

Chapter 1. First steps

Table of Contents

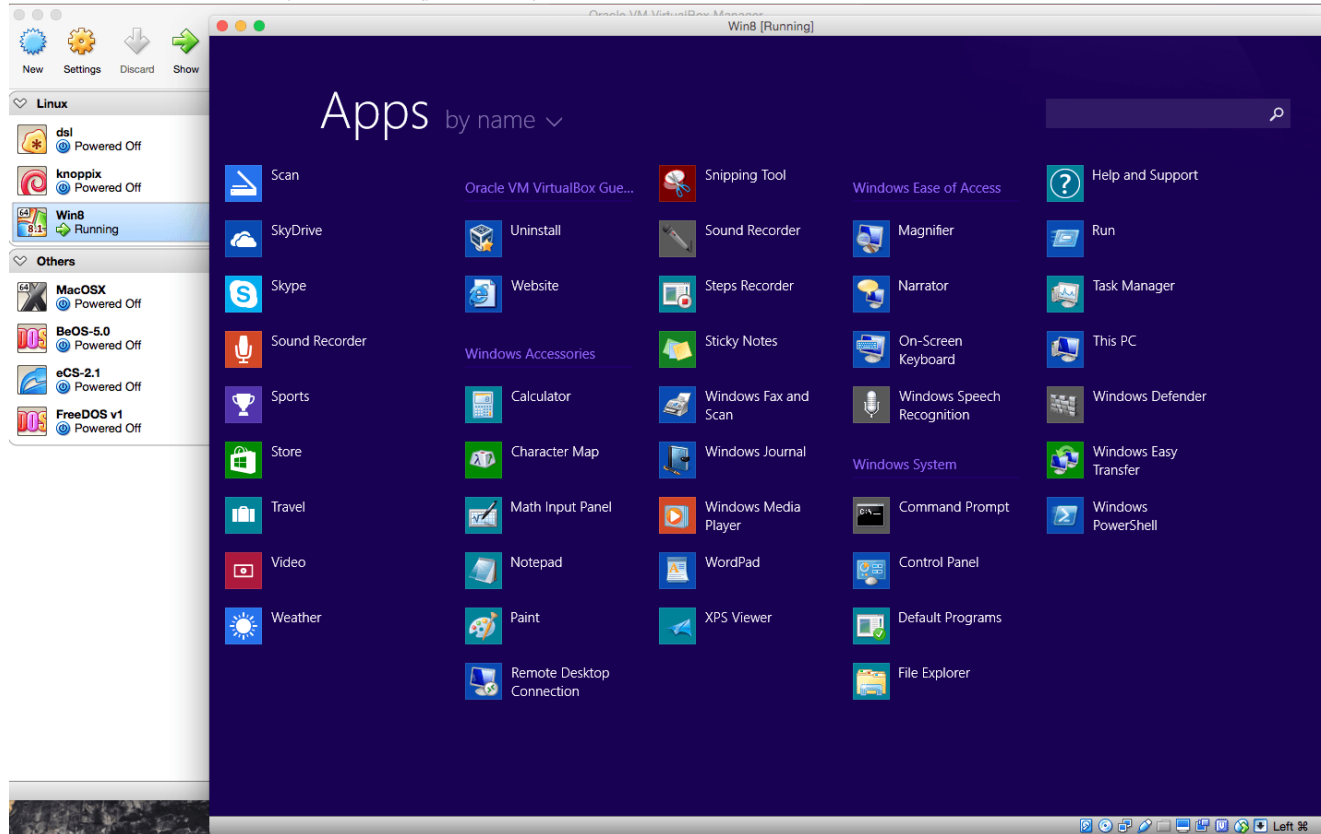
- [1.1. Why is virtualization useful?](#)
- [1.2. Some terminology](#)
- [1.3. Features overview](#)
- [1.4. Supported host operating systems](#)
- [1.5. Installing VirtualBox and extension packs](#)
- [1.6. Starting VirtualBox](#)
- [1.7. Creating your first virtual machine](#)
- [1.8. Running your virtual machine](#)
 - [1.8.1. Starting a new VM for the first time](#)
 - [1.8.2. Capturing and releasing keyboard and mouse](#)
 - [1.8.3. Typing special characters](#)
 - [1.8.4. Changing removable media](#)
 - [1.8.5. Resizing the machine's window](#)
 - [1.8.6. Saving the state of the machine](#)
- [1.9. Using VM groups](#)
- [1.10. Snapshots](#)
 - [1.10.1. Taking, restoring and deleting snapshots](#)
 - [1.10.2. Snapshot contents](#)
- [1.11. Virtual machine configuration](#)
- [1.12. Removing virtual machines](#)
- [1.13. Cloning virtual machines](#)
- [1.14. Importing and exporting virtual machines](#)
- [1.15. Global Settings](#)
- [1.16. Alternative front-ends](#)

Welcome to Oracle VM VirtualBox!

VirtualBox is a cross-platform virtualization application. What does that mean? For one thing, it installs on your existing Intel or AMD-based computers, whether they are running Windows, Mac, Linux or Solaris operating systems. Secondly, it extends the capabilities of your existing computer so that it can run multiple operating systems (inside multiple virtual machines) at the same time. So, for example, you can run Windows and Linux on your Mac, run Windows Server 2008 on your Linux server, run Linux on your Windows PC, and so on, all alongside your existing applications. You can install and run as many virtual machines as you like -- the only practical limits are disk space and memory.

VirtualBox is deceptively simple yet also very powerful. It can run everywhere from small embedded systems or desktop class machines all the way up to datacenter deployments and even Cloud environments.

The following screenshot shows you how VirtualBox, installed on a Mac computer, is running Windows 8 in a virtual machine window:



In this User Manual, we'll begin simply with a quick introduction to virtualization and how to get your first virtual machine running with the easy-to-use VirtualBox graphical user interface. Subsequent chapters will go into much more detail covering more powerful tools and features, but fortunately, it is not necessary to read the entire User Manual before you can use VirtualBox.

You can find a summary of VirtualBox's capabilities in [Section 1.3, "Features overview"](#). For existing VirtualBox users who just want to see what's new in this release, there is a detailed list in [Chapter 15, Change log](#).

