض کردن فرمت دستور date این بخش از نام فایل را عوض کنید.

مثال‌های بیشتر:

```
echo ls
echo $(ls)
```

## متغیرهای محلی

متغیرهای محلی می‌توانند با استفاده از کلمه‌ی کلیدی local ساخته شوند.

```
#!/bin/bash
HELLO=Hello
function hello {
    local HELLO=World
    echo $HELLO
}
```



```
}  
echo $HELLO  
hello  
echo $HELLO
```

این مثال برای نمایش نحوه‌ی استفاده از متغیرهای محلی باید کافی باشد.

## شروط

شروط به شما اجازه می‌دهند تصمیم بگیرید که عملی انجام شود یا نه. این تصمیم‌گیری بر اساس بررسی درستی عبارات انجام می‌شود.

## تئوری خشک:

شروط، فرم‌های زیادی دارند. پایه‌ای‌ترین فرم: **if expression then statement** است که در این فرم بخش «statement» زمانی اجرا می‌شود که عبارت «expression» از نظر منطقی درست باشد. به‌عنوان مثال « $2 < 1$ » یک عبارت غلط است. در صورتی که « $1 < 2$ » یک عبارت درست است.

شروط فرم‌های دیگری هم دارند، مثل: **if expression then statement1 else statement2**. در اینجا «statement1» زمانی اجرا می‌شود که «expression» درست باشد، در غیر این صورت «statement2» اجرا می‌شود.

یک فرم شرطی دیگر این است:

**if expression1 then statement1 else if expression2 then statement2 else statement3**  
اگر «expression1» درست بود «statement1» و اگر «expression2» صحیح بود «statement2» و در غیر این صورت «statement3» اجرا می‌شود. در این فرم فقط یک بخش **“ELSE IF ‘expression2’ THEN** ” اضافه شده است که اجازه می‌دهد «statement2» بعد از اینکه عبارت «expression2» صحیح بود، اجرا شود.

ساختار پایه‌ای (( if )) در bash به این صورت است:

if [عبارت]

```
then
کدی که در صورت درست بودن ((عبارت)) باید اجرا شود
fi
```

مثال. یک مثال ساده به صورت **if .. then**

```
#!/bin/bash
if [ "foo" = "foo" ]; then
echo expression evaluated as true
fi
```

اگر عبارت داخل گروه درست باشد کدی که بعد از **'then'** و قبل از **'fi'** باشد اجرا می‌شود. **'fi'** نمایانگر پایان بلوک شرطی کد است.

مثال. نمونه‌ای از یک شرط ابتدایی به صورت **if .. then .. else**

```
#!/bin/bash
if [ "foo" = "foo" ]; then
echo expression evaluated as true
else
echo expression evaluated as false
fi
```

مثال. شروط به همراه متغیرها

```
#!/bin/bash
T1="foo"
T2="bar"
if [ "$T1" = "$T2" ]; then
echo expression evaluated as true
else
echo expression evaluated as false
fi
```

## حلقه‌های **for** و **while** و **until**

در این بخش شما با حلقه‌های **for** و **while** و **until** آشنا خواهید شد.

حلقه‌ی **for** در اینجا کمی با دیگر زبان‌های برنامه‌نویسی متفاوت است. اساساً، این حلقه به شما اجازه می‌دهد که ((لغات)) یک‌رشته متنی را پیمایش کنید.

حلقه‌ی **while** تا زمانی که عبارت کنترلی درست باشد، قطعه کدی را اجرا می‌کند، و فقط زمانی متوقف می‌شود که عبارت کنترلی غلط شود (یا اینکه روند اجرای تکه کد صراحتاً با عبارت **break** مواجه شود).

حلقه **unit1** تقریباً با حلقه **while** برابر است، جز این که در این حلقه تا زمانی که عبارت کنترلی غلط باشد، کد اجرا می‌شود.

برای نمونه:

```
1 #!/bin/bash
2 for i in $( ls ); do
3 echo item: $i
4 done
```

در خط دوم ما **i** را به عنوان متغیر معرفی کرده ایم که می تواند مختلف حاصل از اجرای **\$( ls )** را بگیرد.

خط سوم در صورت نیاز می تواند طولانی تر باشد و یا می توان خطوط بیشتری پیش از خط **done** اضافه کرد.

'done' بیانگر این است که استفاده از متغیر **\$i** به پایان رسیده و **\$i** می تواند مقدار جدیدی دریافت کند.

این اسکریپت چندان مفید نیست، یک راه برای استفاده مفیدتر از حلقه **for** در این مثال این است که از آن برای دسترسی به فایل های خاصی استفاده شود.

## For شبیه C

Fish پیشنهاد داد که این فرم حلقه را هم اضافه کنیم. این حلقه **for** بیشتر شبیه حلقه های **for** در زبان های **C**، **perl** و... است.

```
#!/bin/bash
for i in `seq 1 10`;
do

echo $i

done
```

نمونه **while**

```
#!/bin/bash
COUNTER =0
while [ $COUNTER -lt 10 ]; do
echo The counter is $COUNTER
let COUNTER=COUNTER +1
done
```

این اسکریپت ساختار for زبان‌های مشهور C، perl و ... ) را سرمشق قرار داده است. نمونه unit1:

```
#!/bin/bash
COUNTER =20
until [ $COUNTER -lt 10 ]; do
echo COUNTER $COUNTER
let COUNTER -=1
done
```



# Android

## فصل پنجم: اندروید چیست؟

اندروید سیستم عامل متن بازی است برای گوشی‌های هوشمند و کامپیوترهای دستی و تبلت‌ها که توسط "Google" ساخته شد.

این سیستم عامل، دارای 12 میلیون کد است که 3 میلیون آن XML، ۲,۸ میلیون خط C و ۲,۱ میلیون آن جاوا است. در سال‌های آینده اندروید در میلیون‌ها سیستم همراه و موبایل مورد استفاده قرار خواهد گرفت. شاید برخی به اشتباه فکر کنند که اندروید یک پلتفرم سخت‌افزاری است. ولی اندروید تنها یک سیستم عامل است که برای موبایل‌ها ساخته شده است. هسته سیستم عامل اندروید لینوکس است و دارای یک رابط کاربری، غنی و کاربرپسند است و دارای توابعی برای مدیریت تماس‌های تلفنی است و همچنین برنامه‌های کاربردی و سودمندی برای کاربران نهایی دارد و کتابخانه‌هایی از کد برای راحتی کار برنامه نویسان و توسعه‌دهندگان نرم‌افزار دارد و از چندرسانه‌ای هم به خوبی پشتیبانی می‌کند. امروزه پلتفرم‌های بسیار زیادی برای موبایل وجود دارد که رقیب اندرویدند. سیمین، آی‌فون، ویندوز موبایل، بلک‌بری، جاوا موبایل، لینوکس موبایل، و غیره از این دسته هستند.

## مقدمه‌ای از جاوا

جاوا زبانی شبیه به C++ است (از نظر نحوی و از نظر شی گرای) ولی از C++ کوچک‌تر است چون که در جاوا از قابلیت‌های غیرضروری C که در C++ هم وجود داشت صرف نظر شده است. حال به بررسی دلایلی که باعث محبوبیت جاوا شد می‌پردازیم:

جاوا زبانی بود که در دهه 90 توسط Sun Micro system توسعه داده شد و هدف آن اجرا در محیط وب (اینترنت) بود ولی حالا جای زبان‌های خوبی مثل ++C را گرفته است و در انواع سیستم‌ها با معماری‌های مختلف کاربرد دارد. جاوا به خاطر اینکه از ++C کوچک‌تر است و قابلیت حمل بالاتری دارد (چون که مترجم جاوا کد ماشین تولید نمی‌کند بلکه کد میانی به نام بایت کد تولید می‌کند) و همچنین استفاده از آن ساده‌تر است ( زیرا جاوا قوی‌تر است و مدیریت حافظه را به صورت خودکار خودش انجام می‌دهد) و نیز بر روی ویژگی‌های مهمی مثل امنیت و مستقل از پلتفرم بودن در آن کار شده است، به همین خاطر ویژگی‌های یک‌زبان خوب را دارد و این مسائل باعث فراگیر شدن این زبان شده است. جاوا دارای مفسری است که کد میانی که اصطلاحاً به آن بایت کد (Byte Code) گفته می‌شود برای ماشین مجازی جاوا تولید می‌کند.

### طریقه نصب و پیکربندی موتور JDK و محیط توسعه اندروید ADT و Eclipse

در این جلسه با طریقه راه‌اندازی محیط به ترتیب نصب JDK و سپس ADT و آی دی ای اکیلیس آشنا می‌شوید.

برای نصب JDK باید ابتدا باید مطابق با سیستم عامل سیستم مورد نظرمان Setup را از سایت Oracle.com دانلود نماییم و سپس ستاپ را اجرا نمودیم تا عملیات نصب شروع گردد فرآیند نصب بسیار راحت بوده و فقط نیاز به دو بار کلیک کردن کلید Next می‌باشد بعد از چند ثانیه و تکمیل فرآیند دکمه‌ی finish ظاهر می‌گردد که بعد از کلیک بر روی آن مرحله اول نصب یعنی نصب JDK به پایان رسیده است به سراغ مرحله دوم نصب می‌رویم یعنی نصب محیط توسعه و یا همان آی دی ای Eclipse برای نصب اکیلیس به سایت Eclipse.org می‌رویم و با توجه به سیستم عاملی که داریم نسخه‌ی مربوطه را باید دانلود کنیم در اینجا ما سیستم عامل ویندوز ۶۴ بیتی داریم که با توجه به آن نسخه‌ی ۶۴ بیتی اکیلیس برای سیستم عامل ویندوز را دانلود نمودیم اکیلیس ستاپ و یا فایل اجرایی برای نصب ندارد و صرفاً بعد از اینکه فایل را دانلود نمودیم باید از حالت فشرده خارج نماییم و آماده برای استفاده است بهتر است مسیر خارج نمودن از حالت فشرده را در مکان فایل‌های نصب شده سایر برنامه‌های ویندوز یعنی داخل درایو Os و در پوشه ProgramsFile قرار دهیم .

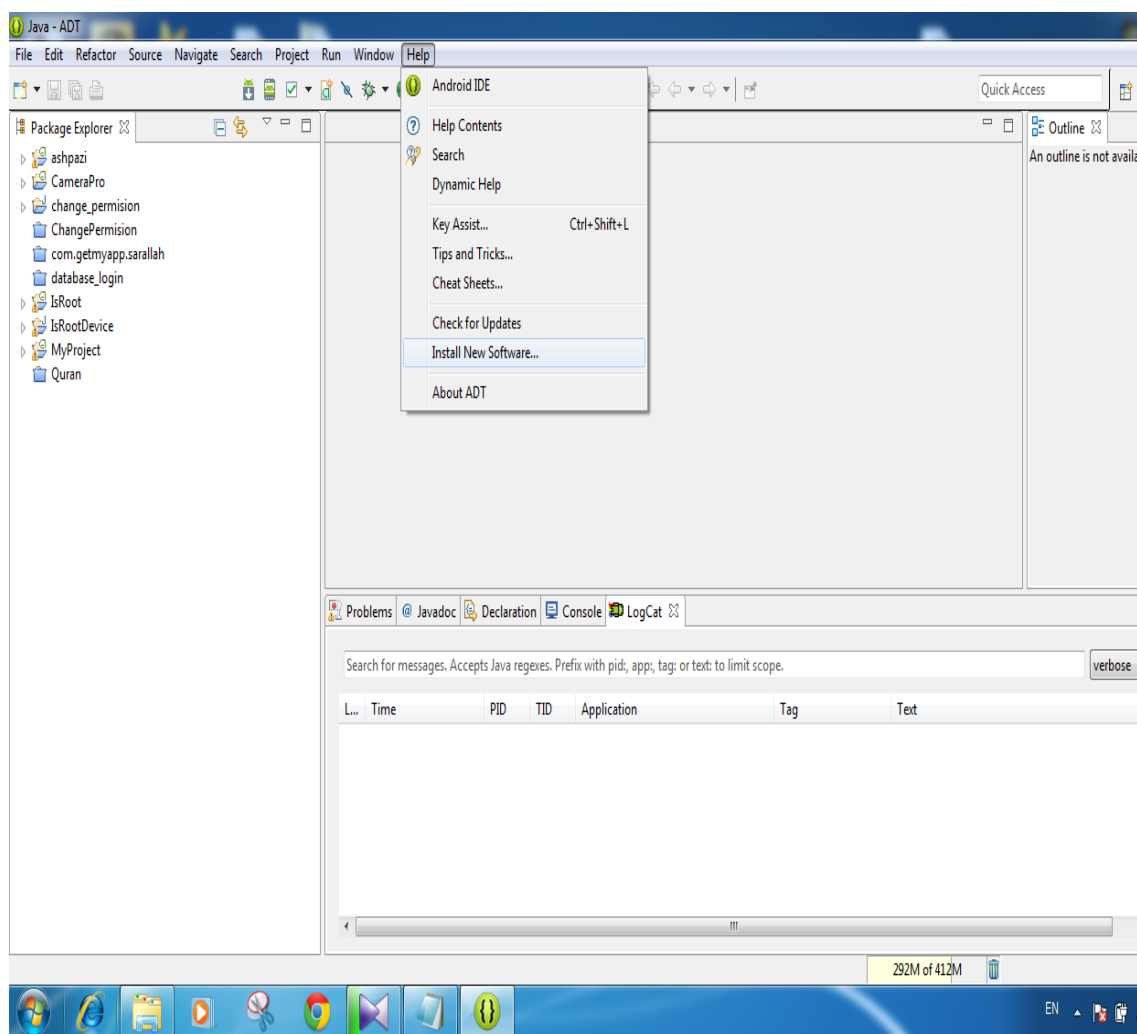
حال بر روی Eclipse.exe موجود در محتوای فولدر اکیلیس کلیک می‌نماییم و بر روی گزینه SendTo بروید و یک نسخه از آیکون را بر روی میز کارتان بفرستید .

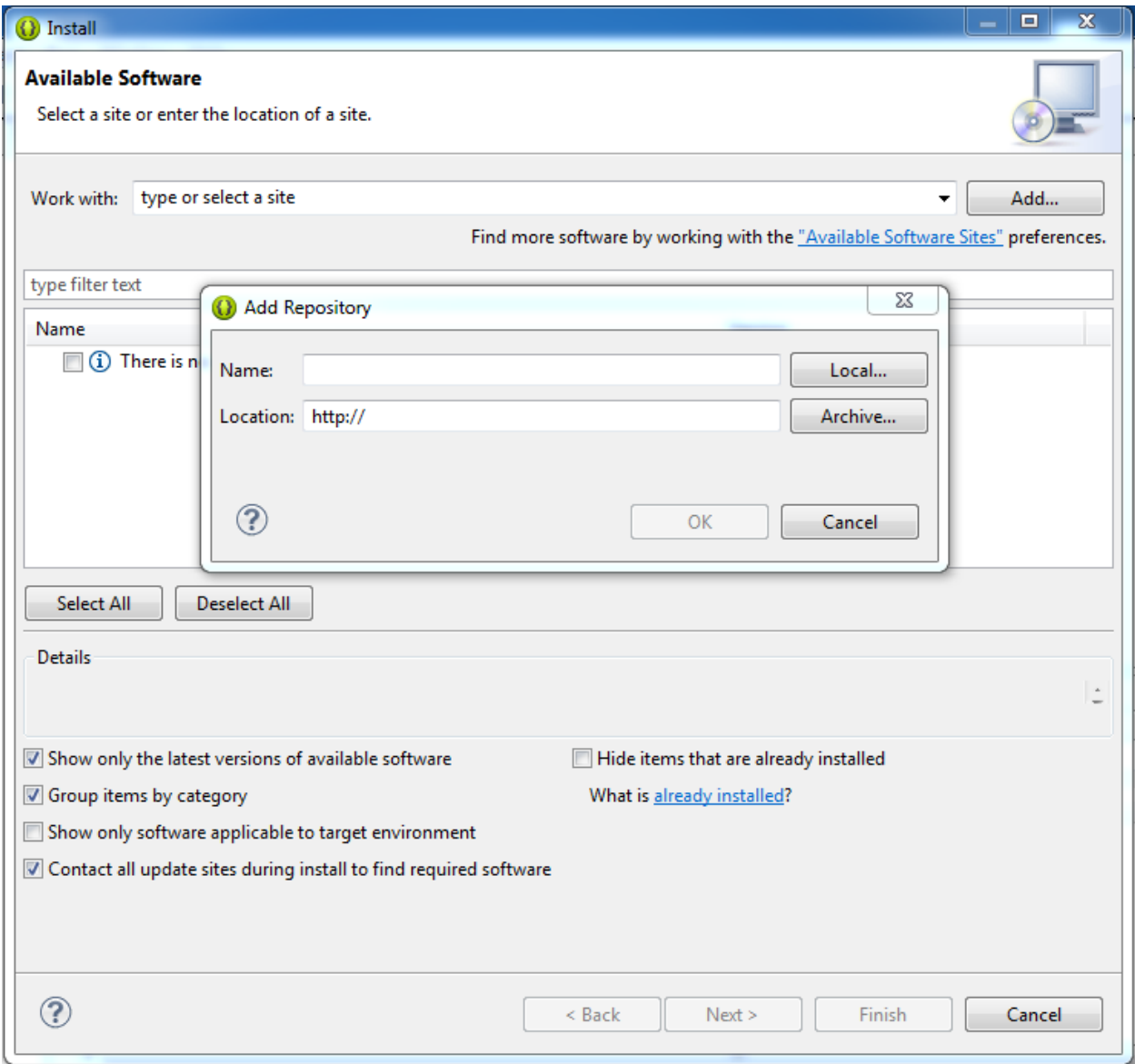
حال به سراغ نصب SDK یا کیت توسعه اندروید می‌رویم

برای به دست آوردن فایل ستاپ پیشنهاد می‌شود به خود سایت Developer.google.com بروید البته این سایت برای آی پی‌های کشور ما در حالت تحریم می‌باشد که برای استفاده از آن حتماً قبل از ورود به سایت از یک روش تعویض آی پی استفاده کنید .

سپس بنا به سیستم عامل مورد نظر ما نسخه ۶۴ بیتی ویندوزی SDK را دانلود نمودیم و پس از آن ستاپ را اجرا نمودیم بعد از اتمام نصب فایل های مربوط در پوشه AppData ذخیره می شوند .

حال ADT را نیز از آدرس [Developer.google.com](http://Developer.google.com) دانلود نمودیم اکیلیپس را اجرا می کنیم بعد از باز شدن محیط اکیلیپس به منوی Help و سپس Install New Software می رویم .

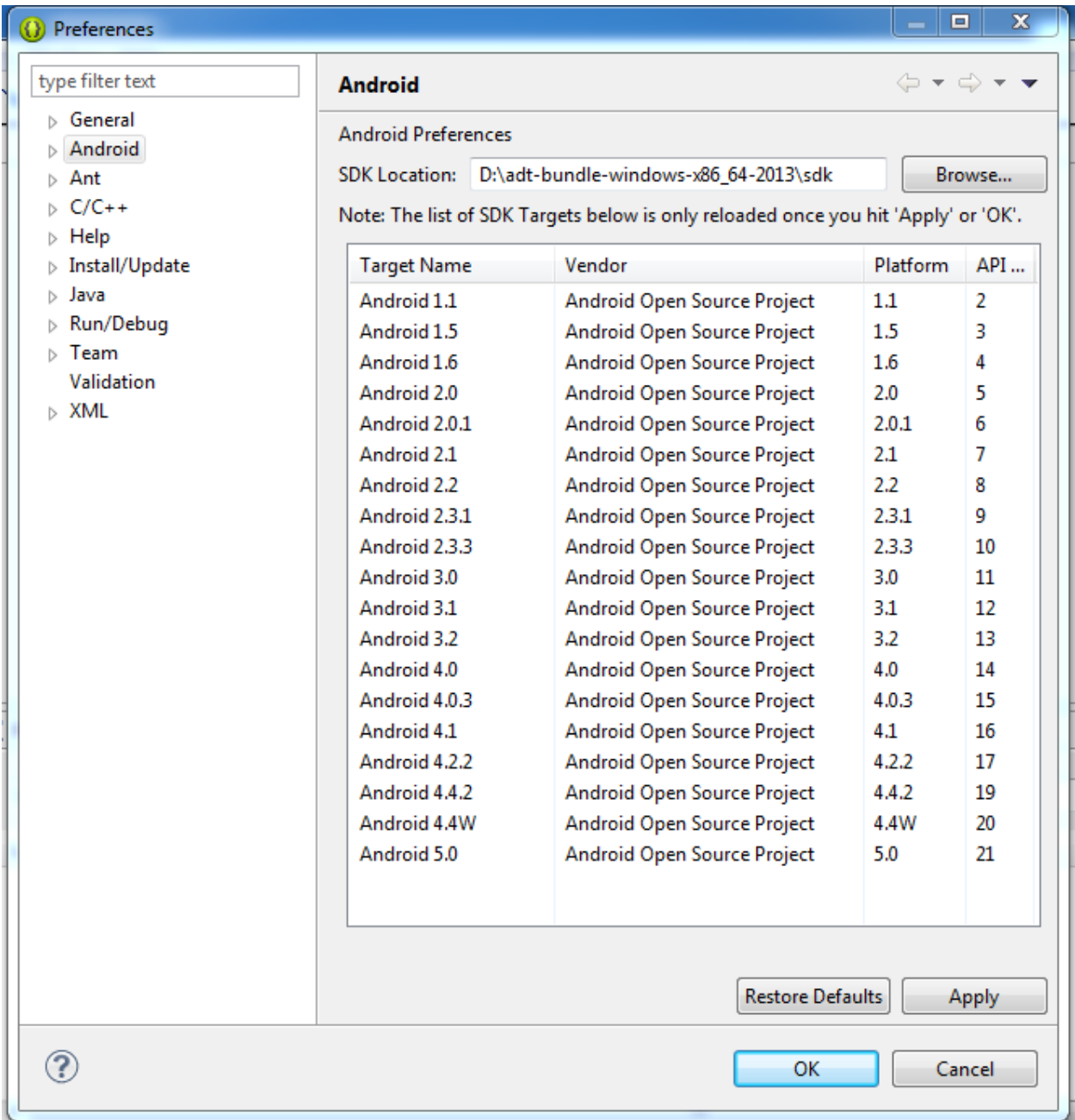




سپس دکمه‌ی **Add** را کلیک و بعدازآن در صفحه‌ی ظاهرشده دکمه‌ی **Arshive** را کلیک می‌کنیم و سپس مسیر فایل **ADT** که دانلود نمودیم را به آن می‌دهیم بعد دکمه‌ی **OK** را کلیک می‌کنیم و سپس در صفحه ظاهرشده دکمه‌ی **Next** را کلیک می‌کنیم و مراحل را ادامه می‌دهیم تا فرآیند نصب تکمیل گردد.