1.1. Why is virtualization useful?

The techniques and features that VirtualBox provides are useful for several scenarios:

- **Running multiple operating systems simultaneously.** VirtualBox allows you to run more than one operating system at a time. This way, you can run software written for one operating system on another (for example, Windows software on Linux or a Mac) without having to reboot to use it. Since you can configure what kinds of "virtual" hardware should be presented to each such operating system, you can install an old operating system such as DOS or OS/2 even if your real computer's hardware is no longer supported by that operating system.
- **Easier software installations.** Software vendors can use virtual machines to ship entire software configurations. For example, installing a complete mail server solution on a real machine can be a tedious task. With VirtualBox, such a complex setup (then often called an "appliance") can be packed into a virtual machine. Installing and running a mail server becomes as easy as importing such an appliance into VirtualBox.
- **Testing and disaster recovery.** Once installed, a virtual machine and its virtual hard disks can be considered a "container" that can be arbitrarily frozen, woken up, copied, backed up, and transported between hosts.

On top of that, with the use of another VirtualBox feature called "snapshots", one can save a particular state of a virtual machine and revert back to that state, if necessary. This way, one can freely experiment with a computing environment. If something goes wrong (e.g. after installing misbehaving software or infecting the guest with a virus), one can easily switch back to a previous snapshot and avoid the need of frequent backups and restores.

Any number of snapshots can be created, allowing you to travel back and forward in virtual machine

time. You can delete snapshots while a VM is running to reclaim disk space.

- **Infrastructure consolidation.** Virtualization can significantly reduce hardware and electricity costs. Most of the time, computers today only use a fraction of their potential power and run with low average system loads. A lot of hardware resources as well as electricity is thereby wasted. So, instead of running many such physical computers that are only partially used, one can pack many virtual machines onto a few powerful hosts and balance the loads between them.

1.2. Some terminology

When dealing with virtualization (and also for understanding the following chapters of this documentation), it helps to acquaint oneself with a bit of crucial terminology, especially the following terms:

Host operating system (host OS).

This is the operating system of the physical computer on which VirtualBox was installed. There are versions of VirtualBox for Windows, Mac OS X, Linux and Solaris hosts; for details, please see [Section 1.4, "Supported host operating systems"](#).

Most of the time, this User Manual discusses all VirtualBox versions together. There may be platform-specific differences which we will point out where appropriate.

Guest operating system (guest OS).

This is the operating system that is running inside the virtual machine. Theoretically, VirtualBox can run any x86 operating system (DOS, Windows, OS/2, FreeBSD, OpenBSD), but to achieve near-native performance of the guest code on your machine, we had to go through a lot of optimizations that are specific to certain operating systems. So while your favorite operating system *may* run as a guest, we officially support and optimize for a select few (which, however, include the most common ones).

See [Section 3.1, "Supported guest operating systems"](#) for details.

Virtual machine (VM).

This is the special environment that VirtualBox creates for your guest operating system while it is running. In other words, you run your guest operating system "in" a VM. Normally, a VM will be shown as a window on your computer's desktop, but depending on which of the various frontends of VirtualBox you use, it can be displayed in full screen mode or remotely on another computer.

In a more abstract way, internally, VirtualBox thinks of a VM as a set of parameters that determine its behavior. They include hardware settings (how much memory the VM should have, what hard disks VirtualBox should virtualize through which container files, what CDs are mounted etc.) as well as state information (whether the VM is currently running, saved, its snapshots etc.). These settings are mirrored in the VirtualBox Manager window as well as the `VBoxManage` command line program; see [Chapter 8, *VBoxManage*](#). In other words, a VM is also what you can see in its settings dialog.

Guest Additions.

This refers to special software packages which are shipped with VirtualBox but designed to be installed *inside* a VM to improve performance of the guest OS and to add extra features. This is described in detail in [Chapter 4, *Guest Additions*](#).

1.3. Features overview

Here's a brief outline of VirtualBox's main features:

- **Portability.** VirtualBox runs on a large number of 32-bit and 64-bit host operating systems (again,

see [Section 1.4, "Supported host operating systems"](#) for details).

VirtualBox is a so-called "hosted" hypervisor (sometimes referred to as a "type 2" hypervisor). Whereas a "bare-metal" or "type 1" hypervisor would run directly on the hardware, VirtualBox requires an existing operating system to be installed. It can thus run alongside existing applications on that host.

To a very large degree, VirtualBox is functionally identical on all of the host platforms, and the same file and image formats are used. This allows you to run virtual machines created on one host on another host with a different host operating system; for example, you can create a virtual machine on Windows and then run it under Linux.

In addition, virtual machines can easily be imported and exported using the Open Virtualization Format (OVF, see [Section 1.14, "Importing and exporting virtual machines"](#)), an industry standard created for this purpose. You can even import OVF files that were created with a different virtualization software.

- **No hardware virtualization required.** For many scenarios, VirtualBox does not require the processor features built into newer hardware like Intel VT-x or AMD-V. As opposed to many other virtualization solutions, you can therefore use VirtualBox even on older hardware where these features are not present. The technical details are explained in [Section 10.3, "Hardware vs. software virtualization"](#).
- **Guest Additions: shared folders, seamless windows, 3D virtualization.** The VirtualBox Guest Additions are software packages which can be installed *inside* of supported guest systems to improve their performance and to provide additional integration and communication with the host system. After installing the Guest Additions, a virtual machine will support automatic adjustment of video resolutions, seamless windows, accelerated 3D graphics and more. The Guest Additions are described in detail in [Chapter 4, Guest Additions](#).

In particular, Guest Additions provide for "shared folders", which let you access files from the host system from within a guest machine. Shared folders are described in [Section 4.3, "Shared folders"](#).

- **Great hardware support.** Among others, VirtualBox supports:
 - **Guest multiprocessing (SMP).** VirtualBox can present up to 32 virtual CPUs to each virtual machine, irrespective of how many CPU cores are physically present on your host.
 - **USB device support.** VirtualBox implements a virtual USB controller and allows you to connect arbitrary USB devices to your virtual machines without having to install device-specific drivers on the host. USB support is not limited to certain device categories. For details, see [Section 3.10.1, "USB settings"](#).
 - **Hardware compatibility.** VirtualBox virtualizes a vast array of virtual devices, among them many devices that are typically provided by other virtualization platforms. That includes IDE, SCSI and SATA hard disk controllers, several virtual network cards and sound cards, virtual serial and parallel ports and an Input/Output Advanced Programmable Interrupt Controller (I/O APIC), which is found in many modern PC systems. This eases cloning of PC images from real machines and importing of third-party virtual machines into VirtualBox.
 - **Full ACPI support.** The Advanced Configuration and Power Interface (ACPI) is fully supported by VirtualBox. This eases cloning of PC images from real machines or third-party virtual machines into VirtualBox. With its unique **ACPI power status support**, VirtualBox can even report to ACPI-aware guest operating systems the power status of the host. For mobile systems running on battery, the guest can thus enable energy saving and notify the user of the remaining power (e.g. in full screen modes).
 - **Multiscreen resolutions.** VirtualBox virtual machines support screen resolutions many times that of a physical screen, allowing them to be spread over a large number of screens attached

to the host system.

- **Built-in iSCSI support.** This unique feature allows you to connect a virtual machine directly to an iSCSI storage server without going through the host system. The VM accesses the iSCSI target directly without the extra overhead that is required for virtualizing hard disks in container files. For details, see [Section 5.10, "iSCSI servers"](#).
- **PXE Network boot.** The integrated virtual network cards of VirtualBox fully support remote booting via the Preboot Execution Environment (PXE).
- **Multigeneration branched snapshots.** VirtualBox can save arbitrary snapshots of the state of the virtual machine. You can go back in time and revert the virtual machine to any such snapshot and start an alternative VM configuration from there, effectively creating a whole snapshot tree. For details, see [Section 1.10, "Snapshots"](#). You can create and delete snapshots while the virtual machine is running.
- **VM groups.** VirtualBox provides a groups feature that enables the user to organize and control virtual machines collectively, as well as individually. In addition to basic groups, it is also possible for any VM to be in more than one group, and for groups to be nested in a hierarchy -- i.e. groups of groups. In general, the operations that can be performed on groups are the same as those that can be applied to individual VMs i.e. Start, Pause, Reset, Close (Save state, Send Shutdown, Poweroff), Discard Saved State, Show in fileSystem, Sort.
- **Clean architecture; unprecedented modularity.** VirtualBox has an extremely modular design with well-defined internal programming interfaces and a clean separation of client and server code. This makes it easy to control it from several interfaces at once: for example, you can start a VM simply by clicking on a button in the VirtualBox graphical user interface and then control that machine from the command line, or even remotely. See [Section 1.16, "Alternative front-ends"](#) for details.

Due to its modular architecture, VirtualBox can also expose its full functionality and configurability through a comprehensive **software development kit (SDK)**, which allows for integrating every aspect of VirtualBox with other software systems. Please see [Chapter 11, VirtualBox programming interfaces](#) for details.

- **Remote machine display.** The VirtualBox Remote Desktop Extension (VRDE) allows for high-performance remote access to any running virtual machine. This extension supports the Remote Desktop Protocol (RDP) originally built into Microsoft Windows, with special additions for full client USB support.

The VRDE does not rely on the RDP server that is built into Microsoft Windows; instead, it is plugged directly into the virtualization layer. As a result, it works with guest operating systems other than Windows (even in text mode) and does not require application support in the virtual machine either. The VRDE is described in detail in [Section 7.1, "Remote display \(VRDP support\)"](#).

On top of this special capacity, VirtualBox offers you more unique features:

- **Extensible RDP authentication.** VirtualBox already supports Winlogon on Windows and PAM on Linux for RDP authentication. In addition, it includes an easy-to-use SDK which allows you to create arbitrary interfaces for other methods of authentication; see [Section 7.1.5, "RDP authentication"](#) for details.
- **USB over RDP.** Via RDP virtual channel support, VirtualBox also allows you to connect arbitrary USB devices locally to a virtual machine which is running remotely on a VirtualBox RDP server; see [Section 7.1.4, "Remote USB"](#) for details.

1.4. Supported host operating systems

Currently, VirtualBox runs on the following host operating systems:

- **Windows** hosts:^[1]
 - Windows Vista SP1 and later (32-bit and 64-bit)
 - Windows Server 2008 (64-bit)
 - Windows Server 2008 R2 (64-bit)
 - Windows 7 (32-bit and 64-bit)
 - Windows 8 (32-bit and 64-bit)
 - Windows 8.1 (32-bit and 64-bit)
 - Windows 10 RTM build 10240 (32-bit and 64-bit)
 - Windows Server 2012 (64-bit)
 - Windows Server 2012 R2 (64-bit)

- **Mac OS X** hosts (64-bit):^[2]
 - 10.9 (Mavericks)
 - 10.10 (Yosemite)
 - 10.11 (El Capitan)
 - 10.12 (Sierra)

Intel hardware is required; please see [Chapter 14, *Known limitations*](#) also.

- **Linux** hosts (32-bit and 64-bit^[3]). Among others, this includes:
 - Ubuntu 12.04 to 16.10
 - Debian GNU/Linux 7 ("Wheezy") and 8 ("Jessie")
 - Oracle Enterprise Linux 5, Oracle Linux 6 and 7
 - Redhat Enterprise Linux 5, 6 and 7
 - Fedora Core / Fedora 6 to 25
 - Gentoo Linux
 - openSUSE 11.4 to 13.2

It should be possible to use VirtualBox on most systems based on Linux kernel 2.6 or 3.x using either the VirtualBox installer or by doing a manual installation; see [Section 2.3, "Installing on Linux hosts"](#). However, the formally tested and supported Linux distributions are those for which we offer a dedicated package.

Note that starting with VirtualBox 2.1, Linux 2.4-based host operating systems are no longer supported.

- **Solaris** hosts (64-bit only) are supported with the restrictions listed in [Chapter 14, *Known limitations*](#):
 - Solaris 11

- Solaris 10 (U10 and higher)

Note that the above list is informal. Oracle support for customers who have a support contract is limited to a subset of the listed host operating systems. Also, any feature which is marked as **experimental** is not supported. Feedback and suggestions about such features are welcome.

1.5. Installing VirtualBox and extension packs

VirtualBox comes in many different packages, and installation depends on your host operating system. If you have installed software before, installation should be straightforward: on each host platform, VirtualBox uses the installation method that is most common and easy to use. If you run into trouble or have special requirements, please refer to [Chapter 2, Installation details](#) for details about the various installation methods.

Starting with version 4.0, VirtualBox is split into several components.

1. The base package consists of all open-source components and is licensed under the GNU General Public License V2.
2. Additional extension packs can be downloaded which extend the functionality of the VirtualBox base package. Currently, Oracle provides the one extension pack, which can be found at <http://www.virtualbox.org> and provides the following added functionality:
 - a. The virtual USB 2.0 (EHCI) device; see [Section 3.10.1, "USB settings"](#).
 - b. The virtual USB 3.0 (xHCI) device; see [Section 3.10.1, "USB settings"](#).
 - c. VirtualBox Remote Desktop Protocol (VRDP) support; see [Section 7.1, "Remote display \(VRDP support\)"](#).
 - d. Host webcam passthrough; see chapter [Section 9.7.1, "Using a host webcam in the guest"](#).
 - e. Intel PXE boot ROM.
 - f. Experimental support for PCI passthrough on Linux hosts; see [Section 9.6, "PCI passthrough"](#).
 - g. Disk image encryption with AES algorithm; see [Section 9.31, "Encryption of disk images"](#).