اکلیپس پس از اتمام نصب **ADT** پیام ریست می‌دهد که باید ای دی ای ریست شود بعد از ریست شدن ای دی ای و دوباره باز شدن اکلیپس برای مسیره‌ی **SDK** باید به منوی **Windows** و سپس **Preference** می‌رویم در صفحه‌ی ظاهرشده.



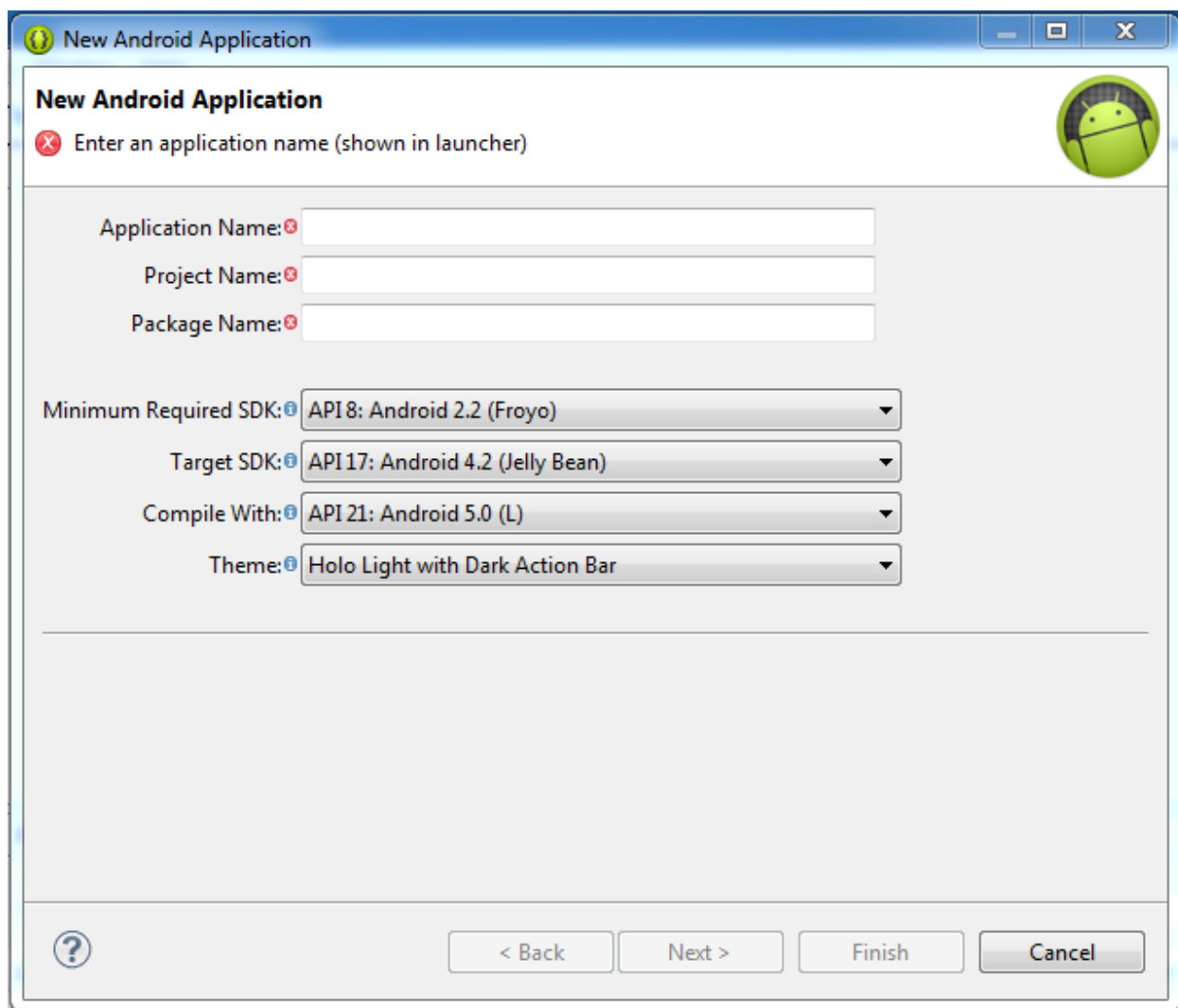


و در این صفحه در سمت چپ بر روی **Android** کلیک می‌کنیم سپس در محتویات ظاهرشده در سمت راست دکمه‌ی **Browse** را کلیک می‌کنیم و مسیر فایل‌های **SDK** را به آن می‌دهیم .

حال محیط آماده استفاده می‌باشد.

## ایجاد یک پروژه جدید

برای ایجاد یک پروژه جدید در اکتیو ایس بر روی فایل می‌رویم سپس بر روی گزینه **New** و بر روی **Android Project** کلیک می‌کنیم صفحه زیر ظاهر می‌گردد.



در قسمت **ApplicationName**: نام اپلیکیشن را تعیین می‌کنیم این نام زیر آیکون برنامه زیر آن درج می‌شود .

در قسمت **ProjectName**: نام پروژه نامی است که پروژه‌ها را می‌توان در ای دی اکتیو ایس از هم متفاوت ساخت

و در نهایت در قسمت **PackageName**: نام پکیج را تعیین می‌کنیم. نام پکیج‌ها معمولاً چندقسمتی می‌باشد که توسط دات و یا همان نقطه از هم جدا می‌شود.

در قسمت **Minimum Required SDK** حداقل سیستم‌عامل برای اجرای برنامه‌مان را تعیین می‌کنیم.

در قسمت **Target SDK** نسخه اندرویدی که قرار است برنامه با آن ایجاد شود را مشخص می‌کنیم.

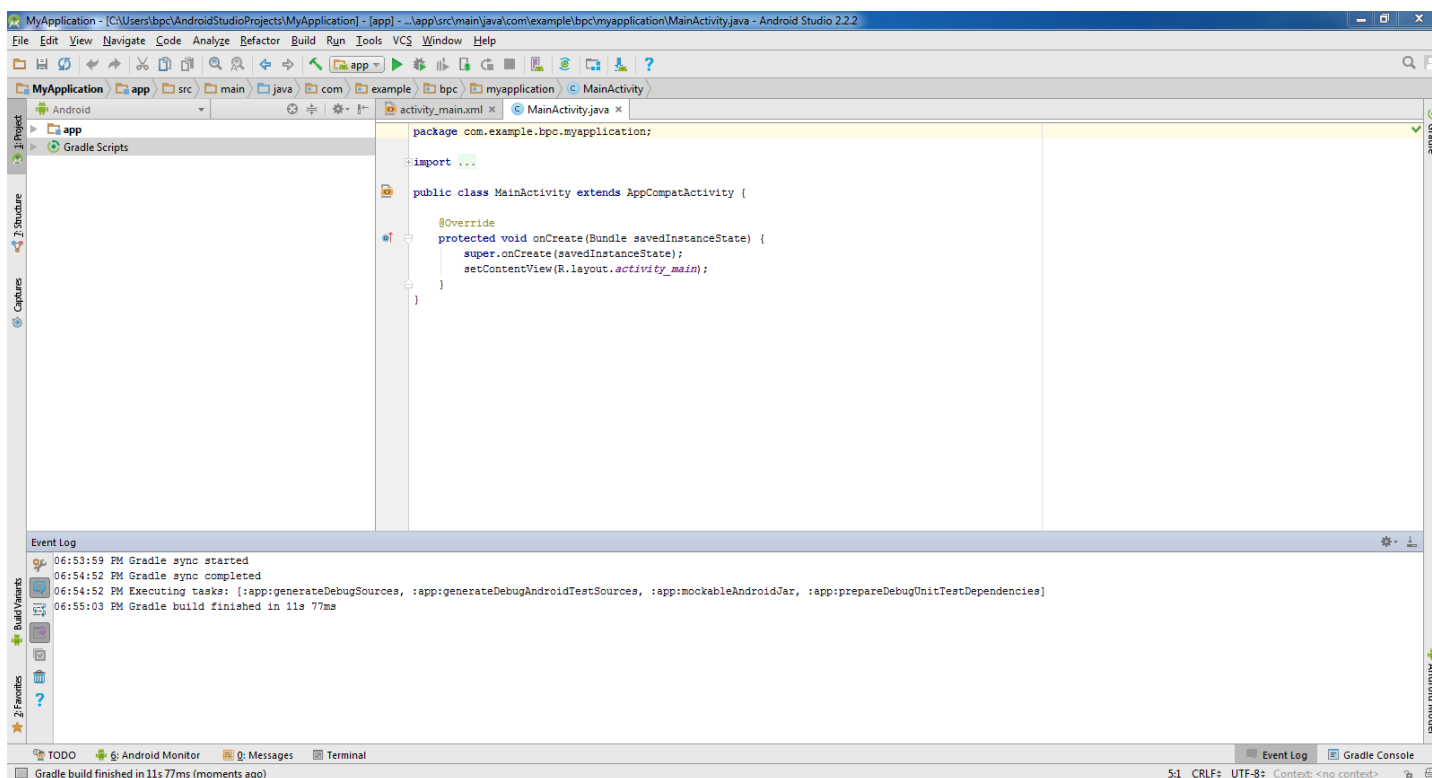
در قسمت **CompileWhit**: نسخه اندرویدی که قرار است برنامه با آن کامپایل شود را مشخص می‌کنیم.

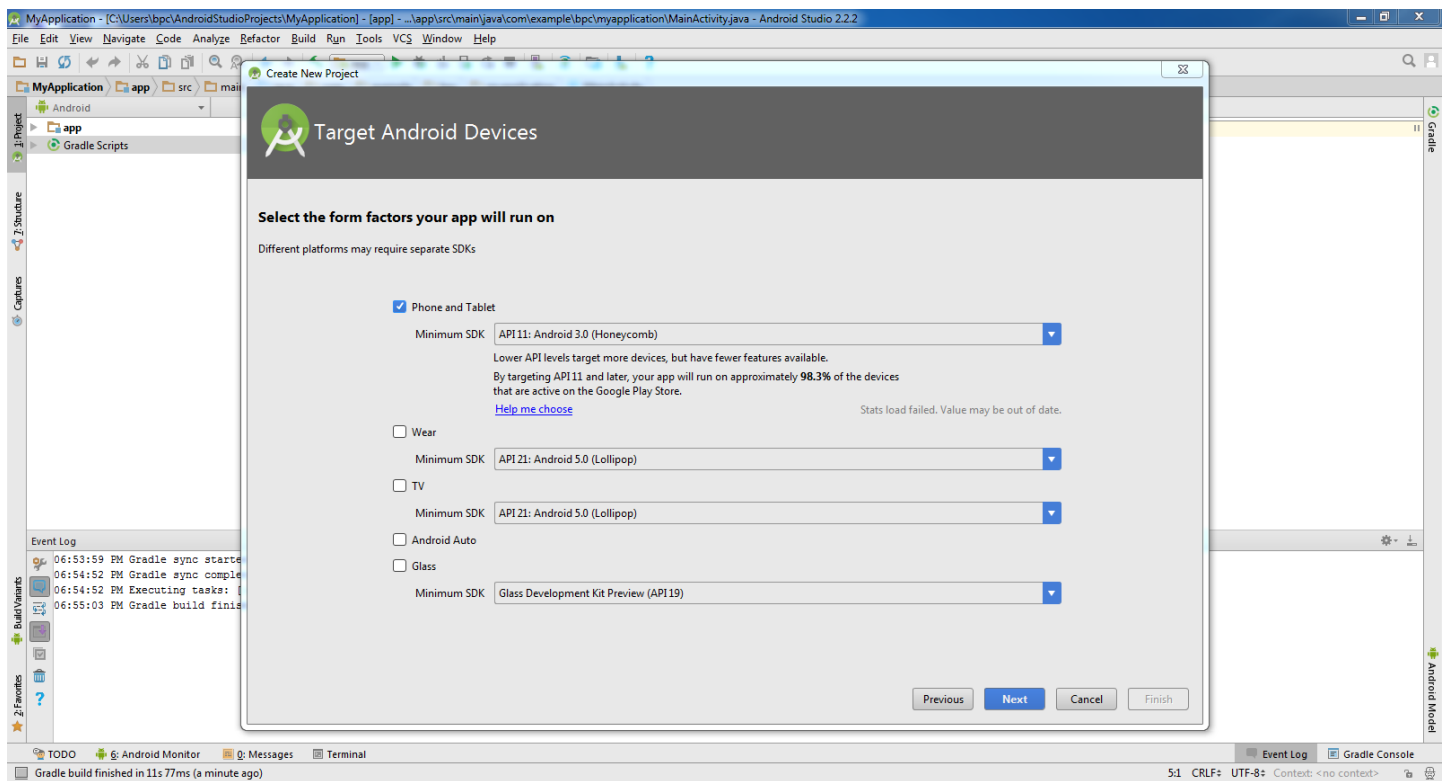
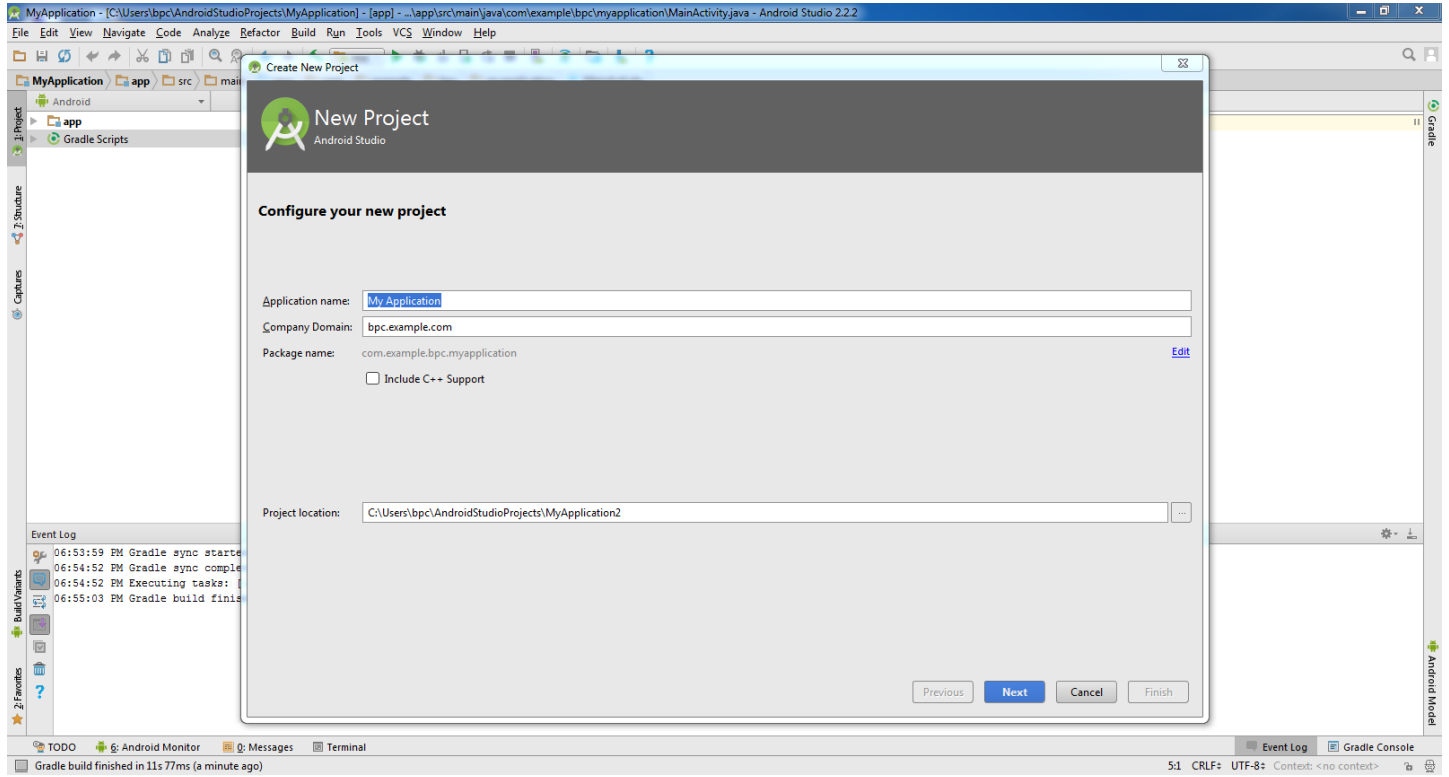
و در قسمت **Theme** تم برنامه را مشخص می‌کنیم.