VirtualBox extension packages have a `.vbox-extpack` file name extension. To install an extension, simply double-click on the package file and a Network Operations Manager window will appear, guiding you through the required steps.

To view the extension packs that are currently installed, please start the VirtualBox Manager (see the next section). From the "File" menu, please select "Preferences". In the window that shows up, go to the "Extensions" category which shows you the extensions which are currently installed and allows you to remove a package or add a new one.

Alternatively you can use VBoxManage on the command line: see [Section 8.41, "VBoxManage extpack"](#) for details.

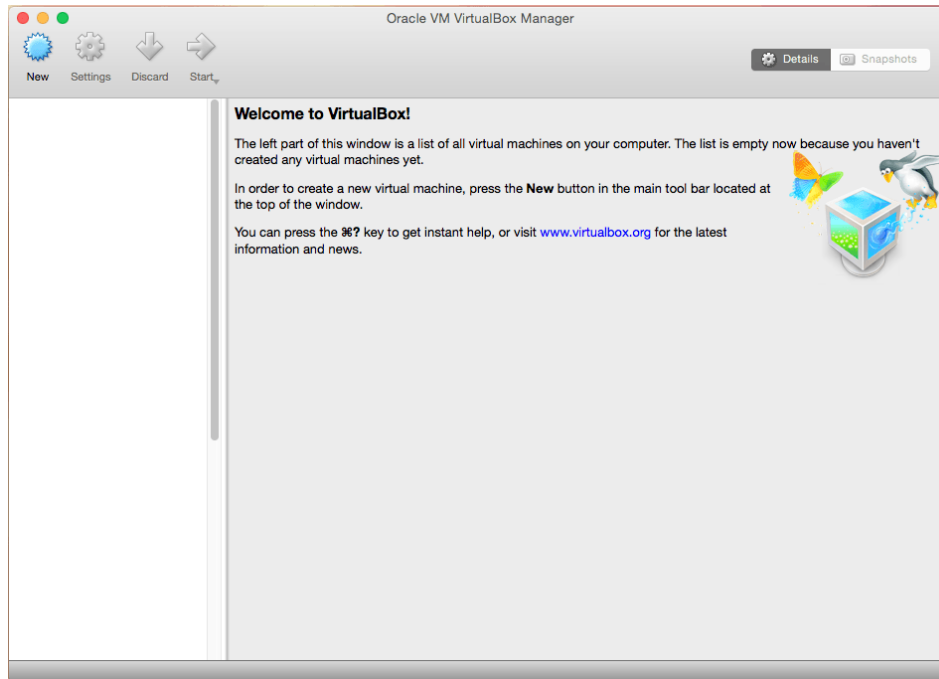
1.6. Starting VirtualBox

After installation, you can start VirtualBox as follows:

- On a Windows host, in the standard "Programs" menu, click on the item in the "VirtualBox" group. On Vista or Windows 7, you can also type "VirtualBox" in the search box of the "Start" menu.
- On a Mac OS X host, in the Finder, double-click on the "VirtualBox" item in the "Applications" folder. (You may want to drag this item onto your Dock.)

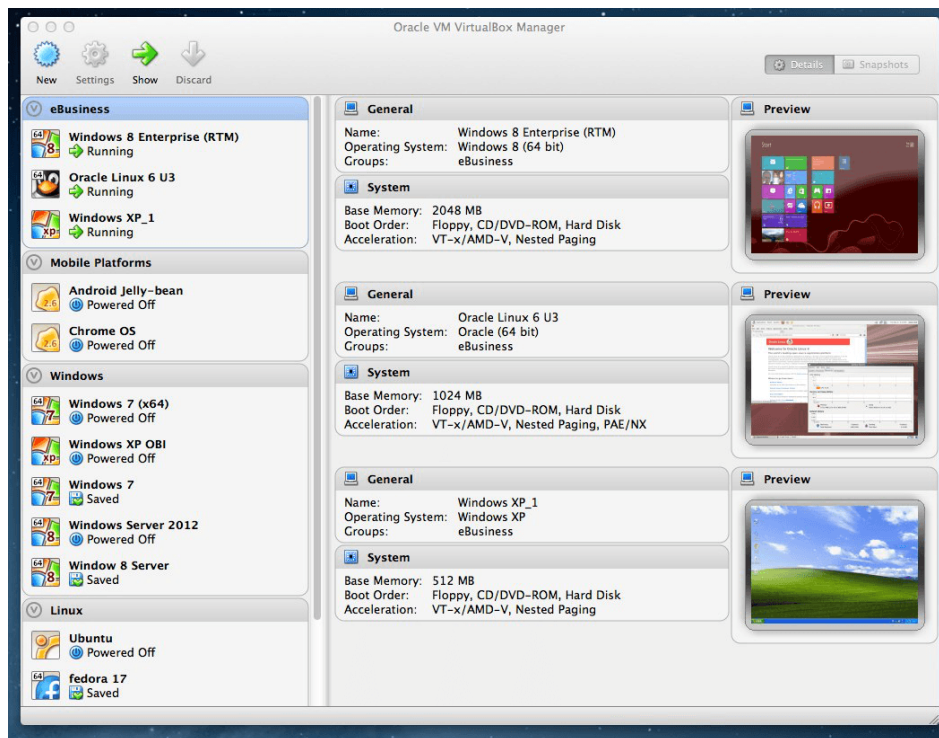
- On a Linux or Solaris host, depending on your desktop environment, a "VirtualBox" item may have been placed in either the "System" or "System Tools" group of your "Applications" menu. Alternatively, you can type `VirtualBox` in a terminal.