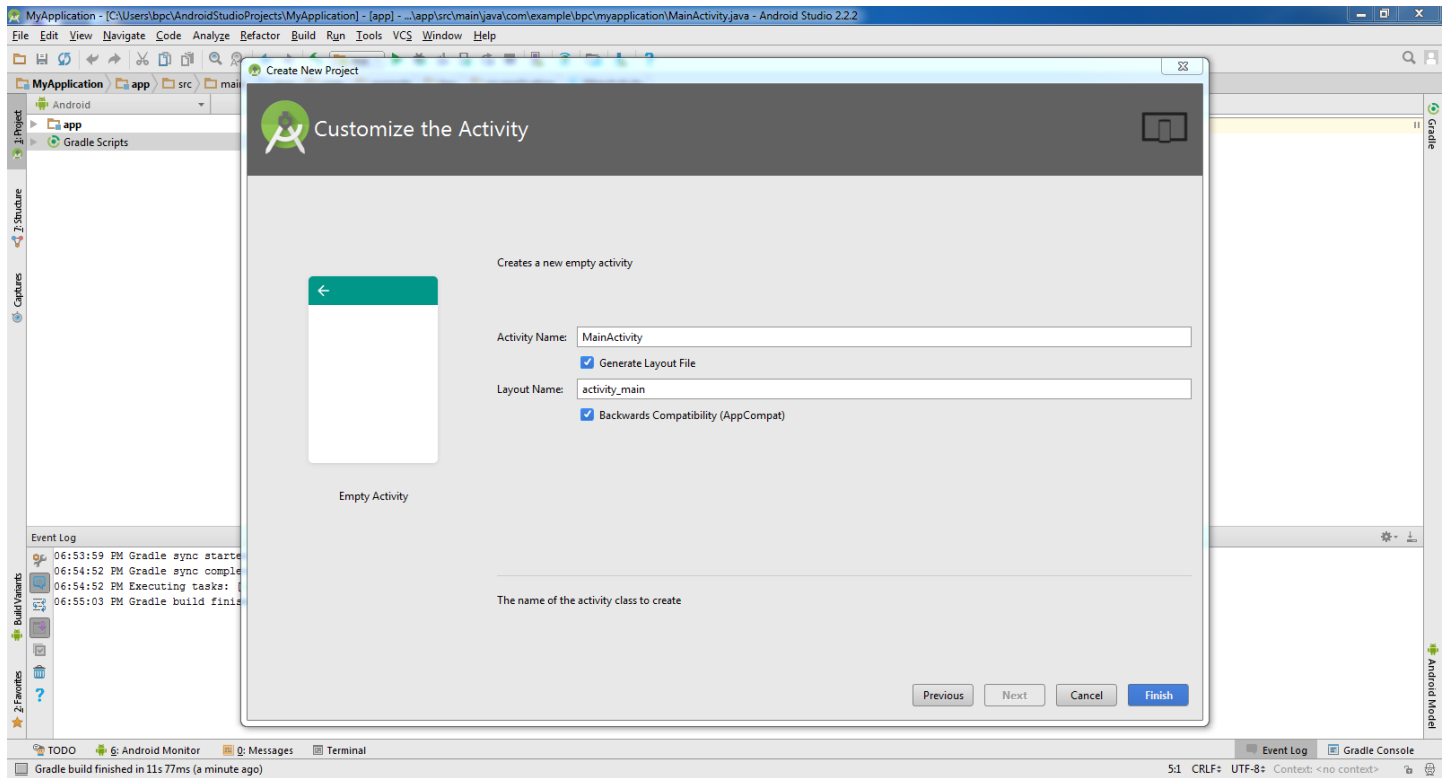
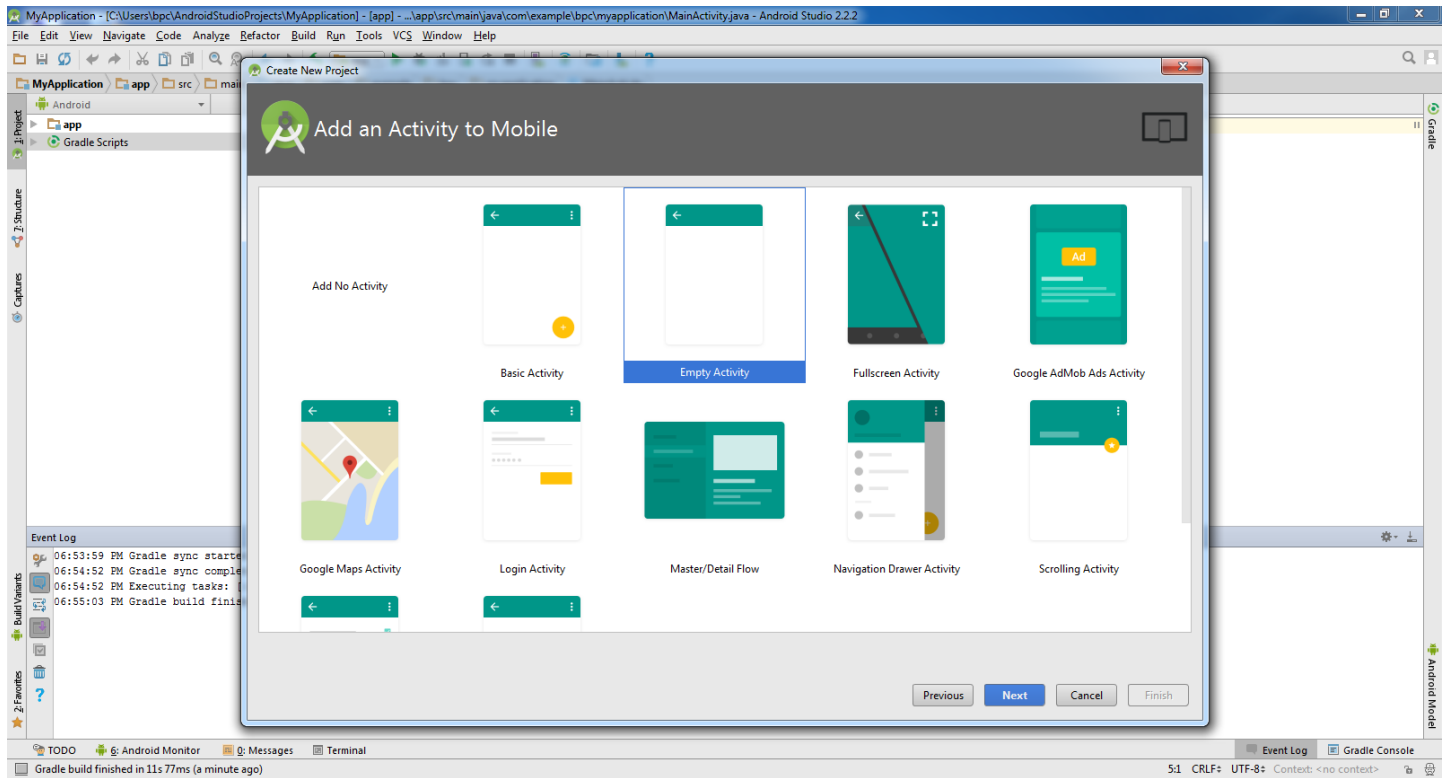
سپس در چند گام بعدی **Next** را کلیک می‌کنیم و **Finish** را کلیک می‌کنیم.

## اندروید استادیو

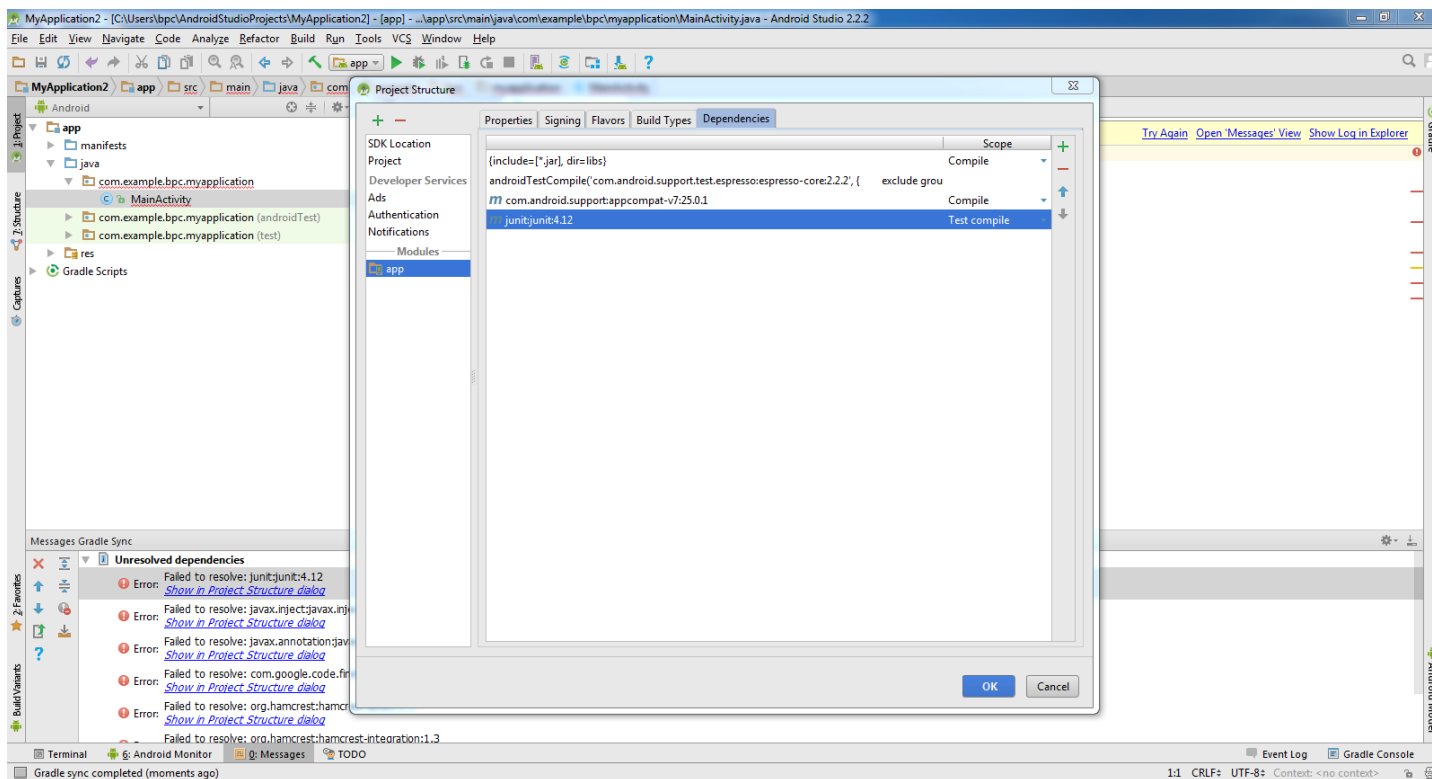
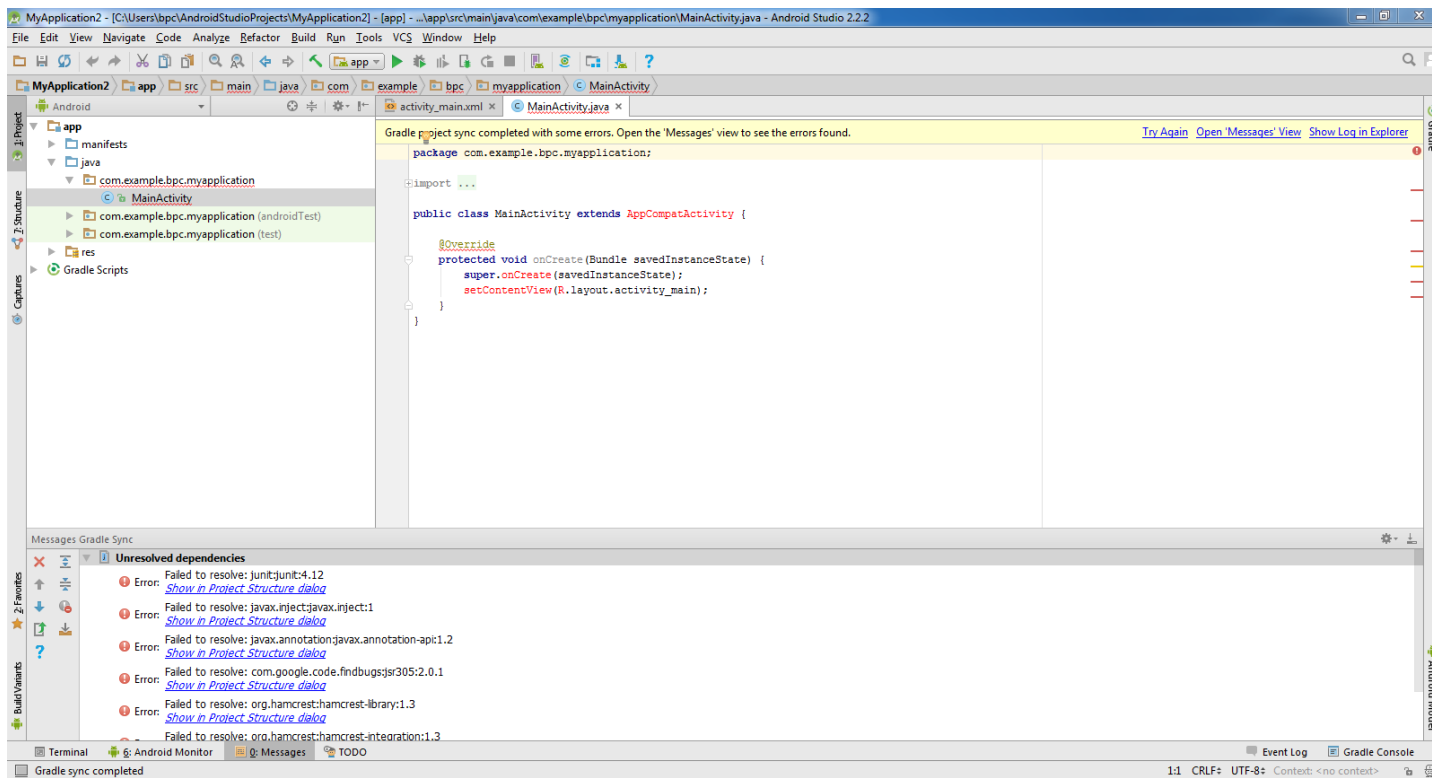
صفحه اول نرم افزار اندروید استادیو:



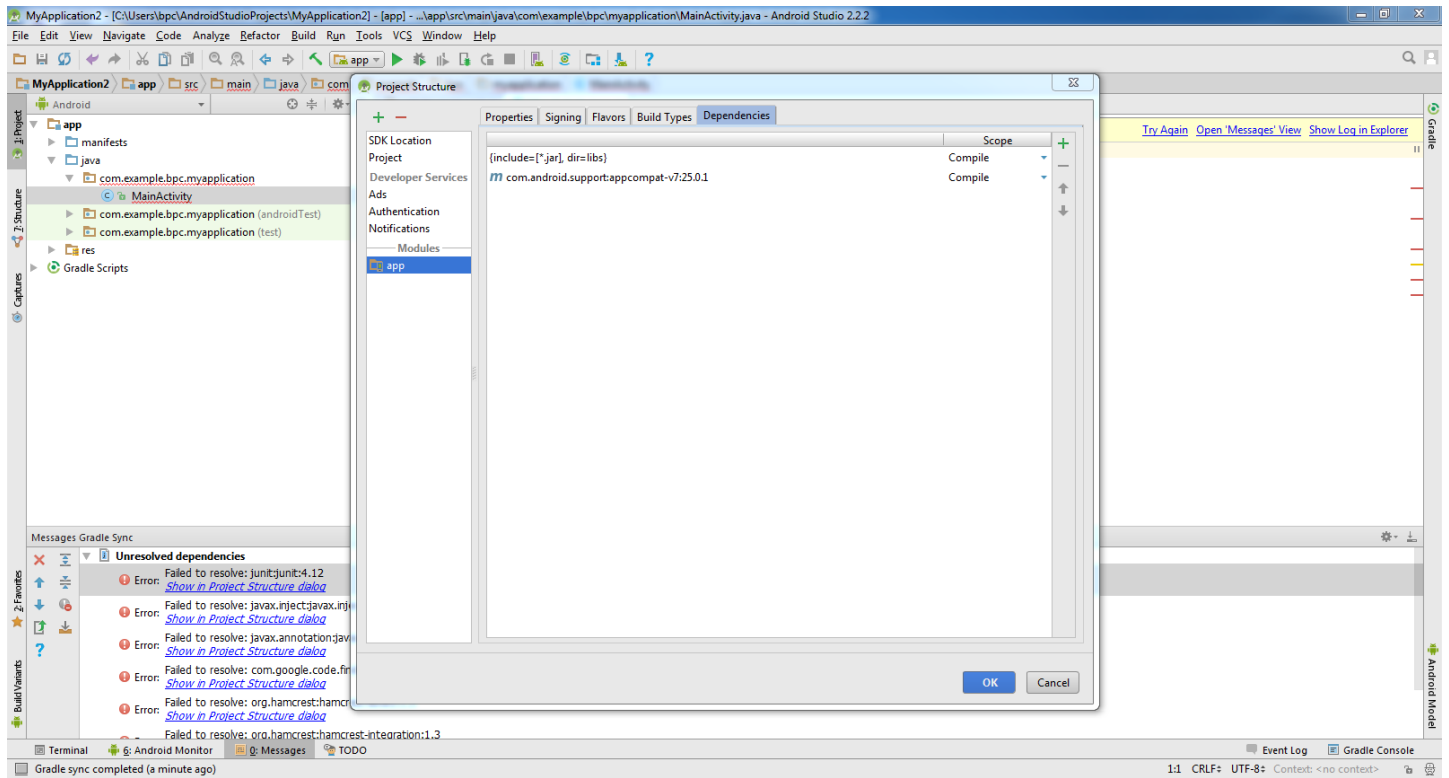




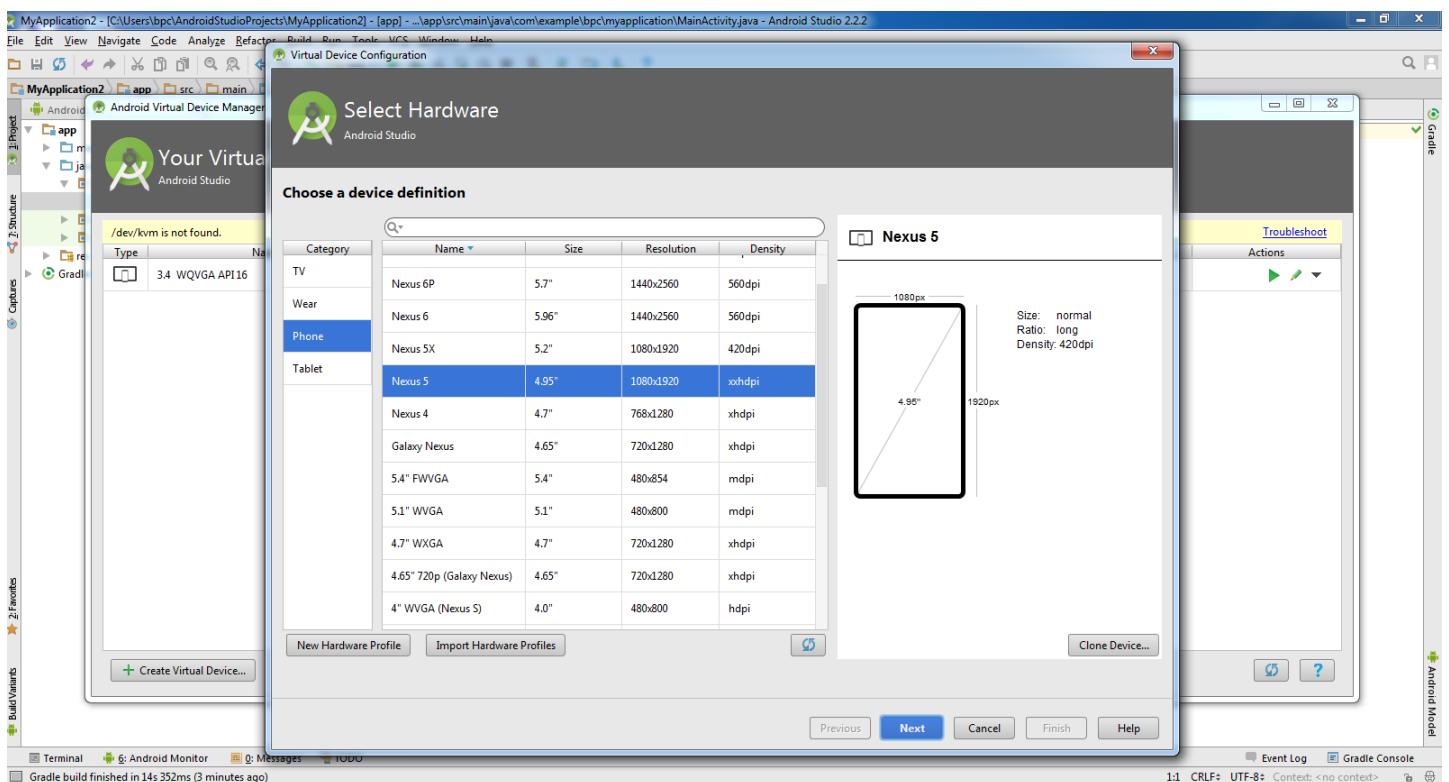
در صورت عدم دسترسی به اینترنت با ارور زیر مواجه خواهید شد برای رفع اشکال مراحل زیر را انجام دهید:

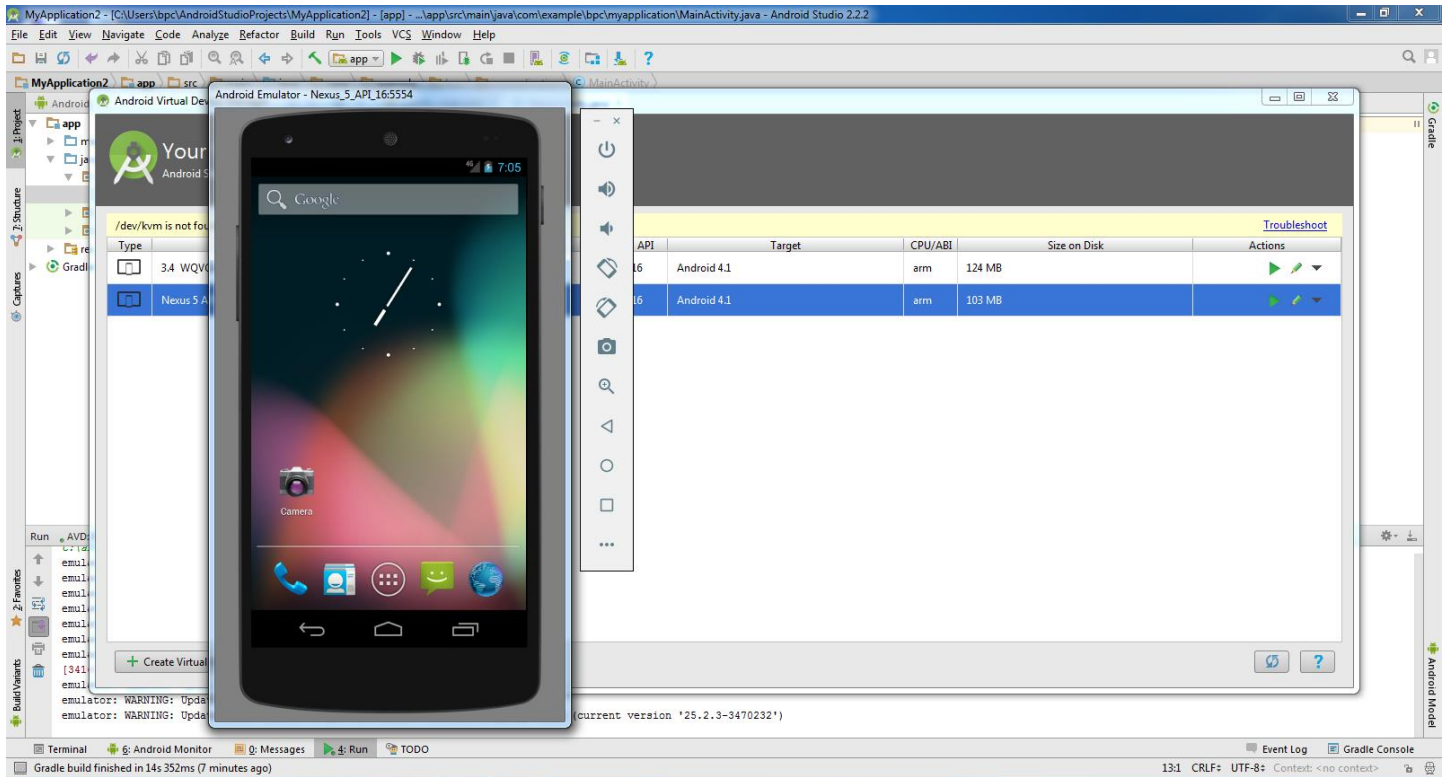
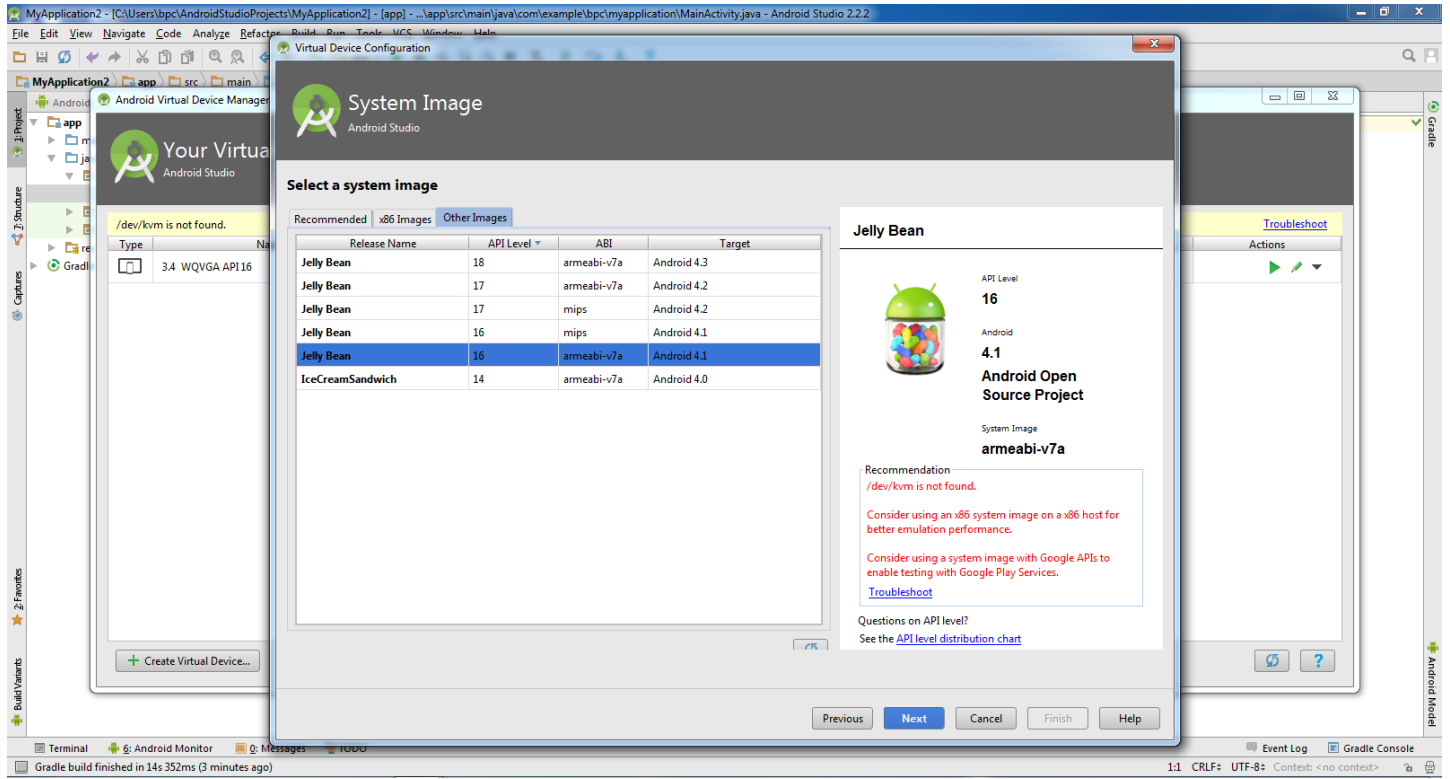


Junit و androidtestcompile را از لیست بالا حذف کنید و بر روی ok کلیک کنید.



ایجاد یک ماشین مجازی از طریق AVD:







## رفتن از یک اکتیویته به اکتیویته دیگر

مطابق ایجاد ساخت یک پروژه که در بالا گفته شد یک پروژه ایجاد می‌کنیم

سپس کد XML آن را مطابق زیر برای اکتیویته مبدا می‌نویسیم :

```
"RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android">
"xmlns:tools="http://schemas.android.com/tools
"android:layout_width="match_parent
"android:layout_height="match_parent
"android:paddingBottom="@dimen/activity_vertical_margin
"android:paddingLeft="@dimen/activity_horizontal_margin
"android:paddingRight="@dimen/activity_horizontal_margin
"android:paddingTop="@dimen/activity_vertical_margin
<"tools:context=".Activity_Aval

Button>
"android:id="@+id/borobe_avtivity2
"android:layout_width="80dp
"android:layout_height="80dp
"android:layout_centerHorizontal="true
</"android:background="@drawable/icon_list

Button>
"android:id="@+id/borobe_avtivity3
"android:layout_width="match_parent
"android:layout_height="50dp
"android:layout_below="@+id/borobe_avtivity2
"android:layout_marginTop="10dp
"android:background="#f40f40
</"android:text="@string/name_dokme2
```

```
<RelativeLayout/>
```

همان‌طور که در بالا نوشته‌شده در اکتیویته‌ی بالا دو دکمه با نام‌های `borobe_avtivity2` و `borobe_avtivity3`

داریم که قرار است با فشردن هرکدام از دکمه‌ها به اکتیویته‌ی های ۲ و ۳ برویم

به این منظور به سراغ کدهای جاوا در اکتیویته‌ی اول یا اکتیویته‌ی مقصد می‌رویم:

```
import android.os.Bundle;
import android.app.Activity;
import android.content.Intent;
import android.util.Log;
import android.view.Menu;
import android.view.View;
import android.widget.Button;

public class Activity_Aval extends Activity{

    Button btn1,btn2;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_activity__aval);

        btn1 = (Button) findViewById(R.id.borobe_avtivity2);
        btn2 = (Button) findViewById(R.id.borobe_avtivity3);
```

```

btn1.setOnClickListener(new View.OnClickListener() {

    @Override
    public void onClick(View arg0) {
        //TODO Auto-generated method stub

        Intent i = new Intent();
        i.setClass(Activity_Aval.this, Activity_2vom.class);
        startActivity(i);
        Log.d("test1", "roydad dokme 1 be dorosty anjam shod!!!");

    }
});

btn2.setOnClickListener(new View.OnClickListener() {

    @Override
    public void onClick(View arg0) {
        //TODO Auto-generated method stub

        Intent borobe_activity3 = new Intent();

        borobe_activity3.setClass(Activity_Aval.this, Activity3vom.class);
        startActivity(borobe_activity3);

    }
});
}
}

```

در اصل کد رفتن به اکتیویته مقصد تکه کد زیر است که در رویداد کلیک دکمه‌ها نوشتیم

```
Intent i = new Intent();  
i.setClass(Activity_Aval.this, Activity_2vom.class);  
startActivity(i);
```

حال تعداد دکمه‌ها کم بود و در متد **OnCreate** دکمه‌ها را حساس به کلیک کردیم در مرحله‌ی بعد می‌خواهیم تعدادی بیشتری دکمه را حساس به کلیک کنیم .

رفتن از یک اکتیویته به چند اکتیویته با متد **startActivity()** و خارج از رویداد **onClick()**

در این حالت از یک اکتیویته که دارای چندین دکمه است می‌خواهیم با استفاده از هر یک دکمه‌ها به اکتیویته‌های بعدی برویم اکتیویته اول شامل دکمه‌ها را ایجاد می‌کنیم به این منظور ابتدا به بخش **XML** رفته و یک لایه ایجاد می‌کنیم و کدهای زیر را در آن می‌نویسیم:

```
"android:paddingLeft="@dimen/activity_horizontal_margin  
"android:paddingRight="@dimen/activity_horizontal_margin  
"android:paddingTop="@dimen/activity_vertical_margin  
"android:background="@drawable/main  
< "tools:context=".MainActivity
```

Button>

```
"android:id="@+id/btn1  
"android:layout_width="80dp  
"android:layout_height="80dp  
"android:background="@drawable/information_icon  
"android:layout_centerHorizontal="true
```

</

Button>

"android:id="@+id/btn2

"android:layout\_width="80dp

"android:layout\_height="80dp

"android:background="@drawable/list\_icon

"android:layout\_centerHorizontal="true

"android:layout\_below="@+id/btn1

"android:layout\_marginTop="10dp

</

Button>

"android:id="@+id/btn3

"android:layout\_width="80dp

"android:layout\_height="80dp

"android:background="@drawable/question\_icon

"android:layout\_centerHorizontal="true

"android:layout\_below="@+id/btn2

"android:layout\_marginTop="10dp

</

Button>

"android:id="@+id/btn4

"android:layout\_width="80dp

"android:layout\_height="80dp

"android:background="@drawable/share\_icon

"android:layout\_centerHorizontal="true

"android:layout\_below="@+id/btn3

"android:layout\_marginTop="10dp

</

RelativeLayout/>

سپس به بخش کدهای جاوا می‌رویم و دکمه‌ها را مطابق کدهای زیر حساس به کلیک می‌کنیم :

```
package com.sss.callotheractivitybyswichmethod;

import java.io.File;

import android.net.Uri;

import android.os.Bundle;

import android.app.Activity;

import android.app.AlertDialog;

import android.content.DialogInterface;

import android.content.Intent;

import android.content.pm.ApplicationInfo;

import android.content.pm.PackageManager;

import android.util.Log;

import android.view.Menu;

import android.view.View;

import android.view.View.OnClickListener;

import android.widget.Button;

public class MainActivity extends Activity{

    Button btn1,btn2,btn3,btn4;

    @Override

    protected void onCreate(Bundle savedInstanceState){

        super.onCreate(savedInstanceState);

        setContentView(R.layout.activity_main);

        btn1 = (Button)findViewById(R.id.btn1);

        btn2 = (Button)findViewById(R.id.btn2);

        btn3 = (Button)findViewById(R.id.btn3);

        btn4 = (Button)findViewById(R.id.btn4);

        btn1.setOnClickListener(new onClk());
```

```
btn2.setOnClickListener(new onClk());  
btn3.setOnClickListener(new onClk());  
btn4.setOnClickListener(new onClk());  
  
{
```

```
class onClk implements OnClickListener  
{
```

```
    @Override
```

```
    public void onClick(View arg0)
```

```
    {
```

```
        Intent i = new Intent();
```

```
        switch (arg0.getId())
```

```
        {
```

```
            case R.id.btn1:
```

```
                i.setClass(MainActivity.this, Activity_1.class);
```

```
                startActivity(i);
```

```
                break;
```

```
            case R.id.btn2:
```

```
                i.setClass(MainActivity.this, Activity_2.class);
```

```
                startActivity(i);
```

```
                break;
```

```
            case R.id.btn3:
```

```
                i.setClass(MainActivity.this, Activity_3.class);
```

```
                startActivity(i);
```

```
                break;
```

case R.id.btn4:

```
        i.setClass(MainActivity.this, Activity_4.class);  
        startActivity(i);  
        break;  
    }  
    }  
    }  
    }
```

تفاوت این روش با حساس به کلیک کردن در متد **OnCreat** () آن است که در این روش می توان دکمه های زیاد را بهتر در حساس به کلیک بودن مدیریت کرد .

### ایجاد پروژه Hello world

در ابتدا برای ساخت یک پروژه از قسمت **file** از قسمت **New** ، گزینه ی **project** را انتخاب کرده و سپس از پوشه ی **android** ، گزینه ی **Android Application Project** را انتخاب کرده و در صفحه ای که ایجاد می شود در قسمت **Application Name** ، نام اپلیکیشن مورد نیاز را نوشته که در این قسمت **Hello world** تایپ می کنیم. در قسمت **Project Name** ، نام پروژه را مشخص کرده که همان **Hello world** را تایپ می کنیم.

در قسمت بعدی نام پکیج را انتخاب می کنیم که انتخاب این نام خیلی مهم هست و بعداً با ایجاد ورژن های متعدد برای این نرم افزار کاربرد دارد.

گزینه ی بعدی **Minimum Required SDK** است که منظور کمترین شماره اندرویدی است که پروژه ما می تواند بر روی آن اجرا شود.

گزینه ی **Target SDK** یعنی شماره ای که قرار است برنامه با آن ایجاد شود.

گزینه ی **Compile With** یعنی شماره اندرویدی که قرار است برنامه با آن کامپایل شود.



بعدازآن بر روی دکمه **Next** کلیک کرده، در صفحه‌ی بعدی تیک گزینه‌ی **Create activity** را برداشته و دوباره **Next** را می‌زنیم. در صفحه بعدی می‌توانیم آیکون برنامه را تعیین کنیم و بعد دوباره دکمه‌ی **Next** را زده و در صفحه‌ی ایجادشده تیک گزینه‌ی **Create Activity** را گذاشته و بر روی دکمه‌ی **Next** کلیک کرده.

در این صفحه ۲گزینه وجود دارد که خیلی مهم و ضروری هستند.

گزینه‌ی **Activity Name** که باید در آن نام اکتیویتی خود را انتخاب کرد که مربوط به کد جاوا هست.

گزینه‌ی **Layout Name** که به فایل **xml** اختصاص داده می‌شود.

و در پایان گزینه‌ی **Finish** را می‌زنیم تا پروژه جدید ما با نام **Hello world** ایجاد شود.

حال ما می‌خواهیم یک **TextView** ایجاد کنیم.

زمانی که این پروژه ایجادشده آن را باز کرده و در قسمت **res** گزینه‌ی **layout**، فایل **activity\_main.xml** را باز می‌کنیم و در صفحه‌ی بازشده عبارت **TextView** را تایپ می‌کنیم. برای این کار ما می‌توانیم حروف **Te** را تایپ کرده و کنترل + **space** را زده تا عباراتی را که به صورت پیش فرض وجود دارند را برای ما بیاورد و ما گزینه‌ی **<TextView/>** را انتخاب می‌کنیم. برای ایجاد کردن هر خاصیت باید ابتدا کلمه‌ی **android:** را تایپ نموده و بعد مشخصه‌ای را که می‌خواهیم مشخص کنیم. ما برای هر **TextView**،

**id- width- height- background- Color- text** را مشخص کرده. در اینجا ما به صورت زیر تعریف می‌کنیم.

```
<TextView
    android:id="@+id/text_view_helloworld" // شناسه
    android:layout_width="match_parent" // عرض
    android:layout_height="30dp" // طول
    android:background="#000000" // رنگ پس‌زمینه
    android:text="@string/hello_world" // متنی که قرار است در آن نوشته شود
    android:textColor="#ffffff" // رنگ متن
/>
```

نکته‌ای که وجود دارد این است که برای آوردن هر کدام از گزینه‌ها ۱ یا ۲ حرف اول آن را تایپ کرده و سپس کنترل + **space** را زده تا کلماتی را که به صورت پیش فرض وجود دارند برای ما بیاورد، آنگاه ما کلمه‌ای را که بخواهیم انتخاب می‌کنیم.

برای مشخص کردن id همیشه باید گزینه‌ی `@+id` را انتخاب می‌کنیم، با این کار به صورت خودکار یک id ایجاد شده است. `text_view_helloworld` شناسه‌ای است که جاوا با آن، آن را شناسایی می‌کند.

برای مشخص کردن عرض می‌توانیم `match_parent` را انتخاب نماییم، یعنی کل صفحه را پر کند.

برای مشخص کردن طول می‌توانیم به آن عدد بدهیم.

ما می‌توانیم متنی را که می‌خواهیم درون `text_view` نمایش داده شود به صورت مستقیم جمله‌ای را که می‌خواهیم تایپ نماییم، اما به این صورت استاندارد نیست و حالت استاندارد آن به صورت زیر هست.

همان‌طور که قبلاً گفته شده است رشته (string) های ما در قسمت `res` در قسمت `values` نوشته می‌شوند.

در قسمت `string_xml`، رشته‌ی مربوط به این جمله را نوشته.

```
<string name="hello_world">به برنامه‌نویسی اندروید خوش آمدید</string>
```

در ادامه به صورت کلی تمام رشته‌هایی را که می‌خواهیم به کار ببریم را در قسمت `activity_main.xml`

بافرم روبرو نوشته و جمله مربوطه را فراخوانی می‌کنیم.

```
"@string/hello_world"
```

با این کار می‌توانیم هرچند باری که بخواهیم آن را فراخوانی کنیم بدون آن‌که لازم به نوشتن کل متن باشد.

در قسمت بعدی اگر بخواهیم یک دکمه هم قرار بدهیم مانند `TextView` جلو می‌رویم و مشخصه‌های زیر را برای آن مشخص می‌کنیم.

```
<Button
    android:id="@+id/btn1"
    android:layout_width="match_parent"
    android:layout_height="80dp"
    android:text="@string/name_dokme"
    android:layout_below="@id/text_view_helloworld"
    android:background="#f30f30"
/>
```

نکته‌ای که وجود دارد این است که باید با دستور

```
android:layout_below="@id/text_view_helloworld"
```

دکمه‌ی موردنظر ما باید در زیر `TextView` قرار بگیرد (برای این کار از شناسه‌ی `TextView` استفاده می‌کنیم).

و برای نوشتن جمله‌ای که قرار است درون دکمه ما قرار بگیرد به قسمت `activity_main.xml` رفته و به صورت زیر عمل می‌کنیم.

```
<string name="name_dokme">چاپ سلام</string>
```

**Name** = نامی که بعد قرار است در کد `activity_main.xml` با آن فراخوانی شود. (باید در دبل کوتیشن قرار بگیرد).

چاپ سلام = متنی که درون دکمه نوشته می‌شود.

در قسمت `activity_main.xml`، باید آن رشته به وسیله‌ی نامی که برای آن مشخص گردیده است فراخوانی شود.

حالا باید این متنی را که درون دکمه قرار دادیم درون یک `TextView` چاپ کند.

مجدداً یک `TextView` با مشخصه‌های زیر ایجاد می‌کنیم.

هر چیزی که ما ایجاد می‌کنیم باید یک شناسه‌ی خاص و متفاوت از بقیه داشته باشد.

```
<TextView
    android:id="@+id/matny_ke_gharar_ast_chap_beshe"
    android:layout_width="match_parent"
    android:layout_height="30dp"
    android:layout_below="@+id/btn1"
    android:textColor="#f30f30"
    android:background="#000000"
/>
```

دکمه‌ی موردنظر ما باید در زیر `Button` قرار بگیرد (برای این کار از شناسه‌ی `Button` استفاده می‌کنیم).

تا این قسمت موارد راجع به طراحی **UI** بوده است حال در ادامه به سراغ کد جاوا رفته و کار را به اتمام می‌رسانیم.

از قسمت `src`، قسمت `MainActivity.java` را باز کرده تا شروع به نوشتن کدهایمان بکنیم.

در قسمت `public class MainActivity extends Activity` هر چیزی را که ایجاد کرده‌ایم به صورت زیر تعریف

می‌کنیم. (زیر `public` و خارج از `onCreate` باید تعریف شوند).

برای مشخص کردن یک نوع از یک ویجت، اسم آن را نوشته و یک اسم دلخواه برای آن تعیین می‌کنیم و در آخر آن یک `z`

می‌گذاریم.

```
public class MainActivity extends Activity {
```

```
TextView jaye_matn;  
Button dokme;
```

بعد از تعریف ویجت ها باید اسم نوع‌هایی را که داریم در قسمت `onCreate` تعریف کنیم.

```
jaye_matn = (TextView) findViewById(R.id.matny_ke_gharar_ast_chap_beshe)  
dokme = (Button) findViewById(R.id.btn1);
```

در اصل با این دستور به آن فرمان می‌دهیم که از داخل `xml` ، با آن `id` که در `xml` تعیین کرده‌ایم، `TextView` را بخوانیم و درون `jaye_matn` قرار بدهیم.

به صورت زیر عمل کرده:

```
jaye_matn = (نوع) findViewById(R.id. اسم id آن); // متنی که قرار است چاپ شود  
dokme = (نوع) findViewById(R.id. اسم id آن);
```

همان‌طور که در قبل گفتیم ما می‌خواهیم با فشار دادن دکمه متنی را در `TextView` دوم چاپ کنیم، یعنی دکمه را حساس به کلیک کنیم.

جمله روبرو را می‌نویسیم

```
dokme.setOnClickListener(new View.OnClickListener())
```

بعد از نوشتن این عبارت به صورت خودکار ساختار زیر ایجاد می‌شود.

```
dokme.setOnClickListener(new View.OnClickListener() {  
    @Override  
    public void onClick(View arg0) {  
        // TODO Auto-generated method stub  
    }  
});
```

باید در انتهای این ساختار `( )` و `;` گذاشته شود (تا علامت قرمز رنگی که در مواقع `error` گذاشته می‌شود از بین برود).

در این قسمت `TODO Auto-generated method stub` ، هر چه بنویسیم دکمه آن را انجام می‌دهد.

```
jaye_matn.setText("رشته‌ای که می‌خواهیم بعد از زدن دکمه آن را چاپ کند");
```

`jaye_matn` یک `TextView` ای است که `id` اش مکانی است که باید متن باشد //

و بعد از هر عملی باید گزینه‌ی `save` حتماً زده شود.

در همین مسئله حالتی وجود دارد که متنی که خود کاربر بخواهد چاپ شود.

در این صورت باید در قسمت `activity_main.xml` یک `EditText` ( جای متنی که متن در آن قابل اصلاح باشد ) با مشخصه‌های زیر ایجاد نماییم.

```
<EditText
    android:id="@+id/matn_ra_migire"
    android:layout_width="match_parent"
    android:layout_height="50dp"
    android:layout_below="@+id/matny_ke_gharar_ast_chap_beshe"
/>
```

کد نهایی در قسمت `activity_main.xml` برای این پروژه به صورت زیر می‌باشد:

```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingBottom="@dimen/activity_vertical_margin"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    tools:context=".MainActivity" >

    <TextView
        android:id="@+id/text_view_helloworld"
        android:layout_width="match_parent"
        android:layout_height="30dp"
        android:background="#000000"
        android:text="@string/hello_world"
        android:textColor="#ffffff"
    />

    <Button
        android:id="@+id/btn1"
        android:layout_width="match_parent"
        android:layout_height="80dp"
        android:text="@string/name_dokme"
        android:layout_below="@id/text_view_helloworld"
        android:background="#f30f30"
    />

    <TextView
        android:id="@+id/matny_ke_gharar_ast_chap_beshe"
        android:layout_width="match_parent"
        android:layout_height="30dp"
        android:layout_below="@+id/btn1"
        android:textColor="#f30f30"
        android:background="#000000"
    />

    <EditText
        android:id="@+id/matn_ra_migire"
        android:layout_width="match_parent"
        android:layout_height="50dp"
```

```

        android:layout_below="@+id/matny_ke_gharar_ast_chap_beshe"
    />
</RelativeLayout>

```

کد نهایی در قسمت MainActivity.java برای این پروژه به صورت زیر می باشد:

```

public class MainActivity extends Activity {
    TextView jaye_matn;
    Button dokme;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        jaye_matn
= (TextView) findViewById(R.id.matny_ke_gharar_ast_chap_beshe);
        dokme = (Button) findViewById(R.id.btn1);
        dokme.setOnClickListener(new View.OnClickListener() {

            @Override
            public void onClick(View arg0) {
                // TODO Auto-generated method stub
                jaye_matn.setText("salam khoby??");
            }
        });
    }

    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        // Inflate the menu; this adds items to the action bar if it is
present.
        getMenuInflater().inflate(R.menu.main, menu);
        return true; }
}

```

اجرای آن به صورت زیر می باشد.



## ایجاد یک صفحه login

به همان روش قبلی یک پروژه ساخته و نام آن را LoginActivity می‌گذاریم.

زمانی که یک پروژه ساخته می‌شود یک textview به صورت پیش فرض برای Hello world ایجاد شده، برای اینکه آن را پاک کنیم به قسمت activity\_login.xml رفته و آن را پاک می‌کنیم.

در قسمت activity\_login.xml یک ویژگی با دستور android:background="#a838ef" برای زمینه می‌دهیم.

حال لوگویی را که داریم به صورت زیر در آن صفحه لاگین قرار می‌دهیم.

ابتدا بر روی آن عکسی که داریم گزینه‌ی کپی را می‌زنیم بعد از آن به پوشه‌ی res می‌رویم و بر روی آن کلیک راست می‌کنیم و New را می‌زنیم و گزینه‌ی Folder را انتخاب می‌کنیم و نام آن را drawable گذاشته و در پایان Finish را می‌زنیم.

حال در قسمت res فولدر جدیدی با این نام ایجاد شده است.

بعد از آن عکسی را که کپی کرده‌ایم در این فولدر کپی می‌کنیم.

بعد از آن می‌خواهیم آیکنی را که در این فولدر قرار دادیم در صفحه لاگین استفاده کنیم. برای این کار در قسمت

activity\_login.xml یک ImageView با مشخصات زیر برای نمایش عکسمان تعریف می‌کنیم.

```
<ImageView  
    android:id="@+id/logo_proje"  
    android:layout_width="90dp"  
    android:layout_height="90dp"  
    android:layout_centerHorizontal="true"  
    android:layout_marginTop="45dp"  
    android:background="@drawable/icon_login" />
```

نکته‌ای که وجود دارد ما می‌خواهیم background ما در این ImageView عکسی باشد که در قسمت drawable است. پس به همین خاطر آن را با عبارت زیر آدرس‌دهی می‌کنیم. (نام عکس‌هایی را که در پوشه‌ی drawable قرار می‌دهیم به هیچ عنوان نباید با عدد شروع شوند.)

"نام عکس/نام پوشه@"

حال با دستور زیر عکس مورد نظرمان را به وسط منتقل می‌کنیم.

```
android:layout_centerHorizontal="true"
```

با دستور زیر به آن یک فاصله از بالای صفحه می‌دهیم که خیلی در بالای صفحه قرار نگیرد.

```
android:layout_marginTop="45dp"
```

در مرحله بعد باید یک **EditText** با مشخصات زیر ایجاد کنیم.

```
<EditText
    android:id="@+id/edt1"
    android:layout_width="250dp"
    android:layout_height="50dp"
    android:layout_below="@+id/logo_proje" // برای آنکه زیر لوگوی پروژه قرار بگیرد
    android:layout_centerHorizontal="true"
    android:hint="محل وارد کردن نام کاربری"
```

در مرحله بعد یک **EditText** با همان ویژگی‌ها طراحی کرده، اما باید ۲ مشخصه آن را تغییر بدهیم ( **id** و عنوان چیزی که قرار است در درون خودش جا بدهد: محل وارد کردن رمز عبور).

```
<EditText
    android:id="@+id/edt2"
    android:layout_width="250dp"
    android:layout_height="50dp"
    android:layout_below="@+id/edt1"
    android:layout_centerHorizontal="true"
    android:background="#ffffff"
    android:hint="محل وارد کردن رمز عبور"
    android:inputType="textPassword"
    android:layout_marginTop="5dp"
/>
```

برای وارد کردن رمز عبور به مشخصه‌ای احتیاج داریم که رمز عبور را به صورت نقطه نمایش دهد، به همین دلیل از مشخصه‌ی **android:inputType="textPassword"** استفاده می‌کنیم.

برای اینکه قبل از آنکه کاربر شروع به پر کردن **EditText** بکند بداند که در هر **EditText** چه چیزی را باید بنویسد از مشخصه‌ی **android:hint** استفاده می‌شود (به صورت کمرنگ درون آن قرار می‌گیرد).

برای اینکه **EditText** دوم از **EditText** اولی فاصله داشته باشد از دستور زیر استفاده می‌شود.

```
android:layout_marginTop
```



در این مرحله باید یک دکمه بانام **Button** و مشخصات زیر قرار بدهیم.

```
<Button
    android:id="@+id/btn1"
    android:layout_height="50dp"
    android:layout_width="250dp"
    android:layout_below="@+id/edt2"
    android:layout_centerHorizontal="true"
    android:layout_marginTop="20dp"
    android:background="#f30f30"
    android:text="ورود"

/>
```

نکته‌ای که وجود دارد این است که باید دکمه را زیر **edt2** قرار بدهیم به همین دلیل طبق دستوراتی که در گذشته گفته شده است از دستور زیر استفاده می‌کنیم.

```
android:layout_below="@+id/edt2"
```

تا این قسمت ما طراحی **UI** را انجام دادیم و بعدازآن به سراغ فایل **src** می‌رویم.

در این قسمت ابتدا عنصرهایی را که استفاده کردیم تعریف کرده:

```
EditText edt1,edt2;
```

```
Button btn1;
```

در قسمت بعدی با دستورات زیر باید آن‌ها را در **onCreate** تعریف کنیم و آن‌ها را از **xml** بخوانیمشان.

```
edt1 = (EditText) findViewById(R.id.edt1);
edt2 = (EditText) findViewById(R.id.edt2);
btn1 = (Button) findViewById(R.id.btn1);
```

حال باید زمانی که **btn1** زده شد بررسی شود که آیا مقادیر **edt1,edt2** درست است و یا نه اجازه ورود به صفحه بعدی داده شود، به همین دلیل از دستور زیر استفاده می‌شود:

```
btn1.setOnClickListener(new View.OnClickListener())
```

بعد از نوشتن این عبارت به صورت خودکار ساختار زیر ایجاد می‌شود.

```
btn1.setOnClickListener(new View.OnClickListener() {
    @Override
```

```

public void onClick(View arg0) {
    // TODO Auto-generated method stub
}
});

```

همان‌طور که در گذشته گفته شد باید در انتهای این ساختار ( `__` و `;` گذاشته شود) تا علامت قرمز رنگی که در مواقع `error` گذاشته می‌شود از بین برود.)

حال ما باید در قسمت `TODO Auto-generated method stub`، شرط‌هایمان را بنویسیم که اگر برقرار بودند به اکتیویته بعدی برود.

به این صورت که دو مقدار واقعی وجود دارد که کاربر هر مقداری وارد کند با آن ۲ مقدار واقعی مقایسه شده و دو مقدار هم وجود دارد که به وسیله کاربر وارد می‌شوند.

۲ مقدار واقعی چون فعلاً در ابتدای کار بانک‌ها را نخوانده‌ایم به صورت یک داده وجود دارد و هر چیزی که وارد شود با آن داده (نام کاربری و ۱ رمز عبور) مقایسه شده و به اکتیویته بعدی رفته.

به همین دلیل باید رشته تعریف کنیم.

این داده‌ها را به صورت زیر تعریف کرده:

```

String str_user_motabar ;
String str_pass_moatabar;
String str_user_feli;
String str_pass_feli;

```

حال در قسمتی که گفتیم (`TODO Auto-generated method stub`) شرط‌ها را وارد می‌کنیم.

```

public void onClick(View arg0) {
    // TODO Auto-generated method stub
    str_user_motabar = "mahdi";
    str_pass_moatabar = "1234";
    str_user_feli = edt1.getText().toString();
    str_pass_feli = edt2.getText().toString();

    if(str_user_feli.equals(str_user_motabar) &&
    str_pass_feli.equals(str_pass_moatabar)) {

        Intent i = new Intent();
        i.setClass(LoginActivity.this,
        Activity_safe_asli.class);
        startActivity(i);
    }
}

```

```

else
{
    Toast.makeText(getApplicationContext(), " نام کاربری و یا رمز عبور "
    Toast.LENGTH_SHORT).show();
}
}
});

```

به این صورت عمل می‌کند که ما یک `user,pass` فعلی (یوزر و پسوردی که یک کاربر هنگام ورود وارد می‌کند) داریم و یک `user,pass` معتبر (یوزر و پسوردی که در سیستم از قبل ثبت شده است) داریم که در این قسمت باهم مقایسه می‌شوند و اگر باهم همخوانی داشتند به اکتیویتی (صفحه) بعدی می‌رود، در غیر این صورت چاپ می‌کند که رمز و یا نام کاربری اشتباه هست. برای اینکه به اکتیویتی بعدی برود باید آن اکتیویتی را از قبل بسازیم و نام آن را اکتیویتی صفحه اصلی می‌گذاریم، برای این کار در قسمت `src`، بر روی `com.er.loginproject` کلیک راست کرده و `New` را انتخاب کرده و گزینه‌ی `Class` را می‌زنیم و یک صفحه باز می‌شود و در آن نام اکتیویتی را مشخص می‌کنیم، `Activity_safe_asli.java` و `superclass` آن را `android.app.Activity` می‌گذاریم و گزینه `finish` را می‌زنیم.

بعد از آن باید `layout` آن را در قسمت `res` بسازیم، کلیک راست کرده و `New` را می‌زنیم و بر روی `File` کلیک می‌کنیم تا صفحه‌ای باز شود و نام آن را `safe_asli.xml` انتخاب می‌کنیم.

در قسمت `Activity_safe_asli.java`، ما فقط یک صفحه خالی ایجاد می‌کنیم و فقط کد `onCreate` را در آن وارد می‌کنیم.

```

@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.safe_asli);
}

```

حال گفتیم اگر شرط برقرار بود باید به اکتیویتی بعدی برویم، برای این کار قطعه کد زیر را درون دستور شرطمان می‌نویسیم.

```

Intent i = new Intent();
i.setClass(LoginActivity.this, Activity_safe_asli.class);
startActivity(i);

```

در غیر این صورت باید در قسمت `else` دستور زیر را بنویسیم:

```

Toast.makeText(getApplicationContext(),
    Toast.LENGTH_SHORT).show();

```

زمانی که بخواهیم یک پیامی را به کاربر نشان دهیم هم می‌توانیم از آلت دیالوگ (یک پیام روی صفحه می‌آید) استفاده کنیم و یا از Toast (پیام خاکستری‌رنگی که به صورت چند ثانیه ظاهر می‌شود برای نمایش به کاربر) استفاده کنیم.

نکته: برای معرفی و مقدار دادن به رشته‌ها بهتر است در همان دکمه نام‌گذاری شود.

زمانی که بخواهیم یک اکتیویته باز شود و یا به صفحه بعد برود باید اجازه آن در **AndroidManifest.xml** داده شود، این کار به صورت زیر انجام می‌گیرد:

```
<activity
    android:name="com.er.loginproject.Activity_safe_asli"
    android:label="@string/app_name" >

</activity>
```

اما برای اکتیویته اول به صورت خودکار قطعه کد زیر تولید شده که برای اکتیویته‌های بعدی به خط اول آن احتیاج است.

```
<activity
    android:name="com.er.loginproject.LoginActivity"
    android:label="@string/app_name" >
    <intent-filter>
        <action android:name="android.intent.action.MAIN" />

        <category android:name="android.intent.category.LAUNCHER" />
    </intent-filter>
</activity>
```

کدهای نهایی در قسمت **activity\_login.xml** برای **login** کردن در هر سیستمی به صورت زیر می‌باشد:

```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="#a838ef"
    android:paddingBottom="@dimen/activity_vertical_margin"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    tools:context=".LoginActivity" >
```

```
<ImageView
    android:id="@+id/logo_proje"
```

```
android:layout_width="90dp"
android:layout_height="90dp"
android:layout_centerHorizontal="true"
android:layout_marginTop="45dp"
android:background="@drawable/icon_login" />
```

```
<EditText
    android:id="@+id/edt1"
    android:layout_width="250dp"
    android:layout_height="50dp"
    android:layout_below="@+id/logo_proje"
    android:layout_centerHorizontal="true"
    android:hint="کاربری نام وارد کردن محل"

    android:background="#ffffff" />
```

```
<EditText
    android:id="@+id/edt2"
    android:layout_width="250dp"
    android:layout_height="50dp"
    android:layout_below="@+id/edt1"
    android:layout_centerHorizontal="true"
    android:background="#ffffff"
    android:hint="عبور رمز وارد کردن محل"
    android:inputType="textPassword"
    android:layout_marginTop="5dp"
/>
```

```
<Button
    android:id="@+id/btn1"
    android:layout_height="50dp"
    android:layout_width="250dp"
    android:layout_below="@+id/edt2"
    android:layout_centerHorizontal="true"
    android:layout_marginTop="20dp"
    android:background="#f30f30"
    android:text="ورود"
```

```
/>
```

```
</RelativeLayout>
```

کد نهایی در قسمت LoginActivity.java به صورت زیر می باشد:

```
public class LoginActivity extends Activity {
    EditText edt1,edt2;
    Button btn1;
    String str_user_motabar ;
    String str_pass_moatabar;
    String str_user_feli;
    String str_pass_feli;
```

```

@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_login);
    edt1 = (EditText) findViewById(R.id.edt1);
    edt2 = (EditText) findViewById(R.id.edt2);
    btn1 = (Button) findViewById(R.id.btn1);

    btn1.setOnClickListener(new View.OnClickListener() {

        @Override
        public void onClick(View arg0) {
            // TODO Auto-generated method stub
            str_user_motabar = "mahdi";
            str_pass_moatabar = "1234";
            str_user_feli = edt1.getText().toString();
            str_pass_feli = edt2.getText().toString();

            if(str_user_feli.equals(str_user_motabar) &&
str_pass_feli.equals(str_pass_moatabar)) {

                Intent i = new Intent();
                i.setClass(LoginActivity.this,
Activity_safe_asli.class);
                i.putExtra("nam", str_user_feli);
                startActivity(i);

            }
            else
            {
                Toast.makeText(getApplicationContext(), "رمز یا و کاربری نام
است اشتباه عبور", Toast.LENGTH_SHORT).show();
            }

        }

    });
}
}

```

در پروژه‌ی قبل هر کاربر با یک **user** خاص وارد می‌شود و به صفحه اصلی می‌رود، حال اگر بخواهیم در صفحه‌ی اصلی به آن کاربر خوش‌آمد بگوییم باید دستورات زیر را اجرا کنیم.

```
i.putExtra("nam",str_user_feli);
```

**num**: نام تگ ماست یعنی چیزی که قرار است در اکتیویتی مقصد بگیریم (در اکتیویتی مقصد یک مقداری از Intent بگیر که اسم آن **num** است).

str\_user\_feli : مقدار آن می باشد که همان اسمی که user فعلی وارد می کند.

در این قسمت کار ما تمام شده است و ما به سراغ قسمت Activity\_safe\_asli.java می رویم.

در قسمت safe\_asli.xml یک TextView با مشخصات زیر می سازیم.

```
<TextView
    android:id="@+id/txt1"
    android:layout_height="50dp"
    android:layout_width="match_parent"
    android:textColor="#000000"
    android:background="#f30f30"

/>
```

حال در قسمت Activity\_safe\_asli.java ، onCreate آن را ایجاد کرده با دستور زیر:

```
@Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.safe_asli);
```

\*\*\* یک TextView بانام txt1 تعریف می کنیم.

\*\*\* یک رشته ی username هم تعریف کرده.

\*\*\* با جمله زیر txt1 را از xml فراخوانی کرده:

```
txt1 = (TextView) findViewById(R.id.txt1);
```

برای آنکه بتوانیم آن چیزی را که در اکتوییتی مبدأ ارسال کردیم با تگ name را اینجا بگیریم از دستور زیر استفاده می کنیم:

```
Intent gereftan_nam = getIntent ();
```

در آن تابع getIntent را فراخوانی کردیم.

Username ای که در بالا تعریف کردیم را برابر با gereftan\_nam قرار داده و به صورت زیر نمایش می دهیم:

```
username = gereftan_nam.getStringExtra("نام تگ موردنظر ما");
```



یعنی آن چیزی که putExtra شد را بگیرد

حال برای نمایش آن از دستور زیر استفاده می کنیم:

```
txt1.setText(username+"خوش آمدید");
```

در کل به صورت زیر هست:

```
public class Activity_safe_asli extends Activity {
    TextView txt1;
    String username;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.safe_asli);
        txt1 = (TextView) findViewById(R.id.txt1);
        Intent gereftan_nam = getIntent();
        username = gereftan_nam.getStringExtra("nam");
        txt1.setText(username+"خوش آمدید");
    }
}
```

### ایجاد یک صفحه‌ی Splash

یعنی اینکه صفحه‌ی اول ما به مدت چند ثانیه نمایش داده شود و بعد آن به صفحه‌ی بعد برود.

ابتدا قبل از هر کاری یک پروژه به نحوه‌ای که گفته شده است، می‌سازیم و نام آن را Splash\_edu می‌گذاریم.

در آن نام اکتوییتی اول را SplashActivity می‌گذاریم.

در قسمت src، ما یک صفحه بانام SplashActivity.java داریم و می‌خواهیم در آن به مدت چند میلی‌ثانیه بماند و پس آن به اکتوییتی بعدی برو برای همین باید در این قسمت یک اکتوییتی دیگر ایجاد کنیم.

این اکتوییتی را به روشی که قبلاً گفته شد ایجاد می‌کنیم و نام آن را MainActivity.java می‌گذاریم.

حال به سراغ قسمت res آن می‌رویم و فایل xml آن را می‌سازیم و نام آن را activity\_main.xml می‌گذاریم.

حال به سراغ قسمت activity\_main.xml می‌رویم و در آن یک RelativeLayout با مشخصه‌های زیر می‌سازیم.

```
<RelativeLayout
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="#f30f30"
```

حال به سراغ ساخت یک صفحه‌ی Splash می‌رویم.

در قسمت src و در اکتوییتی MainActivity.java، با دستور زیر onCreate آن را می‌دهیم.

```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
}
```



حالا از اکتوییتی MainActivity.java بیرون می‌آییم و تمام کارها را در قسمت MainActivity.java انجام می‌دهیم.