When you start VirtualBox for the first time, a window like the following should come up:



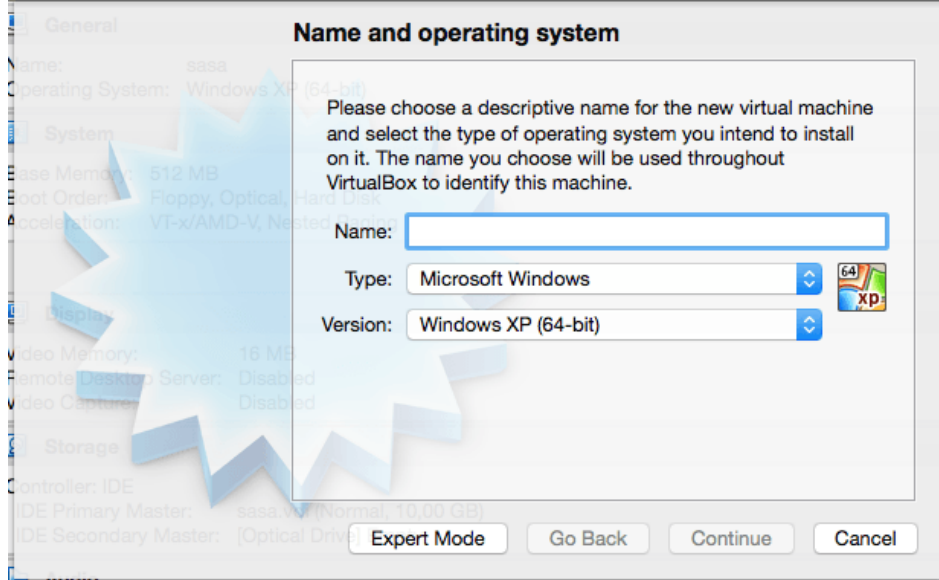
This window is called the **"VirtualBox Manager"**. On the left, you can see a pane that will later list all your virtual machines. Since you have not created any, the list is empty. A row of buttons above it allows you to create new VMs and work on existing VMs, once you have some. The pane on the right displays the properties of the virtual machine currently selected, if any. Again, since you don't have any machines yet, the pane displays a welcome message.

To give you an idea what VirtualBox might look like later, after you have created many machines, here's another example:



1.7. Creating your first virtual machine

Click on the "New" button at the top of the VirtualBox Manager window. A wizard will pop up to guide you through setting up a new virtual machine (VM):



On the following pages, the wizard will ask you for the bare minimum of information that is needed to create a VM, in particular:

1. The **VM name** will later be shown in the VM list of the VirtualBox Manager window, and it will be used for the VM's files on disk. Even though any name could be used, keep in mind that once you have created a few VMs, you will appreciate if you have given your VMs rather informative names; "My VM" would thus be less useful than "Windows XP SP2 with OpenOffice".
2. For **"Operating System Type"**, select the operating system that you want to install later. The supported operating systems are grouped; if you want to install something very unusual that is not listed, select "Other". Depending on your selection, VirtualBox will enable or disable certain VM settings that your guest operating system may require. This is particularly important for 64-bit guests (see [Section 3.1.2, "64-bit guests"](#)). It is therefore recommended to always set it to the correct value.
3. On the next page, select the **memory (RAM)** that VirtualBox should allocate every time the virtual machine is started. The amount of memory given here will be taken away from your host machine and presented to the guest operating system, which will report this size as the (virtual) computer's installed RAM.

Note

Choose this setting carefully! The memory you give to the VM will not be available to your host OS while the VM is running, so do not specify more than you can spare. For example, if your host machine has 1 GB of RAM and you enter 512 MB as the amount of RAM for a particular virtual machine, while that VM is running, you will only have 512 MB left for all the other software on your host. If you run two VMs at the same time, even more memory will be allocated for the second VM (which may not even be able to start if that memory is not available). On the other hand, you should specify as much as your guest OS (and your applications) will require to run properly.

A Windows XP guest will require at least a few hundred MB RAM to run properly, and Windows Vista will even refuse to install with less than 512 MB. Of course, if you want to run graphics-intensive applications in your VM, you may require even more RAM.

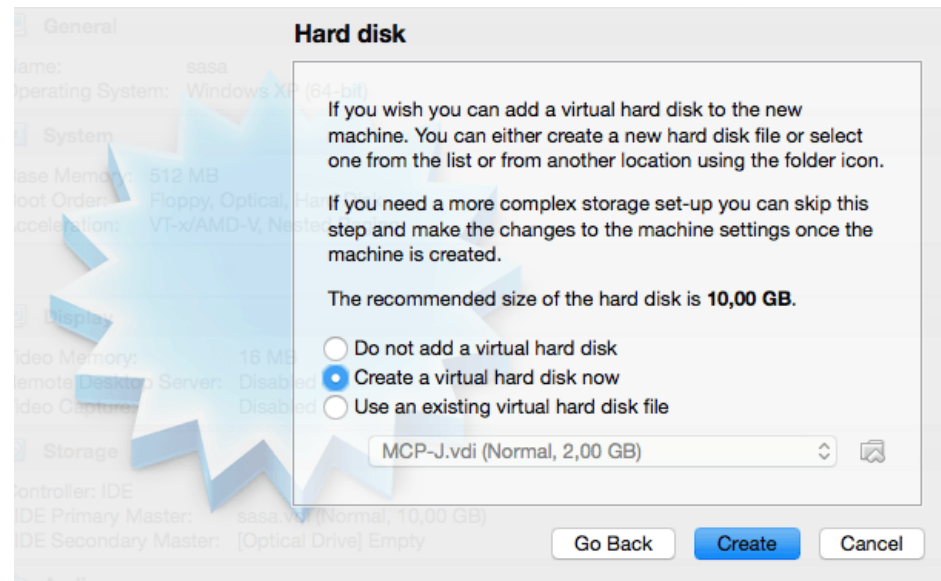
So, as a rule of thumb, if you have 1 GB of RAM or more in your host computer, it is usually safe to allocate 512 MB to each VM. But, in any case, make sure you always have at least 256 to 512 MB of RAM left on your host operating system. Otherwise you may cause your host OS to excessively swap out memory to your hard disk, effectively bringing your host system to a standstill.

As with the other settings, you can change this setting later, after you have created the VM.

4. Next, you must specify a **virtual hard disk** for your VM.

There are many and potentially complicated ways in which VirtualBox can provide hard disk space to a VM (see [Chapter 5, Virtual storage](#) for details), but the most common way is to use a large image file on your "real" hard disk, whose contents VirtualBox presents to your VM as if it were a complete hard disk. This file represents an entire hard disk then, so you can even copy it to another host and use it with another VirtualBox installation.

The wizard shows you the following window:



Here you have the following options:

- To create a new, empty virtual hard disk, press the **"New"** button.
- You can pick an **existing** disk image file.

The **drop-down list** presented in the window contains all disk images which are currently remembered by VirtualBox, probably because they are currently attached to a virtual machine (or have been in the past).

Alternatively, you can click on the small **folder button** next to the drop-down list to bring up a standard file dialog, which allows you to pick any disk image file on your host disk.

Most probably, if you are using VirtualBox for the first time, you will want to create a new disk image. Hence, press the "New" button.

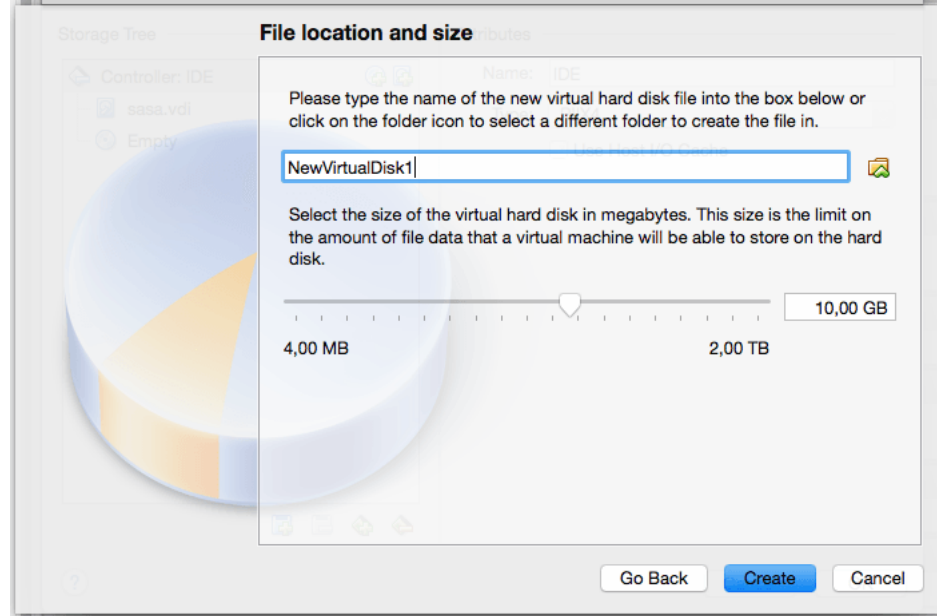
This brings up another window, the **"Create New Virtual Disk Wizard"**, which helps you create a new disk image file in the new virtual machine's folder.

VirtualBox supports two types of image files:

- A **dynamically allocated file** will only grow in size when the guest actually stores data on its virtual hard disk. It will therefore initially be small on the host hard drive and only later grow to the size specified as it is filled with data.
- A **fixed-size file** will immediately occupy the file specified, even if only a fraction of the virtual hard disk space is actually in use. While occupying much more space, a fixed-size file incurs less overhead and is therefore slightly faster than a dynamically allocated file.

For details about the differences, please refer to [Section 5.2, "Disk image files \(VDI, VMDK, VHD, HDD\)"](#).

To prevent your physical hard disk from running full, VirtualBox limits the size of the image file. Still, it needs to be large enough to hold the contents of your operating system and the applications you want to install -- for a modern Windows or Linux guest, you will probably need several gigabytes for any serious use. The limit of the image file size can be changed later (see [Section 8.23, "VBoxManage modifymedium"](#) for details).



After having selected or created your image file, again press **"Next"** to go to the next page.

5. After clicking on **"Finish"**, your new virtual machine will be created. You will then see it in the list on the left side of the Manager window, with the name you entered initially.

Note

After becoming familiar with the use of wizards, consider using the Expert Mode available in some wizards. Where available, this is selectable using a button, and speeds up user processes using wizards.

1.8. Running your virtual machine

To start a virtual machine, you have several options:

- Double-click on its entry in the list within the Manager window or
- select its entry in the list in the Manager window it and press the "Start" button at the top or
- for virtual machines created with VirtualBox 4.0 or later, navigate to the "VirtualBox VMs" folder in your system user's home directory, find the subdirectory of the machine you want to start and double-click on the machine settings file (with a `.vbox` file extension).

This opens up a new window, and the virtual machine which you selected will boot up. Everything which would normally be seen on the virtual system's monitor is shown in the window, as can be seen with the image in [Section 1.2, "Some terminology"](#).

In general, you can use the virtual machine much like you would use a real computer. There are couple of points worth mentioning however.

1.8.1. Starting a new VM for the first time

When a VM gets started for the first time, another wizard -- the **"First Start Wizard"** -- will pop up to help you select an **installation medium**. Since the VM is created empty, it would otherwise behave just like a real computer with no operating system installed: it will do nothing and display an error message that no bootable operating system was found.

For this reason, the wizard helps you select a medium to install an operating system from.

- If you have physical CD or DVD media from which you want to install your guest operating system (e.g. in the case of a Windows installation CD or DVD), put the media into your host's CD or DVD drive.

Then, in the wizard's drop-down list of installation media, select **"Host drive"** with the correct drive

letter (or, in the case of a Linux host, device file). This will allow your VM to access the media in your host drive, and you can proceed to install from there.

- If you have downloaded installation media from the Internet in the form of an ISO image file (most probably in the case of a Linux distribution), you would normally burn this file to an empty CD or DVD and proceed as just described. With VirtualBox however, you can skip this step and mount the ISO file directly. VirtualBox will then present this file as a CD or DVD-ROM drive to the virtual machine, much like it does with virtual hard disk images.

For this case, the wizard's drop-down list contains a list of installation media that were previously used with VirtualBox.

If your medium is not in the list (especially if you are using VirtualBox for the first time), select the small folder icon next to the drop-down list to bring up a standard file dialog, with which you can pick the image file on your host disks.

In both cases, after making the choices in the wizard, you will be able to install your operating system.

1.8.2. Capturing and releasing keyboard and mouse

As of version 3.2, VirtualBox provides a virtual USB tablet device to new virtual machines through which mouse events are communicated to the guest operating system. As a result, if you are running a modern guest operating system that can handle such devices, mouse support may work out of the box without the mouse being "captured" as described below; see [Section 3.4.1, "'Motherboard' tab"](#) for more information.

Otherwise, if the virtual machine only sees standard PS/2 mouse and keyboard devices, since the operating system in the virtual machine does not "know" that it is not running on a real computer, it expects to have exclusive control over your keyboard and mouse. This is, however, not the case since, unless you are running the VM in full screen mode, your VM needs to share keyboard and mouse with other applications and possibly other VMs on your host.

As a result, initially after installing a guest operating system and before you install the Guest Additions (we will explain this in a minute), only one of the two -- your VM or the rest of your computer -- can "own" the keyboard and the mouse. You will see a *second* mouse pointer which will always be confined to the limits of the VM window. Basically, you activate the VM by clicking inside it.

To return ownership of keyboard and mouse to your host operating system, VirtualBox reserves a special key on your keyboard for itself: the **"host key"**. By default, this is the *right Control* key on your keyboard; on a Mac host, the default host key is the left Command key. You can change this default in the VirtualBox Global Settings, see [Section 1.15, "Global Settings"](#). In any case, the current setting for the host key is always displayed *at the bottom right of your VM window*, should you have forgotten about it:



In detail, all this translates into the following:

- Your **keyboard** is owned by the VM if the VM window on your host desktop has the keyboard focus (and then, if you have many windows open in your guest operating system as well, the window that has the focus in your VM). This means that if you want to type within your VM, click on the title bar of your VM window first.