برای این کار ابتدا یک عکس را کپی کرده و بعد آن یک پوشه‌ی drawable داخل res ایجاد می‌کنیم و عکس موردنظرمان را که می‌خواهیم در صفحه اول قرار بگیرد در آن کپی می‌کنیم.

حال به قسمت activity\_splash.xml می‌رویم و در آن یک خاصیت background به قسمت RelativeLayout اضافه می‌کنیم برای آنکه بتوانیم آن عکسی که در پوشه‌ی drawable قرار دارد را به‌عنوان زمینه به این صفحه بدهیم.

```
android:background="@drawable/splash_screen"
```

حال وقتی آن را به‌صورت گرافیکی مشاهده می‌کنیم در بالای این اکتوییتی MainActivity نوشته‌شده است و ما می‌خواهیم عکس ما به‌صورت تنها و بدون هیچ نوشته‌ای قرار داشته باشد، برای این کار به‌صورت زیر عمل می‌کنیم:

در قسمت گرافیکی بر روی گزینه‌ی AppThemes کلیک کرده و گزینه‌ی Theme را زده و گزینه‌ی

Fullscreen را انتخاب می‌کنیم. (با این کار تایتل بار آن از بین می‌رود).

حال ممکن است با این کار در اجرا همچنان به‌صورت Fullscreen نشود به همین دلیل باید در قسمت AndroidManifest.xml ، Theme آن به‌صورت Fullscreen تعریف کنیم و در قسمت MainActivity.java با

نوشتن ۲ خط کد Fullscreen بودن را نمایش دهیم.

در فایل جاوا در متد onCreate آن کد زیر را اضافه می‌کنیم:

```
this.getWindow().setFlags(WindowManager.LayoutParams.FLAG_FULLSCREEN,
    WindowManager.LayoutParams.FLAG_FULLSCREEN);
```

بعد از این کار به سراغ ادامه کارمان می‌رویم.

ما باید تایمی را به آن بدهیم و بگوییم بعد از این تایم به سراغ اکتوییتی بعدی برو پ در همان فایل جاوا کارهای زیر را انجام داده.

ابتدا در آن یک نوع از نوع private تعریف کرده: (به این دلیل آن را خصوصی تعریف کردیم چون آن را فقط در همین کلاس احتیاج داریم)

```
private final int splash_time = 3000; // به اندازه ۳۰۰۰ میلی ثانیه
```

در چرخه‌ی اکتیویته‌ها یک متد `onStart` داریم، یعنی زمانی که یک متد `onStart` شد. حالا می‌خواهیم از این متد استفاده کنیم به همین دلیل کار زیر را انجام می‌دهیم:

```
protected void onStart() {  
    super.onStart();  
}
```

حال برای اینکه بخواهیم تایمر ایجاد کنیم از خاصیت چند نخ‌ی و یا `Thread` استفاده می‌کنیم. (یعنی یک کار را به `cpu` سپرده و می‌گوییم طی یک‌زمان مشخص کاری را انجام بده.

به همین دلیل کدهای زیر را می‌نویسیم:

```
new Handler().postDelayed(new Thread() {  
  
    @Override  
    public void run() {  
        سازنده- خودشان باید اجرا شوند //  
        Intent i = new Intent();  
        i.setClass(SplashActivity.this, MainActivity.class);  
        startActivity(i);  
        SplashActivity.this.finish(); // بسته شده //  
    }  
    }, splash_time); // برای اینکه زمانی را صبر کند و به اکتیویته بعدی برود //  
}
```

حال می‌خواهیم به اکتیویته `main` برویم، اما آن را ذخیره نکرده‌ایم در `AndroidManifest.xml` به همین در قسمت `AndroidManifest.xml` کد زیر را وارد کرده:

```
android:theme="@android:style/Theme.Black.NoTitleBar.Fullscreen"
```

`Theme` سیاه باشد، `TitleBar` نداشته باشد و به صورت `Fullscreen` باشد.

حال باید به اکتیویته دوم `main` هم در این قسمت اجازه داده شود، پس به صورت زیر عمل کرده:

```
<activity  
    android:name="com.example.splash_edu.MainActivity"  
    android:screenOrientation="portrait"  
    android:label="@string/app_name" > </activity>
```

## نحوه افکت گذاری بر روی اکتویته ها

در قسمت `res` و در قسمت `drawable` باید تا `xml` بسازیم، یکی برای اینکه اکتویته اول با یک انیمیشن خاص وارد اکتویته دوم شود و انیمیشنی که اکتویته اول از صفحه با آن خارج شود. (روی هم ۳۰۰۰ میلی ثانیه است)

اولی بانام `fade_in.xml` و دومی بانام `fade_out.xml` را می سازیم.

به این `xml` ها ویژگی هایی را داده و بعد آن ها را به اکتویته اول می فرستیم.

اکتویته `fade_in.xml` را به صورت زیر می سازیم:

(انواع حالت انیمیشن درون اینترنت وجود دارد و می توان از آن استفاده کرد.)

```
<?xml version="1.0" encoding="utf-8"?>
<alpha xmlns:android="http://schemas.android.com/apk/res/android"
    android:duration="2000"
    android:fromAlpha="0.0"//یک نوع افکت است
    android:interpolator="@android:anim/accelerate_interpolator"
    android:toAlpha="1.0" />
```

آن را سیو کرده و به سراغ `fade_out.xml` می رویم و آن را به صورت زیر می سازیم:

```
<?xml version="1.0" encoding="utf-8"?>
<alpha xmlns:android="http://schemas.android.com/apk/res/android"
    android:duration="2000"
    android:fromAlpha="1.0"
    android:interpolator="@android:anim/accelerate_interpolator"
    android:toAlpha="0.0" />
```

حال این دو افکت خاص را ساخته ایم و بعدازآن به سراغ اکتویته ها رفته و بر روی آن ها اعمال می کنیم.

در `SplashActivity.java` رفته و در آخرین خط قسمتی که به صورت خودکار اجرا می شد خط زیر را نوشته:

```
overridePendingTransition(R.drawable.fade_in,R.drawable.fade_out);
```



کد کامل این برنامه در قسمت `SplashActivity.java` به صورت زیر می باشد:

```
public class SplashActivity extends Activity {
    private final int splash_time = 3000;
    @Override

    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_splash);
    }
}
```

```

this.getWindow().setFlags(WindowManager.LayoutParams.FLAG_FULLSCREEN,
    WindowManager.LayoutParams.FLAG_FULLSCREEN);
}

protected void onStart() {
    super.onStart();

    new Handler().postDelayed(new Thread() {

        @Override
        public void run() {
            super.run();
            Intent i = new Intent();
            i.setClass(SplashActivity.this, MainActivity.class);
            startActivity(i);
            SplashActivity.this.finish();
            overridePendingTransition(R.drawable.fade_in, R.drawable.fade_out);

        }
    }, splash_time);
}
}

```

## پروژهی ALARM در ساعت مشخص

ابتدا یک پروژهی جدید می‌سازیم و اسم آن را **Notifer** (یادآور) می‌گذاریم. و نام اکتیویتهی آن را **MainActivity** گذاشته.

در قسمت **activity\_main.xml** در ابتدا یک کلید با مشخصات زیر ساخته:

```

<Button
    android:id="@+id/button1"
    android:layout_width="match_parent"
    android:layout_height="50dp"
    android:text="یادآوری" />

```

کاربرد این دکمه این است که زمانی که بر روی آن دکمه زده شد، یادآور کوک شود.

زمانی که دکمه را ساختیم از این قسمت خارج شده و به قسمت **MainActivity.java** می‌رویم.

دکمه را باید حساس به کلیک بکنیم که به یک **Broadcast Reciver** برود.

می‌خواهیم یک اکتیویتهی بسازیم که از نوع **Broadcast Reciver** باشد.

دو راه وجود دارد:

۱. یک کلاس جاوا خالی بسازیم و بعد در آن `extends Broadcast Reciver`

یعنی ارث‌بری بکن از کلاس `Broadcast Reciver` اندروید.

۲. برای ساخت یک `Broadcast Reciver` ، کلیک راست می‌کنیم، `New` و بعد از آن `Class` را می‌زنیم و `superclass` آن را از نوع `Broadcast Reciver` می‌گذاریم.

همان‌طور که مشاهده می‌کنیم یک کلاس برای ما ساخت بانام `Reciver.java` و خودش ارث‌بری کرد از `Broadcast Reciver` :

```
public class Reciver extends BroadcastReceiver {  
  
    @Override  
    public void onReceive(Context context, Intent intent) {  
        // TODO Auto-generated method stub  
    }  
}
```

قرار است رسیوری باشد که گوش می‌دهد تا ساعت ما را `set` کند، یعنی چیزهایی را که قرار است چه ساعتی `set` شود در این قسمت `set` می‌کنیم.

یک تابع `onReceive` دارد، که در بین جمله `public class Reciver extends BroadcastReceiver` و کلمه `@Override`

ثابت‌هایی را که می‌خواهیم تعریفشان کرده و بعد در قسمت زیر می‌گوییم باید آن‌ها را اجرا کنی.

```
public void onReceive(Context context, Intent intent) {  
    // TODO Auto-generated method stub  
}
```

در این قسمت دو ثابت تعریف می‌کنیم که تا پایان برنامه مقدارش تغییری نکند:

```
public static final String ACTION_SET_ALARM = "com.ir.notifer.SET_ALARM";
```

عملیاتی که قرار است `ALARM` ، `set` شود باید معرفی کنیم که چه پکیجی است که می‌خواهد `ALARM` را `set` کند. نام پکیج ما در بالای کد مربوط به `Reciver.java` نوشته شده و عبارت است از `com.ir.notifer`

در حقیقت رشته‌ای است که قرار است برای ثبت **ALARM** باشد. (ثبت **ALARM**)

```
public static final String ACTION_SHOW_NOTIFICATION =  
"com.ir.notifer.SHOW_NOTIFICATION";
```

این رشته برای **NOTIFICATION** دادن به کاربر استفاده شده. که ایجاد آن مانند رشته قبلی است که فقط نام آن متفاوت است چون برای **NOTIFICATION** دادن به کاربر است. (نشان دادن **ALARM**)

حال در تابع **onReceive** ، آرگومان داریم که نام آن‌ها را **context, intent** می‌گذاریم (همان ورودی‌های تابع ما می‌باشد).

**Context** : شامل اکتویته‌ها، فعالیت‌های مربوط به اکتویته‌ها می‌باشد.

**Intent** : راه‌های ارتباطی در اندروید را شامل می‌شود.

در قسمت **onReceive** ، باید یک تابع **SET\_ALARM** و یک تابع **SHOW\_NOTIFICATION** نوشته شود.

**SET\_ALARM** : **set** کنیم که چه زمانی می‌خواهیم **ALARM** اتفاق بیفتد. (در اصل شرط ما در این قسمت نوشته شده و نحوه‌ی کار توابع در خارج از آن تعریف می‌شود)

**SHOW\_NOTIFICATION** : وقتی زمانش رسید چه شکلی نشان دهد.

در قسمت **onReceive** ، ما یک شرط باید بنویسیم که می‌گوید اگر عملیاتی را که **set** کردیم زمانش رسیده است، **NOTIFICATION** را نشان بده در غیر این صورت باید **SET\_ALARM** بکند.

فعالاً از قسمت **onReceive** بیرون آمده و شروع به نوشتن تابع **SET\_ALARM** می‌کنیم.

تمام این عملیات به‌جز زمان آن برای هر یادآور ثابت می‌باشد.

```
private void setAlarm(Context context)  
{
```

```
    Intent intent = new Intent(context, this.getClass()); // یکسری تنظیمات از یک کلاس  
    دیگر می‌آید که ما اینجا باید آن‌ها را بگیریم.  
    کلاسی که قرار است بیاید (ورودی تابع).  
}
```

```
    intent.setAction(ACTION_SHOW_NOTIFICATION); // ست کرده است  
    PendingIntent pendingintent = PendingIntent.getBroadcast(context, 0,  
    intent, 0);
```

کد درخواستی  flag

```
AlarmManager alarmManager =  
(AlarmManager) context.getSystemService(context.ALARM_SERVICE);  
alarmManager.cancel(pendingintent);
```

نحوه ی ساخت یک نوع Calendar

```
Calendar calender = Calendar.getInstance(); // یک نوع از تقویم می سازیم  
calender.setTimeInMillis(System.currentTimeMillis()); // تبدیل به میلی  
calender.set(calender.HOURL OF DAY, 15); // تنظیم ساعت  
calender.set(calender.MINUTE, 19); // تنظیم دقیقه
```

```
alarmManager.setRepeating(AlarmManager.RTC_WAKEUP,  
calender.getTimeInMillis(), alarmManager.INTERVAL_DAY, pendingintent);  
}
```

AlarmManager و Calendar : یک نوع مانند string,int می باشد.


شروع به نوشتن تابع SHOW\_NOTIFICATION می کنیم.

یک نوع نوتیفیکشن تعریف کرده و برای آن یک آیکون و عنوان تعریف کرده.

```
private void showNotification(Context context){  
    long adad[] = {2000,2000};
```

```
    NotificationCompat.Builder mBuilder= new  
    NotificationCompat.Builder(context).setSmallIcon(R.drawable.  
ic_launcher).setContentTitle("آنلاین آزمون").setVibrate(adad);
```

```
    NotificationManager mNotificationManager =  
(NotificationManager) context.getSystemService  
(context.NOTIFICATION_SERVICE);  
    mNotificationManager.notify(200,mBuilder.build());
```

مدت زمان ویبره 

مدیریت کننده- چه شکلی ظاهر شود و چقدر بماند //

```
}
```

تا این قسمت دو تابعی را که احتیاج داشتیم، ساخته‌ایم و حال به سراغ تابع `onReceive` می‌رویم و کد زیر را نوشته:

اگر زمان نوتیفیکیشن فرارسیده است، برو و آن را اجرا کن.

```
@Override
public void onReceive(Context context, Intent intent) {
    // TODO Auto-generated method stub
    if(intent.getAction().equalsIgnoreCase(ACTION_SHOW_NOTIFICATION)) {

        showNotification(context); // صدا کردن تابع نوتیفیکیشن
        return;
    }

    setAlarm(context); // در غیر آن صورت برو و محاسبات ست آلارم را انجام بده
}
```

تا این قسمت کارهای مربوط به تایمر را انجام داده و فقط ما باید در قسمت `main` کلید را حساس به کلیک کنیم.

ابتدا یک دکمه تعریف کرده:

```
Button btn1 ;
```

بعد از آن در قسمت `onCreate` آن را تعریف می‌کنیم:

```
btn1 = (Button) findViewById(R.id.button1);
```

حال آن را حساس به کلیک کرده:

```
btn1.setOnClickListener(new View.OnClickListener())
```

با نوشتن این جمله یک تابع به صورت زیر ایجاد می‌شود:

```
btn1.setOnClickListener(new View.OnClickListener() {

    @Override
    public void onClick(View arg0) {
        // TODO Auto-generated method stub
```

در این قسمت ما می‌خواهیم `Reciver` را صدا بزنیم، به همین دلیل کد زیر را نوشته:

```
Intent intent = new Intent(MainActivity.this, Reciver.class);
```

در اینجا می‌خواهیم تابع `Reciver` را صدا بزنیم، اول کلاسی را که در آن هستیم می‌نویسیم و بعد از آن کلاس مقصد



```

intent.setAction(Receiver.ACTION_SET_ALARM); //تنظیم آلارم
sendBroadcast(intent);

Toast.makeText(MainActivity.this, " آلارم تنظیم شد ",
Toast.LENGTH_SHORT).show(); // چاپ یک پیام

```

مدت کوتاهی صبر کند.

در این مرحله ما اگر اجرا بگیریم به ما خطا می دهد، چون در قسمت **AndroidManifest.xml** اجازه را صادر نکرده ایم. پس به قسمت **AndroidManifest.xml** رفته و کارمان را به پایان می رسانیم، باید یک نوع از **Receiver** تعریف کنیم. ما کلمه **Receiver** را تایپ کرده و خودش تگ را باز و بسته می کند برایمان. در این **Receiver** قرار است به آن یک نام بدهیم که ما نام پکیجمان را تایپ می کنیم.

**android:name="com.ir.notifer.Reciver**

در این قسمت باید یکسری گزینه را **set** کنیم، برای این قسمت هم باید فیلتر را قرار بدهیم.

هم باید برای تابع **setAlarm** و **showNotification** و یک قابلیت **BOOT\_COMPLETED**

برای ما اجازه صادر نماید.

درون **intent-filter** ها باید عملیات تعریف شود.

```

</activity>

<receiver android:name="com.ir.notifer.Reciver">
  <intent-filter>
    <actionandroid:name="android.intent.action.BOOT_COMPLETED"/> در
    <action android:name="com.ir.notifer.SETALARM"/> // در پکیج خودمان است
    <actionandroid:name="com.ir.notifer.SHOW_NOTIFICATION"/>
  </intent-filter>
</receiver>

```

این پروژه هم به اتمام رسیده است، کدهای کلی آن به صورت زیر می باشد.

کدها در قسمت **Receiver.java** به صورت زیر می باشد:

```

public class Reciver extends BroadcastReceiver {

    public static final String ACTION_SET_ALARM = "com.ir.notifer.SET_ALARM";

```

```

    public static final String ACTION_SHOW_NOTIFICATION =
"com.ir.notifer.SHOW_NOTIFICATION";

    @Override
    public void onReceive(Context context, Intent intent) {
// TODO Auto-generated method stub
    if(intent.getAction().equalsIgnoreCase(ACTION_SHOW_NOTIFICATION)) {

        showNotification(context);
        return;
    }
        setAlarm(context);    }
    private void setAlarm(Context context)
    {
        Intent intent = new Intent(context, this.getClass());
        intent.setAction(ACTION_SHOW_NOTIFICATION);
        PendingIntent pendingintent = PendingIntent.getBroadcast(context, 0,
intent, 0);

        AlarmManager alarmManager =
(AlarmManager) context.getSystemService(context.ALARM_SERVICE);
        alarmManager.cancel(pendingintent);

        Calendar calender = Calendar.getInstance();
        calender.setTimeInMillis(System.currentTimeMillis());
        calender.set(calender.HOUR_OF_DAY, 15);
        calender.set(calender.MINUTE, 19);

        alarmManager.setRepeating(AlarmManager.RTC_WAKEUP,
calender.getTimeInMillis(), alarmManager.INTERVAL_DAY, pendingintent);

    }

    private void showNotification(Context context){
        long adad[] = {2000,2000};
        NotificationCompat.Builder mBuilder = new
NotificationCompat.Builder(context).setSmallIcon(R.drawable.ic_launcher).setConte
ntTitle("آنلاین آزمون").setVibrate(adad);

        NotificationManager mNotificationManager =
(NotificationManager) context.getSystemService(context.NOTIFICATION_SERVICE);
        mNotificationManager.notify(200,mBuilder.build());
    }
}

```

کدها در قسمت MainActivity.java به صورت زیر می باشد:

```

public class MainActivity extends Activity {
    Button btn1 ;

    @Override
    protected void onCreate(Bundle savedInstanceState) {

```

```

        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        btn1 = (Button)findViewById(R.id.button1);

        btn1.setOnClickListener(new View.OnClickListener() {

            @Override
            public void onClick(View arg0) {
                // TODO Auto-generated method stub
                Intent intent = new Intent(MainActivity.this, Reciver.class);
                intent.setAction(Reciver.ACTION_SET_ALARM);

                sendBroadcast(intent);

                Toast.makeText(MainActivity.this, "شد تنظیم آلارم",
                Toast.LENGTH_SHORT).show();
            }
        });
    }
}

```

کدها در قسمت AndroidManifest.xml به صورت زیر می باشد:

```

<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.ir.notifer"
    android:versionCode="1"
    android:versionName="1.0" >

    <uses-sdk
        android:minSdkVersion="11"
        android:targetSdkVersion="17" />

    <application
        android:allowBackup="true"
        android:icon="@drawable/ic_launcher"
        android:label="@string/app_name"
        android:theme="@style/AppTheme" >
        <activity
            android:name="com.ir.notifer.MainActivity"
            android:label="@string/app_name" >
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

```

```

        <category android:name="android.intent.category.LAUNCHER" />
    </intent-filter>
</activity>

<receiver android:name="com.ir.notifer.Reciver">

    <intent-filter>
    <action android:name="android.intent.action.BOOT_COMPLETED"/>
    <action android:name="com.ir.notifer.SETALARM"/>
    <action android:name="com.ir.notifer.SHOW_NOTIFICATION"/>

    </intent-filter>
</receiver>

</application>
</manifest>

```

### طریقه‌ی ساختن یک پایگاه داده در اندروید

پایگاه داده اندروید **sql lite** است (ترجیحاً از نسخه‌ی ۳,۴,۷۷,۲۳۰۷ **SQLite Expert Professional** استفاده نمایید) که از چند جدول تشکیل شده است و تقریباً ساده می‌باشد ولی قدرتمند عمل کرده.

زمانی که می‌خواهیم پایگاه داده داشته باشیم به **eclipse** رفته.