To release keyboard ownership, press the Host key (as explained above, typically the right Control key).

Note that while the VM owns the keyboard, some key sequences (like Alt-Tab for example) will no longer be seen by the host, but will go to the guest instead. After you press the host key to re-

enable the host keyboard, all key presses will go through the host again, so that sequences like Alt-Tab will no longer reach the guest. For technical reasons it may not be possible for the VM to get all keyboard input even when it does own the keyboard. Examples of this are the Ctrl-Alt-Del sequence on Windows hosts or single keys grabbed by other applications on X11 hosts like the GNOME desktop's "Control key highlights mouse pointer" functionality.

- Your **mouse** is owned by the VM only after you have clicked in the VM window. The host mouse pointer will disappear, and your mouse will drive the guest's pointer instead of your normal mouse pointer.

Note that mouse ownership is independent of that of the keyboard: even after you have clicked on a titlebar to be able to type into the VM window, your mouse is not necessarily owned by the VM yet.

To release ownership of your mouse by the VM, also press the Host key.

As this behavior can be inconvenient, VirtualBox provides a set of tools and device drivers for guest systems called the "VirtualBox Guest Additions" which make VM keyboard and mouse operation a lot more seamless. Most importantly, the Additions will get rid of the second "guest" mouse pointer and make your host mouse pointer work directly in the guest.

This will be described later in [Chapter 4, Guest Additions](#).

1.8.3. Typing special characters

Operating systems expect certain key combinations to initiate certain procedures. Some of these key combinations may be difficult to enter into a virtual machine, as there are three candidates as to who receives keyboard input: the host operating system, VirtualBox, or the guest operating system. Who of these three receives keypresses depends on a number of factors, including the key itself.

- Host operating systems reserve certain key combinations for themselves. For example, it is impossible to enter the **Ctrl+Alt+Delete** combination if you want to reboot the guest operating system in your virtual machine, because this key combination is usually hard-wired into the host OS (both Windows and Linux intercept this), and pressing this key combination will therefore reboot your *host*.

Also, on Linux and Solaris hosts, which use the X Window System, the key combination **Ctrl+Alt+Backspace** normally resets the X server (to restart the entire graphical user interface in case it got stuck). As the X server intercepts this combination, pressing it will usually restart your *host* graphical user interface (and kill all running programs, including VirtualBox, in the process).

Third, on Linux hosts supporting virtual terminals, the key combination **Ctrl+Alt+Fx** (where Fx is one of the function keys from F1 to F12) normally allows to switch between virtual terminals. As with Ctrl+Alt+Delete, these combinations are intercepted by the host operating system and therefore always switch terminals on the *host*.

If, instead, you want to send these key combinations to the *guest* operating system in the virtual machine, you will need to use one of the following methods:

- Use the items in the "Input" → "Keyboard" menu of the virtual machine window. There you will find "Insert Ctrl+Alt+Delete" and "Ctrl+Alt+Backspace"; the latter will only have an effect with Linux or Solaris guests, however.
- Press special key combinations with the Host key (normally the right Control key), which VirtualBox will then translate for the virtual machine:
 - **Host key + Del** to send Ctrl+Alt+Del (to reboot the guest);
 - **Host key + Backspace** to send Ctrl+Alt+Backspace (to restart the graphical user interface of a Linux or Solaris guest);

- **Host key + F1** (or other function keys) to simulate Ctrl+Alt+F1 (or other function keys, i.e. to switch between virtual terminals in a Linux guest).

- For some other keyboard combinations such as **Alt-Tab** (to switch between open windows), VirtualBox allows you to configure whether these combinations will affect the host or the guest, if a virtual machine currently has the focus. This is a global setting for all virtual machines and can be found under "File" → "Preferences" → "Input" → "Auto-capture keyboard".

1.8.4. Changing removable media

While a virtual machine is running, you can change removable media in the "Devices" menu of the VM's window. Here you can select in detail what VirtualBox presents to your VM as a CD, DVD, or floppy.

The settings are the same as would be available for the VM in the "Settings" dialog of the VirtualBox main window, but since that dialog is disabled while the VM is in the "running" or "saved" state, this extra menu saves you from having to shut down and restart the VM every time you want to change media.

Hence, in the "Devices" menu, VirtualBox allows you to attach the host drive to the guest or select a floppy or DVD image using the Disk Image Manager, all as described in [Section 1.11, "Virtual machine configuration"](#).

1.8.5. Resizing the machine's window

You can resize the virtual machine's window when it is running. In that case, one of three things will happen:

1. If you have **"scale mode"** enabled, then the virtual machine's screen will be scaled to the size of the window. This can be useful if you have many machines running and want to have a look at one of them while it is running in the background. Alternatively, it might be useful to enlarge a window if the VM's output screen is very small, for example because you are running an old operating system in it.

To enable scale mode, press the **host key + C**, or select "Scale mode" from the "Machine" menu in the VM window. To leave scale mode, press the host key + C again.

The aspect ratio of the guest screen is preserved when resizing the window. To ignore the aspect ratio, press Shift during the resize operation.

Please see [Chapter 14, *Known limitations*](#) for additional remarks.

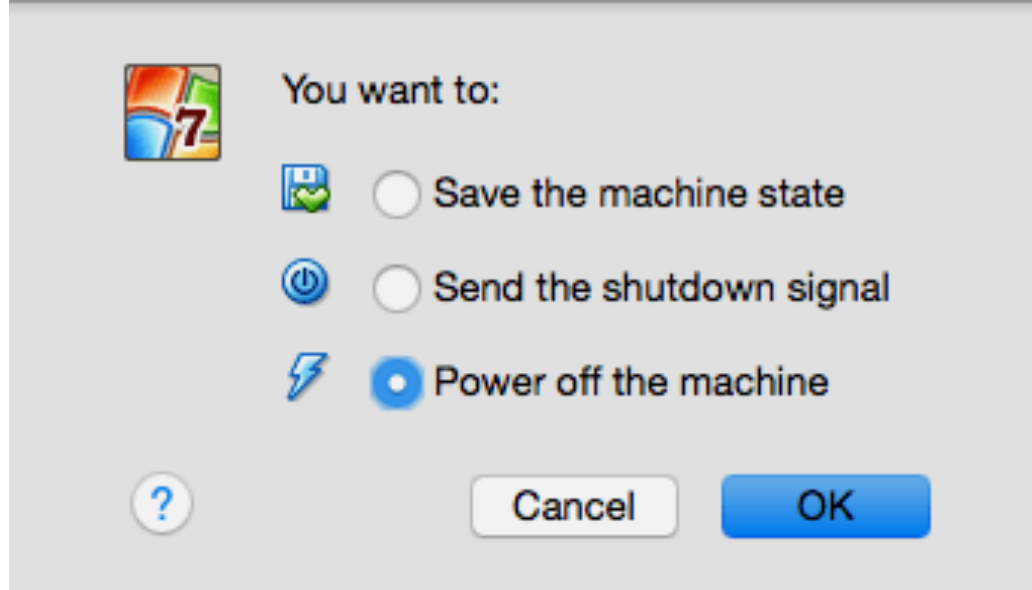
2. If you have the Guest Additions installed and they support automatic **resizing**, the Guest Additions will automatically adjust the screen resolution of the guest operating system. For example, if you are running a Windows guest with a resolution of 1024x768 pixels and you then resize the VM window to make it 100 pixels wider, the Guest Additions will change the Windows display resolution to 1124x768.