قبلاً گفته شد که جای پایگاه داده در پوشه‌ی **asset** می‌باشد. ما باید زمانی که پایگاه داده را ایجاد کردیم، فایلش را در پوشه‌ی **asset** کپی نماییم.

برای ساخت یک پایگاه داده در قسمت **File**، گزینه‌ی **New database** را می‌زنیم و یک صفحه ایجاد شده که از ما یکسری مشخصات را خواسته، نام دیتابیس را از ما پرسیده که ما آن را **azmon.db** را می‌زنیم و بر روی دکمه آن کلیک کرده و مسیرش را به آن بدهیم به‌عنوان مثال مسیر آن را دسکتاپ انتخاب می‌کنیم و نوع آن را **UTF-8** انتخاب کرده که بتوان در آن فارسی هم نوشت و بعد از آن **ok** را می‌زنیم.

حال یک پایگاه داده ساخته شد با نامی که ما انتخاب کردیم، از ابتدا ما پایگاه داده‌ای نداریم و اگر بخواهیم در آن جدول ایجاد کنیم بر روی پایگاه داده کلیک راست کرده و **New Table** را می‌زنیم و در قسمت **Table name** باید نام جدولمان را

بنویسیم و بعد گزینه‌ی **add** را زده تا فیلدهایمان را اضافه کنیم، سپس یک کادر باز شده که در آن ما باید یکسری ویژگی‌ها را پر کنیم. به عنوان مثال اگر نام فیلد ما **id** بود حتماً باید نوع آن را **INTEGER** انتخاب کنیم چون نوع **int** را پشتیبانی نمی‌کند.

بعد از آن گزینه‌ی **ok** را می‌زنیم، به همین منوال فیلدهایمان را اضافه کرده، هر بار پس از اضافه کردن باید **Apply** را بزنیم.

نکته مهمی که وجود دارد این است که مثلاً اگر بخواهیم **id** شخصی برای یک قشر خاص انتخاب کنیم باید از نوع **nvarchar** باشد که بعدها ممکن است به دلیل زیاد شدن کاربران از یکسری حروف در **id** ها استفاده شود. (مانند پلاک خودرو)

تا این مرحله ما در قسمت **Design** بودیم و برای مشاهده‌ی فیلدها به قسمت **Data** می‌رویم.

برای مشخص کردن کلید اصلی باید بر روی **id** کلیک کرده و بعد روی گزینه‌ی **indexes** رفته و **Add** را می‌زنیم، یک کادر برای ما باز شده که باید در آن تیک گزینه‌های **primary, Unique, Autoincrement** را بزنیم و بعد از آن **ok** را می‌زنیم.

**Autoincrement**: به صورت خودکار اضافه بکن.

**Unique**: یک بار تکرار شود.

**primary**: از نوع کلید باشد.

بعد از این کار می‌توانیم کلیدهای خارجی را هم اضافه کنیم.

به همین ترتیب با کلیک بر روی علامت **+** می‌توانیم مقداره‌ی به فیلدها را شروع کنیم.

بعد از آن همه اطلاعات را ثبت کرده و از دیتابیس خارج شده.

حال بر روی فایل دیتابیس‌ی که ایجاد کردیم کلیک راست کرده و گزینه‌ی **copy** را می‌زنیم، سپس **eclipse** را باز کرده و یک پروژه‌ی جدید بانام **LoginWhitDatabase** می‌سازیم، برای ادامه‌ی کار ما از کدهایی که در پروژه‌ی **LoginProject** نوشته‌ایم استفاده کرده و فقط می‌خواهیم به جای ۲ مقدار ثابتی (یوزر و پسورد) که به دلیل نداشتن دیتابیس در آن وارد کرده بودیم را پاک کرده و این بار از پایگاه داده یوزر و پسورد بگیریم که بتوان برای چندین کاربر استفاده شود.

در اصل باید دو جمله‌ی زیر را از قسمت **LoginActivity.java** پاک کرد و از پایگاه داده اطلاعات معتبر را بگیریم.

```
String str_user_motabar ;
```

```
String str_pass_moatabar;
```

برای اینکه از دیتابیس اطلاعات بگیریم باید یک کلاس داشته باشیم که نام آن `DBOpenHelper.java` باشد، به همین دلیل دیتابیس در قسمت `src` و قسمت `comdatabase.loginwhitdatabase` ، و قسمت `DBOpenHelper.java` کپی کنیم، باید دقت داشته باشیم که نام پکیج ما در تمام اکتوییتی ها باهم یکسان باشد.

در این قسمت ما یک کلاس با نام `DBOpenHelper` نوشته ایم که آن را `extends` کرده ایم از `SQLiteOpenHelper` ، ماهیت جاوا دارد و مانند سرویس نیست که در پس زمینه ایجاد شود و یا مانند اکتوییتی نیست که به ما نشان دهد و یا مانند `Broadcast` نیست که وظیفه ای را سر ساعت خاصی انجام دهد.

در اصل کلاسی از که ارث بری از `SQLiteOpenHelper` کرده ایم.

از قبل باید پایگاه داده مان را در `asset` کپی کنیم.

ما در اینجا یکسری متغیر تعریف می کنیم:

```
private static final int DATABASE_VERSION = 3; //ورژن اندروید آن می باشد
private static String DB_PATH = ""; //مسیر دیتابیس، آنجایی که قرار است کپی شود
private static String DB_NAME = "azmon.db"; //نام دیتابیس
private SQLiteDatabase mDataBase; //یک نوع می باشد
private final Context mContext; //یک نوع می باشد
```

بعد از آن یک تابع با نام `public DBHelper(Context context)` تعریف کرده.

این تابع را برای این تعریف کرده ایم که می خواهیم به دیتابیس مان بگوییم که در چه مسیری بساز.

مسیر دیتابیس در پروژه های اندرویدی ثابت است مثلاً می رود و روی پوشه ی خاصی می گذارد.

و شرط زیر را بررسی می کند: ( این قسمت همیشه ثابت می باشد)

```
public DBHelper(Context context)
{
    super(context, DB_NAME, null, DATABASE_VERSION);
    if (android.os.Build.VERSION.SDK_INT >= 17)
    {
        DB_PATH = context.getApplicationInfo().dataDir + "/databases/";
        //مسیر دیتابیس اگر اندروید از ۱,۷ بالاتر بود //com.azmoon.ir
    }
    Else
    {
        DB_PATH = "/data/data/" + context.getPackageName() + "/databases/";
    }
}
```

```
} اگر اندروید ما کمتر از ۱،۴ بود مسیر ما به این صورت است //  
    this.mContext = context; }
```

حال با تابع `public void createDataBase() throws IOException` دیتابیس خودمان را ساخته.

همیشه کلاس دیتابیس ما ثابت است به جز قسمت‌هایی که در حین کار گفته می‌شود.

این جمله `boolean mDataBaseExist = checkDataBase();` می‌گوید که آیا دیتابیس هست یا نه برای همین کار باید قبل از آن یک تابع با نام `private boolean checkDataBase()` نوشته شود که برای بار اول که ما دیتابیس را اضافه می‌کنیم بررسی کند که آیا دیتابیس هست یا نیست.

```
private boolean checkDataBase()  
{  
    File dbFile = new File(DB_PATH + DB_NAME); //مسیر و نام  
    return dbFile.exists();  
}
```

هر جوابی از این تابع به دست آید به تابع `createDataBase()` می‌رود:

```
public void createDataBase() throws IOException  
{  
  
    boolean mDataBaseExist = checkDataBase();  
    if (!mDataBaseExist) // اگر دیتابیس وجود نداشته باشد  
    {  
        this.getReadableDatabase(); // دیتابیس را از آسیت بخوان  
        this.close(); // جریان را ببند چون با رفتن به آسیت یک جریان ایجاد شده  
        try // تلاش کن  
        {  
            copyDataBase(); // اجرای تابع کپی دیتابیس  
        }  
        catch (IOException mIOException)  
        {  
            throw new Error("ErrorCopyingDataBase");  
        }  
        // یک ساختار try-catch می‌باشد که اول می‌گوید برای کاری تلاش کن و حالا اگر نتوانستی ارور را چاپ بکن  
    }  
}
```

بعد از آن به سراغ تابع `copyDataBase` رفته و در زیر آن را بررسی می‌کنیم:

```
private void copyDataBase() throws IOException // چون خروجی ندارد از این نوع است  
{
```

اینجا دیگر در `InputStream mInput = mContext.getAssets().open(DB_NAME);`  
 دیتابیس را دارد و آن را برداشته و درجایی قرار می دهد //

```
String outFileName = DB_PATH + DB_NAME;
OutputStream mOutput = new FileOutputStream(outFileName);
// چیزی را که برداشته باید برود و درون همان فولدری که قرار بوده بگذارد
byte[] mBuffer = new byte[1024]; // آرایه‌ای از بافر بساز
int mLength;
while ((mLength = mInput.read(mBuffer)) > 0)
    {
        mOutput.write(mBuffer, 0, mLength);
    }
mOutput.flush(); // کار را تمام کن
mOutput.close();
mInput.close();
}
```

اگر هنوز بافر دارد داده می گیرد بگیر تا صفر شود //

بعد از آن یک تابع `openDataBase` داریم که زمانی که دیتابیس را سر جایش قرارداد از این تابع استفاده کرده و آن را باز می کند.

```
public SQLiteDatabase openDataBase() throws SQLException
{
    String mPath = DB_PATH + DB_NAME;
    SQLiteDatabase mDataBase = SQLiteDatabase.openDatabase(mPath, null,
        SQLiteDatabase.CREATE_IF_NECESSARY);
    return mDataBase;
} // در اصل به آن گفته می شود که دیتابیس را باز کن و برای ما برگردان //
```

تابع `onUpgrade` به شرح زیر عمل می کند:

این تابع برای زمانی است که ما یک محصولی را تولید کرده ایم و چند جدول داشته و بعد ورژن بالاتر آن را که تولید کرده ایم جداولش بیشتر شده و کار آن این است که می گوید برو آن جداول قبلی را پاک کن و اطلاعات جدید را بگیر.

```
public void onUpgrade(SQLiteDatabase db, int ov, int nv)
{ db.execSQL("DROP TABLE IF EXISTS user_tabel");
  onCreate(db); }
// تابع synchronized زمانی که کار ما با دیتابیس تمام می شود، ارتباط با دیتابیس را می بندد.
```

```
public synchronized void close()
{
    if (mDataBase != null)
        mDataBase.close();
    super.close();
}
```



خلاصه:

این تابع `DBOpenHelper` بود که ما می‌توانستیم به پایگاه داده متصل شویم و بررسی کنیم که آیا پایگاه داده در مسیرش وجود دارد یا خیر، که اگر در مسیرش بود که عملی را انجام نمی‌دهیم و اگر در مسیر نبود باید آن را از آسیت برداریم و کپی کنیم در مسیری که باید وجود داشته باشد.  
کد کلی آن به صورت زیر می‌باشد:

```
public class DBOpenHelper extends SQLiteOpenHelper
{
    private static final int DATABASE_VERSION = 3;
    private static String DB_PATH = "";
    private static String DB_NAME = "azmon.db"; // در هر کلاسی باید این نام عوض شود
    private SQLiteDatabase mDataBase;
    private final Context mContext;

    public DBOpenHelper(Context context)
    {
        super(context, DB_NAME, null, DATABASE_VERSION);
        if (android.os.Build.VERSION.SDK_INT >= 17)
        {
            DB_PATH = context.getApplicationInfo().dataDir + "/databases/";
            //com.azmoon.ir
        }
        else
        {
            DB_PATH = "/data/data/" + context.getPackageName() +
            "/databases/";
        }
        this.mContext = context;
    }

    public void createDataBase() throws IOException
    {
        boolean mDataBaseExist = checkDataBase();
        if (!mDataBaseExist)
        {
            this.getReadableDatabase();
            this.close();
            try
            {
                {
                    copyDataBase();
                }
            }
            catch (IOException mIOException)
            {
                throw new Error("ErrorCopyingDataBase");
            }
        }
    }

    private boolean checkDataBase()
    {
        File dbFile = new File(DB_PATH + DB_NAME);
        return dbFile.exists();
    }
}
```

```

}

private void copyDataBase() throws IOException
{
    InputStream mInput = mContext.getAssets().open(DB_NAME);
    String outFileName = DB_PATH + DB_NAME;
    OutputStream mOutput = new FileOutputStream(outFileName);
    byte[] mBuffer = new byte[1024];
    int mLength;
    while ((mLength = mInput.read(mBuffer)) > 0)
    {
        mOutput.write(mBuffer, 0, mLength);
    }
    mOutput.flush();
    mOutput.close();
    mInput.close();
}

public SQLiteDatabase openDataBase() throws SQLException
{
    String mPath = DB_PATH + DB_NAME;
    SQLiteDatabase mDataBase = SQLiteDatabase.openDatabase(mPath, null,
SQLiteDatabase.CREATE_IF_NECESSARY);
    return mDataBase;
}

@Override
public void onUpgrade(SQLiteDatabase db, int ov, int nv)
{
    db.execSQL("DROP TABLE IF EXISTS user_tabel");

    onCreate(db);
}

@Override
public synchronized void close()
{
    if (mDataBase != null)
        mDataBase.close();
    super.close();
}

@Override
public void onCreate(SQLiteDatabase arg0)
{
}
}

```

// نام پایگاه داده‌ای که افزودیم

## ادامه‌ی پروژه‌ی SQLite

ما باید یک کلاس DataSource.java را اضافه کردیم.

زمانی که بخواهیم تراکنشی در دیتابیس انجام بدهیم ( پاک کردن- اضافه کردن- به روزرسانی کردن- انتخاب کردن)، باید یک نوع از کلاس `DBOpenHelper.java` ایجاد کنیم چون یک نوع کلاس جاوا می باشد و از `SQLiteOpenHelper` ، `extends` شده است و دارای سازنده هست.

اگر سازنده کلاس ما اجرا شود به سراغ متدهایش رفته و می توانیم آن را صدا بزنیم و آن را اجراش کنیم.

حال در کلاس `DataSource.java` یک نوع از `DBOpenHelper dbHelper` ( ما ایجاد کردیم) و یک نوع از `SQLiteDatabase database` تعریف کرده ( `SQLiteDatabase database` از کلاس های خود اندروید می باشد). برای کلاس `DataSource.java` هم یک سازنده به کار می بریم.

```
public DataSource(Context c)
{
    dbHelper = new DBOpenHelper(c); // یک نوع از کلاس برایم ایجاد کن
}
```

ارث بری انجام شده است.

زمانی که بخواهیم یک شی تعریف کنیم باید به صورت زیر باشد ( مثلاً برای کلاس `DBOpenHelper`)

اسم متغیری که می خواهیم تعریف کنیم = `new` + اسم آن کلاس ( نام سازنده);

با دستورات زیر سعی بر آن داریم که دیتابیس را باز کنیم،

```
public void open()
{
    try
    {
        dbHelper.createDataBase();
    }
}
```

اگر دکمه ی کنترل را بگیریم و روی آن کلیک کنیم به سراغ `DBOpenHelper` رفته و تابع `createDataBase` را باز کرده و یعنی نشان داده که ارث بری می کند.

و اگر بر روی آن کلیک کنیم آن را اجرا کرده.

```
database = dbHelper.openDataBase();
```

زمانی که همیشه نام یک شی را می‌نویسد و اسم یک تابع را می‌نویسد و بعد از آن = گذاشته و درون چیزی ریخته است باید متوجه شویم که تابع ما `void` نیست و نوعی را برمی‌گرداند، برای مثال در این قسمت با این کار تابع `DBOpenHelper` یک نوع برای ما برگردانده از نوع `SQLiteDatabase` که ما آن را در قسمت `DataSource.java` به فرم `SQLiteDatabase database` تعریف کرده‌ایم.

ما آن نوع را ساختیم و مقدار را درونش ریختیم.

```
}
catch (IOException e) // اگر نتوانستی، ارور را نشان بده
{
    e.printStackTrace(); } }
```

در آخر هر دیتابیس باید آن را بست، پس ما با کد زیر تابع بستن را در `DBOpenHelper` فراخوانی می‌کنیم.

```
public void close()
{
    dbHelper.close();
}
```

بعد از این قسمت‌ها می‌خواهیم یکسری عملیات را انجام بدهیم.

مثلاً اگر بخواهیم نام کاربری و رمز عبور را از پایگاه داده بخوانیم و بعد بانام کاربری و رمز عبوری که کاربر وارد می‌کند مقایسه کنیم باید کد زیر را بنویسیم.

ابتدا یک تابع از نوع `public` ایجاد کرده که باید از نوع `boolean` باشد و نام کلاس را `User_setificate`

انتخاب کند که یوزر و پسورد را بگیرد و مقایسه نماید.

بعد از آن دیتابیس را باز کرده و خودش به سراغ تابع `open` می‌رود. (ارث‌بری)

```
public boolean User_setificate(String user_feli, String pass_feli) {
    open();
```

یک‌لایه از ارث‌بری داریم که همان `presentation` است که می‌بریم پشت اکتیویتی (همان‌جا که می‌بریم و چک می‌کنیم) که آنجا هم باید یک نوع از دیتاسورس بسازیم و مثلاً نوع آن را که ساختیم اسم آن را `ds` بگذاریم و بعد از آن نقطه بگذاریم و در مرحله‌ی بعد در داخل نقطه‌اش کلاس `User_setificate` می‌آید و بعد در `User_setificate`، پسورد و یوزری که کاربر می‌دهد را به آن بدهیم و بعد با یک `if` نتیجه را چک کنیم و اگر `Boolean` نتیجه را `false` داد یعنی کاربر رمز ورود خود را

اشتباه وارد کرده است در غیر این صورت درست وارد کرده است. حال باید یک **Cursor** ( عمل کننده یک تراکنش در پایگاه داده هست که مثلاً زمانی که بخواهیم یک نتیجه‌ای را برگردانیم باید در داخل نوعی از **Cursor** بریزیم) تعریف کنیم

```
Cursor cur = database.rawQuery("SELECT * FROM user_tabel WHERE User_name = "+user_feli+"AND Pass="+pass_feli, null);
```

یک ورودی از نوع عملیات پایگاه داده‌ای به آن داده‌ایم، انتخاب کن از جدول **user\_tabel** همه فیلدهایش را که یوزرنیم برابر با یوزر فعلی باشد و گفتیم هم‌زمان هم چک کند که پسورد برابر با پس فعلی باشد.

که بعد از این کار نتیجه را در داخل **Cursor** می‌ریزد.

بعد از این کار با یک **if** چک می‌کند که تعداد سطری که **Cursor** برگردانده را می‌شمارد و اگر بزرگ‌تر از صفر بود **true** را برگردان و تابع **Boolean** نتیجه را گرفته و می‌رود بر سر تابع لاگین و وارد می‌شود.

در غیر این صورت **false** را برگردان.

**Cursor** یک سری سطر برمی‌گرداند که ما با نوشتن **getCount** تعداد سطرهای برگردانده شده را می‌دهد، اگر صفر را برگرداند یعنی اشتباه بوده و لاگین نمی‌شود اما در غیر این صورت لاگین می‌شود.

```
if (cur.getCount() > 0)
{
    String str = cur.getString(cur.getColumnIndex("Pass"));
    Log.d("hh", str);
    return true;
}
else {
    return false;
}
```

حال در کل کد ما در قسمت عملیات به صورت زیر می‌باشد.

```
public boolean User_setificate(String user_feli, String pass_feli) {
    open();
    Cursor cur = database.rawQuery("SELECT * FROM user_tabel WHERE User_name = '"+user_feli+"'AND Pass='"+pass_feli+"'", null);
    if (cur.getCount() > 0)
    {
        bool1 = true;
    }
}
```

```

    }
    else{
        bool1 = false;
    }
    cur.close();
    close();
    return bool1;
}
}
}

```

تا این قسمت کارهای مربوط به دیتابیس انجام شده است و حال باید به سراغ LoginActivity.java برویم و یک نوع از دیتابیس به صورت سراسری تعریف کنیم.

```
DataSource ds;
```

این نوع از آن را ساخته و بعد باید در قسمتی که آن را کار داریم شیء ای از آن بسازیم، که ما در قسمت onClick آن را نوشته و یک نوع از Boolean تعریف کرده و بعد باید آن را اعتبارسنجی کرد.

```
ds = new DataSource(LoginActivity.this);
```

```
Boolean etebar_sanji;
```

حال در این قسمت کد ما به صورت زیر خواهد شد:

```

@Override
    public void onClick(View arg0) {
        // TODO Auto-generated method stub

        str_user_feli = edt1.getText().toString();
        str_pass_feli = edt2.getText().toString();
        ds = new DataSource(LoginActivity.this);
        etebar_sanji = ds.User_setificate(str_user_feli,
str_pass_feli);
        if(etebar_sanji){

            Intent i = new Intent();
            i.setClass(LoginActivity.this,
Activity_safe_asli.class);
            i.putExtra("nam",str_user_feli);
            startActivity(i);

        }
        else
        {
            Toast.makeText(getApplicationContext(), "رمز یا و کاربری نام"
است اشتباه عبور", Toast.LENGTH_SHORT).show();
        }
    }
}

```

```

    }
    });

```

همان‌طور که می‌بینید محتویات شرط ما به دلیل وجود پایگاه داده تغییر کرد.

کد کلی در قسمت DataSource.java به صورت زیر می‌باشد:

```

public class DataSource
{
    DBOpenHelper dbHelper;
    SQLiteDatabase database;

    public DataSource(Context c)
    {
        dbHelper = new DBOpenHelper(c);
    }

    public void open()
    {
        try
        {
            dbHelper.createDataBase();
            database = dbHelper.openDataBase();
        }
        catch (IOException e)
        {
            e.printStackTrace();
        }
    }

    public void close()
    {
        dbHelper.close();
    }

    public boolean User_setificate(String user_feli,String pass_feli){

        open();
        Cursor cur = database.rawQuery("SELECT * FROM user_tabel WHERE
User_name = '"+user_feli+"'AND Pass='"+pass_feli+"'", null);

        if(cur.getCount()>0)
        {

            bool1 = true;
        }
        else{
            bool1 = false;
        }
        cur.close();
        close();
        return bool1;
    }
}

```

```
}
```

کد نهایی در قسمت LoginActivity.java با تغییرات اعمال شده نسبت به اینکه با دیتابیس می‌باشد به صورت زیر است:

```
public class LoginActivity extends Activity {
    EditText edt1,edt2;
    Button btn1;
    String str_user_motabar ;
    String str_pass_moatabar;
    String str_user_feli;
    String str_pass_feli;
    DataSource ds;
    Boolean etebar_sanji;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_login);
        edt1 = (EditText)findViewById(R.id.edt1);
        edt2 = (EditText)findViewById(R.id.edt2);
        btn1 = (Button)findViewById(R.id.btn1);
        ds = new DataSource(this);
        btn1.setOnClickListener(new View.OnClickListener() {

            @Override
            public void onClick(View arg0) {
                // TODO Auto-generated method stub

                str_user_feli = edt1.getText().toString();
                str_pass_feli = edt2.getText().toString();
                ds = new DataSource(LoginActivity.this);
                etebar_sanji = ds.User_setificate(str_user_feli,
str_pass_feli);

                if(etebar_sanji){
                    Intent i = new Intent();
                    i.setClass(LoginActivity.this,
Activity_safe_asli.class);
                    i.putExtra("nam",str_user_feli);
                    startActivity(i);
                }
                else
                {
                    Toast.makeText(getApplicationContext(), "رمز یا و کاربری نام"
رمز یا و کاربری نام", Toast.LENGTH_SHORT).show();
                }
            }
        });
    }
}
```

## تراکنش Insert

این تراکنش مانند زمانی است که ثبت نام می‌کنیم.



در قسمت LoginActivity.java ، باید نام فایل xml آن را ببینیم و آن را باز نماییم.

از قسمت res ، قسمت activity\_login.xml را باز می‌کنیم.

ما می‌خواهیم دکمه‌ای با نام ثبت‌نام ایجاد کنیم که اگر بر روی آن کلیک شود، صفحه‌ای باز شود و یک نام کاربری و رمز عبور را دریافت کند و اگر کاربری با آن مشخصات ثبت‌نشده قبلاً، آن را ثبت نماید در غیر این صورت ارور را نمایش دهد.

دکمه‌ای با نام Button را با مشخصات زیر در صفحه‌ی activity\_login.xml ایجاد می‌کنیم.

فعالاً چون به صورت آفلاین عمل می‌کنیم، آن را به پایگاه داده وصل کرده.

```
<Button
    android:id="@+id/btn_register"
    android:layout_height="50dp"
    android:layout_width="250dp"
    android:layout_below="@+id/btn1"
    android:layout_centerHorizontal="true"
    android:layout_marginTop="20dp"
    android:background="#f30f30"
    android:text="ثبت‌نام" />
```

حالا باید ثبت‌نام را در LoginActivity.java صدا بزنیم، پس به این ترتیب یک دکمه تعریف می‌کنیم.

```
Button btn_register ;
```

و بعدازآن در قسمت onCreate ، آن را تعریف کرده و سپس آن را حساس به کلیک کرده.

```
btn_register = (Button) findViewById(R.id.btn_register);
btn_register.setOnClickListener(new View.OnClickListener()
```

زمانی که آن را حساس به کلیک کردیم، در قسمتی که باز می‌شود باید کد زیر را وارد کرد.

```
btn_register.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View arg0) {
        // TODO Auto-generated method stub
        Intent i = new Intent();
        i.setClass(LoginActivity.this, Activity_Register.class);
        startActivity(i);
    }
});
```

در اصل این قطعه کد بیان می‌کند که از LoginActivity به یک اکتیویته دیگر برای ثبت‌نام برود. پس ما باید آن اکتیویته را بسازیم، پس یک کلاس جدید ساخته و نام آن را Activity\_Register.java می‌گذاریم.

بعدازآن باید به `AndroidManifest.xml` برویم و اجازهی `Activity_Register` را صادر کنیم.

باید `xml` آن را بانام `register_activity.xml` بسازیم و در آن کدهای زیر را وارد نماییم.

در اصل این کدها برای شمای گرافیکی این صفحه نوشته می شوند.

باید ۲ تا `EditText` ایجاد کنیم.

```
<RelativeLayout
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    xmlns:android="http://schemas.android.com/apk/res/android">

    <EditText
        android:id="@+id/edt_register_1"
        android:layout_width="match_parent"
        android:layout_height="50dip"
        android:hint="UserName"
    />

    <EditText
        android:id="@+id/edt_register_2"
        android:layout_width="match_parent"
        android:layout_height="50dip"
        android:layout_below="@+id/edt_register_1"
        android:hint="Password"
        android:inputType="textPassword"
    />

    <Button
        android:id="@+id/btn_register"
        android:layout_width="match_parent"
        android:layout_height="60dip"
        android:text="ثبت اطلاعات"
        android:layout_below="@+id/edt_register_2"
    />

</RelativeLayout>
```

حالا به سراغ نوشتن کد آن در قسمت `Activity_Register.java` می رویم و با دانسته های پیشین جلو رفته و کد آن را

می نویسیم. (همیشه در کد جاوا متد `onCreate` ثابت می باشد).

باید دکمه هایی را که ایجاد کردیم، تعریف کنیم.

در قسمت ثبت کردن یعنی اینکه می‌خواهیم دسته‌ای از اطلاعات را ببریم سمت دیتابیس، برای این کار مجبوریم که یک پل داده داشته باشیم که نام آن پل داده را **data object** (شیء داده) گذاشته که مثلاً ۶ نوع داده را در خود نگه می‌دارد.

ابتدا یک کلاس جاوا با نام **DataObject.java** می‌سازیم (مانند قبل می‌سازیم)

حال در این کلاس بیان می‌کنیم که این پلی که ایجاد کرده‌ایم واسط چه چیزهایی قرار می‌گیرد، که ما در این برنامه می‌خواهیم واسط یوزرنیم و پسورد قرار بگیرد که از نوع **string** هست. (می‌خواهیم رشته‌ی یوزر و پسورد را با یک واسطه داخل دیتابیس قرار بدهیم)

```
// public long id;  
public String username ;  
public String pass;
```

از نوع سراسری قرار دادیم تا از قسمت‌های دیگر هم به آن‌ها دسترسی داشته باشیم.

نکته‌ای که وجود دارد این است که این کلاس‌ها نیازی به ثبت شدن در **AndroidManifest.xml** ندارند چون اکتیویتی نیستند.

حال یک سازنده برای این کلاس می‌نویسیم و می‌گوییم هر متغیری وارد این کلاس شد و یک مقدار به **id** اختصاص داد کارهای زیر را انجام بده.

```
public DataObject(String username,String pass){
```

```
//    this.id = id;  
    this.username = username;  
    this.pass = pass;
```

چیزهایی که ورودی تابع است.

```
}
```

**This** = یعنی چیزهایی که مربوط به همین کلاس است. (این کلاس یک جایی از اکتیویتی صدا زده می‌شود و این پارامترها به آن داده می‌شود. کلاس زمانی که ساخته می‌شود، تابع سازنده آن ساخته می‌شود و دیتا آبجکت سه موردی که در آنجا وجود دارد را در سه پارامتر این کلاس قرار داده.

قسمت زیر بیان می‌کند که هر مقداری را که به تو دادیم چون که می‌خواهی به دیتابیس بدهی، آن را به صورت رشته پس بده.

```
@Override  
public String toString()  
{  
    return "";
```

```
}
```

در قسمت بعدی به سراغ کلاس `DataSource.java` می‌رویم و تابعی که قرار است عمل `Insert` را انجام دهد، می‌نویسیم.

برای این کار یک تابع می‌نویسیم، این تابع نیازی به خروجی ندارد چون عمل افزودن را انجام می‌دهد.

برای ورودی تابع باید یک نوع از ساختارمان را بنویسیم.

```
public void Insert_User(DataObject info){

    open();
    Cursor cur = database.rawQuery("INSERT INTO      user_tabel(User_name,Pass)
VALUES ('"+info.username+"','"+info.pass+"'), null);
    cur.moveToFirst();
    cur.close();
    close();
}
```

برای اینکه بخواهیم نام تکراری ثبت نشود باید با یک شرط آن را چک کنیم و یک تابع جداگانه‌ای بنویسیم که کار `selct` را انجام دهد و ببینیم که چه مقداری را به ما برمی‌گرداند که اگر `false` را برای ما برگرداند، یعنی اینکه این نام قبلاً ثبت نشده و آن را ثبت کند، در غیر این صورت با یک پیغام به کاربر اطلاع دهد.

تابع آن به صورت زیر می‌باشد:

```
public boolean chek_user(String user){

    open();
    boolean natije;
    Cursor cur = database.rawQuery("SELECT *FROM user_tabel WHERE
User_name='"+user+"'", null);
    if (cur.getCount(>0){ // شمارنده
        natije = true;
    }
    else
    {
        natije = false;
    }
    cur.close();
    close();
    return natije;
}
```

در قسمت `DataSource.java` کد نهایی به صورت زیر می‌شود:

```

public class DataSource
{
    DBOpenHelper dbHelper;
    SQLiteDatabase database;

    public DataSource(Context c)
    {
        dbHelper = new DBOpenHelper(c);
    }

    public void open()
    {
        try
        {
            dbHelper.createDataBase();
            database = dbHelper.openDataBase();
        }
        catch (IOException e)
        {
            e.printStackTrace();
        }
    }

    public void close()
    {
        dbHelper.close();
    }

    public boolean User_setificate(String user_feli,String pass_feli){

        open();
        boolean bool1;
        Cursor cur = database.rawQuery("SELECT * FROM user_tabel WHERE
User_name='"+user_feli+"'AND Pass='"+pass_feli+"'", null);

        if(cur.getCount()>0)
        {
            bool1 = true;
        }
        else{
            bool1 = false;
        }
        cur.close();
        close();
        return bool1;
    }

    public void Insert_User(DataObject info){

        open();
        Cursor cur = database.rawQuery("INSERT INTO user_tabel
(User_name,Pass) VALUES ('"+info.username+"', '"+info.pass+"')", null);
        cur.moveToFirst();
        cur.close();
        close();
    }
}

```

```

}

public boolean chek_user(String user){

    open();
    boolean natije;
    Cursor cur = database.rawQuery("SELECT *FROM user_tabel WHERE
User_name='"+user+"' , null);
    if (cur.getCount()>0){

        natije = true;
    }
    else
    {
        natije = false;
    }
    cur.close();
    close();
    return natije;
}
}

```

حال به سراغ قسمت Activity\_Register.java رفته و تمام دکمه‌ها را به صورت زیر تعریف کرده :

```

EditText edt_username_register,edt_pass_register;
Button btn_reg;

```

و سپس آن‌ها را صدا زده و آن‌ها را می‌شناسانیم:

```

edt_username_register = (EditText) findViewById(R.id.edt_register_1);
edt_pass_register = (EditText) findViewById(R.id.edt_register_2);
btn_reg = (Button) findViewById(R.id.btn_register);

```

دکمه را حساس به کلیک کرده.

```

btn_reg.setOnClickListener(new View.OnClickListener())

```

قبل از آنکه کد زیر نوشته شود باید یک نوع از دیتاسورس و یک نوع از دیتا آبجکت ساخته شود.

حال باید به صورت زیر کدی نوشته شود که اطلاعات را وارد دیتابیس بکند:

```

@Override
public void onClick(View arg0) {
    // TODO Auto-generated method stub
    ds = new DataSource(Activity_Register.this);

    if(!ds.chek_user(edt_username_register.getText().toString())){

```

// DataSource.java است از chek\_user ارث بری کرده است

```

        ds2 = new DataSource(Activity_Register.this);
        ds2.Insert_User(new
        DataObject(edt_username_register.getText().toString(),
        edt_pass_register.getText().toString()));
        Toast.makeText(getApplicationContext(), "کاربر با موفقیت ثبت نام شد",
        Toast.LENGTH_SHORT).show();
    }
    else
    {
        Toast.makeText(getApplicationContext(), "کاربر با این نام کاربری قبلاً ثبت نام شده است",
        Toast.LENGTH_SHORT).show();
    }
}
});

```

حال به سراغ قسمت AndroidManifest.xml می رویم و اجازه‌ی اکتیویتی را صادر کرده :

```

<activity
    android:name="comdatabase.loginwhitdatabase.Activity_Register"
    android:label="@string/app_name" >
</activity>

```

در این قسمت کار ما به اتمام رسیده است.

## آموزش تراکنش Insert و Update و Delete

### متد Insert

از این تراکنش برای ورود اطلاعات به پایگاه داده استفاده می شود برای انجام عملی آن به همان پروژه پایگاه داده مراجعه شود

سپس وارد فایل Datasource می شویم یک تابع بانام insertClient می نویسیم نوع خروجی را Void در نظر می گیریم چون قرار نیست تابع مقداری را برگرداند

سپس تراکنش را مطابق زیر می نویسیم :

```

public void insertClient(ListDataObject info)
{
    open();

    Cursor cur = database.rawQuery("INSERT INTO CLIENT
(FNAME,LNAME,SID,TEL) VALUES ('"+info.fname+"', '"+info.lname+"', '"+info.sid+"',
'"+info.tel+"')", null);

```

```

        cur.moveToFirst();

        cur.close();

        close();
    }

```

این تابع در جدول کلاینت مقادیرهای نام و نام خانوادگی و ای دی و تلفن را از دیتا ابجکت گرفته و در داخل بانک وارد می‌کند.

## متد Update

این متد برای ویرایش یک فیلد از قبل ثبت‌شده در پایگاه داده استفاده می‌شود به این منظور در کلاس **DataSource** یک تابع با نام **Update\_client** می‌نویسیم

```

public void updateClient(ListDataObject info)
{
    open; ()

    Cursor cur = database.rawQuery("UPDATE CLIENT SET FNAME =
'"+info.fname+"', LNAME = '"+info.lname+"', SID = '"+info.sid+"', TEL =
'"+info.tel+"' WHERE ID = '"+Long.toString(info.id)+"'", null);

    cur.moveToFirst; ()

    cur.close; ()

    close; ()
}

```

این تابع جدول **CLIENT** و فیلدهای نام و نام خانوادگی و تلفن را ویرایش می‌کند

## متد Delete

این متد برای پاک کردن یک رکورد از پایگاه داده می‌باشد تابع موردنظر مطابق زیر است این تابع را در کلاس **DataSource** می‌نویسیم

```

public void deleteClientById(long id)

```



```

    {
        open();

        Cursor cur = database.rawQuery("DELETE FROM CLIENT WHERE ID =
"+Long.toString(id), null);

        cur.moveToFirst();

        cur.close();

        close()
    }

```

## ارسال اطلاعات به سرور با استفاده از php,json

برای ارسال اطلاعات در برنامه به پایگاه داده ابتدا کدهای سمت اندروید را توضیح می‌دهیم و سپس سراغ کدهای سمت سرور می‌رویم

به این منظور برای ارسال اطلاعات کلاس ثابت JSON را به پروژه مطابق زیر وارد می‌کنیم

```

import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStream;
import java.io.InputStreamReader;
import java.io.UnsupportedEncodingException;
import java.util.List;

import org.apache.http.HttpEntity;
import org.apache.http.HttpResponse;
import org.apache.http.NameValuePair;
import org.apache.http.client.ClientProtocolException;
import org.apache.http.client.entity.UrlEncodedFormEntity;
import org.apache.http.client.methods.HttpPost;
import org.apache.http.impl.client.DefaultHttpClient;

```

```

import org.json.JSONException;
import org.json.JSONObject;

import android.util.Log;

public class JSONParser {
    static InputStream is = null;
    static JSONObject jsonObj = null;
    static String json = null;

    public JSONParser() {

    }

    public JSONObject getJSONFromUrl(String url, List<NameValuePair> params) {

        // Make HTTP connection
        try {

            DefaultHttpClient httpClient = new DefaultHttpClient();
            HttpPost httpPost = new HttpPost(url);
            httpPost.setEntity(new UrlEncodedFormEntity(params, "UTF-8"));

            HttpResponse httpResponse = httpClient.execute(httpPost);
            HttpEntity httpEntity = httpResponse.getEntity();
            is = httpEntity.getContent();

        } catch (UnsupportedEncodingException e) {
            e.printStackTrace();
        } catch (ClientProtocolException e) {

```

```

        e.printStackTrace();
    } catch (IOException e) {
        e.printStackTrace();
    }

    try {

        BufferedReader reader = new BufferedReader(new InputStreamReader(
            is, "UTF-8"), 16);
        StringBuilder sb = new StringBuilder();
        String line = null;
        while ((line = reader.readLine()) != null) {
            sb.append(line + "\n");
        }
        is.close();
        json = sb.toString();

    } catch (Exception e) {
    }

    try {
        jsonObj = new JSONObject(json.substring(1, json.length()));
    } catch (JSONException e) {

    }

    return jsonObj;
}

}

```

سپس در اکتیویتی موردنظر تابع زیر را می‌نویسیم این تابع به منظور ارسال نام کاربری و رمز عبور برای ثبت نام می‌باشد .

```

public JSONObject registerUser(String email, String password, String name) {
    List<NameValuePair> params = new ArrayList<NameValuePair>();
    params.add(new BasicNameValuePair("tag", register_tag));
    params.add(new BasicNameValuePair("user_email", email));
    params.add(new BasicNameValuePair("user_password", password));
    params.add(new BasicNameValuePair("user_name", name));
    JSONObject jsonObject = jsonParser.getJSONFromUrl(UrlAdrees, params);
    return jsonObject;
}

```

بعد از مقداردهی به تابع و اجرای تابع مقدارهای email و name و password به UrlAdrees ارسال میشود این مقدار سمت سرور با کد زیر دریافت می شود

Php?>

```

/*
 * Following code will get single product details
 * A product is identified by product id (pid)
 */

// array for JSON response
$response = array();

// include db connect class
require_once __DIR__ . '/db_connect.php';

// connecting to db
$db = new DB_CONNECT();

// check for post data
if (isset($_GET["pelak_id"])) {
    $pid = $_GET['pelak_id'];
}

```

```
// get a product from products table
$result = mysql_query("Insert to tbl_user (email,password,name)
value($email,$password,$name)");

}
?>
```

## نکات مهم

۱. زمانی که برای اجرای برنامه شبیه‌ساز را شناسد باید کارهای زیر را انجام بدهد:

بر روی آیکون ایکلیپس کلیک راست کرده و در قسمت **properties** ، گزینه‌ی **open File Location** را زده و در قسمت **adt-bundle-windows** را زده بر روی قسمت **sdk** کلیک کرده و در آن روی پوشه **platform-tools** ، **shift** را گرفته و کلیک راست کرده و گزینه‌ی **open command window** را زده، یک صفحه‌ی مشکی‌رنگ باز شده که در آن **adb Kill-server** را نوشته و سرور را قطع کرده و **enter** را زده و در خط بعد **adb devices** را نوشته، به این صورت شبیه‌ساز را می‌شناسد.

۲. برای مرتب کردن کدها همیشه می‌توان از دستور زیر استفاده کرد:

**Ctrl+A, Ctrl+I**

۳. اگر شبیه‌ساز انیمیشنی را که به اکتوییتی دادیم شناسد باید در خود شبیه‌ساز در قسمت تنظیمات بر روی قسمت **Manage Accounts** کلیک کرده روی گزینه‌های برنامه‌نویسی کلیک کرده و گزینه‌ی مقیاس انیمیشن انتقال را انتخاب کرده و انیمیشن را روشن کرده.

۴. یک‌لایه از ارث‌بری داریم که همان **presentation** است که می‌بریم پشت اکتوییتی (همان‌جا که می‌بریم و چک می‌کنیم) که آنجا هم باید یک نوع از دیتاسورس بسازیم و مثلاً نوع آن را که ساختیم اسم آن را **ds** بگذاریم و بعد از آن نقطه بگذاریم و در مرحله‌ی بعد در داخل نقطه‌اش کلاس **User\_setificate** می‌آید و بعد در **User\_setificate** ، پسورد و یوزری که کاربر می‌دهد را به آن بدهیم و بعد با یک **if** نتیجه را چک کنیم.

۵. برای ورودی تابع دادن همیشه باید کوتیشن را ببندیم و مقدار ورودی تابع را بین دو تا + بگذاریم تا به صورت داینامیک عمل نماید.