Please see [Chapter 4, *Guest Additions*](#) for more information about the Guest Additions.

3. Otherwise, if the window is bigger than the VM's screen, the screen will be centered. If it is smaller, then scroll bars will be added to the machine window.

1.8.6. Saving the state of the machine

When you click on the "Close" button of your virtual machine window (at the top right of the window, just like you would close any other window on your system), VirtualBox asks you whether you want to "save" or "power off" the VM. (As a shortcut, you can also press the Host key together with "Q".)



The difference between these three options is crucial. They mean:

- **Save the machine state:** With this option, VirtualBox "freezes" the virtual machine by completely saving its state to your local disk.

When you start the VM again later, you will find that the VM continues exactly where it was left off. All your programs will still be open, and your computer resumes operation. Saving the state of a virtual machine is thus in some ways similar to suspending a laptop computer (e.g. by closing its lid).

- **Send the shutdown signal.** This will send an ACPI shutdown signal to the virtual machine, which has the same effect as if you had pressed the power button on a real computer. So long as the VM is running a fairly modern operating system, this should trigger a proper shutdown mechanism from within the VM.
- **Power off the machine:** With this option, VirtualBox also stops running the virtual machine, but *without* saving its state.

Warning

This is equivalent to pulling the power plug on a real computer without shutting it down properly. If you start the machine again after powering it off, your operating system will have to reboot completely and may begin a lengthy check of its (virtual) system disks. As a result, this should not normally be done, since it can potentially cause data loss or an inconsistent state of the guest system on disk.

As an exception, if your virtual machine has any snapshots (see the next chapter), you can use this option to quickly **restore the current snapshot** of the virtual machine. In that case, powering off the machine will not disrupt its state, but any changes made since that snapshot was taken will be lost.

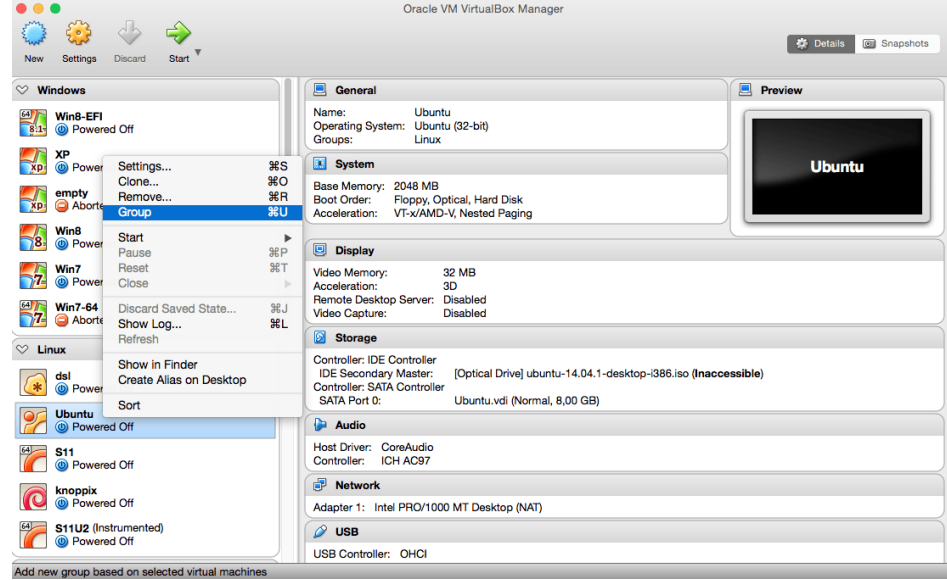
The **"Discard"** button in the VirtualBox Manager window discards a virtual machine's saved state. This has the same effect as powering it off, and the same warnings apply.

1.9. Using VM groups

VM groups enable the user to create ad hoc groups of VMs, and to manage and perform functions on them collectively, as well as individually. There are a number of features relating to groups:

1. Create a group using GUI option 1) Drag one VM on top of another VM.

Create a group using GUI option 2) Select multiple VMs and select "Group" on the right click menu, as follows:



2. Command line option 1) Create a group and assign a VM:

```
VBoxManage modifyvm "Fred" --groups "/TestGroup"
```

creates a group "TestGroup" and attaches the VM "Fred" to that group.

Command line option 2) Detach a VM from the group, and delete the group if empty:

```
VBoxManage modifyvm "Fred" --groups ""
```

It detaches all groups from the VM "Fred" and deletes the empty group.

3. Multiple groups e.g.:

```
VBoxManage modifyvm "Fred" --groups "/TestGroup,/TestGroup2"
```

It creates the groups "TestGroup" and "TestGroup2" (if they don't exist yet) and attaches the VM "Fred" to both of them.

4. Nested groups -- hierarchy of groups e.g.:

```
VBoxManage modifyvm "Fred" --groups "/TestGroup/TestGroup2"
```

It attaches the VM "Fred" to the subgroup "TestGroup2" of the "TestGroup" group.

5. Summary of group commands: Start, Pause, Reset, Close (save state, send shutdown signal, poweroff), Discard Saved State, Show in File System, Sort.

1.10. Snapshots

With snapshots, you can save a particular state of a virtual machine for later use. At any later time, you can revert to that state, even though you may have changed the VM considerably since then. A snapshot of a virtual machine is thus similar to a machine in "saved" state, as described above, but there can be many of them, and these saved states are preserved.

You can see the snapshots of a virtual machine by first selecting a machine in the VirtualBox Manager and then clicking on the "Snapshots" button at the top right. Until you take a snapshot of the machine, the list of snapshots will be empty except for the "Current state" item, which represents the "Now" point in the lifetime of the virtual machine.

1.10.1. Taking, restoring and deleting snapshots

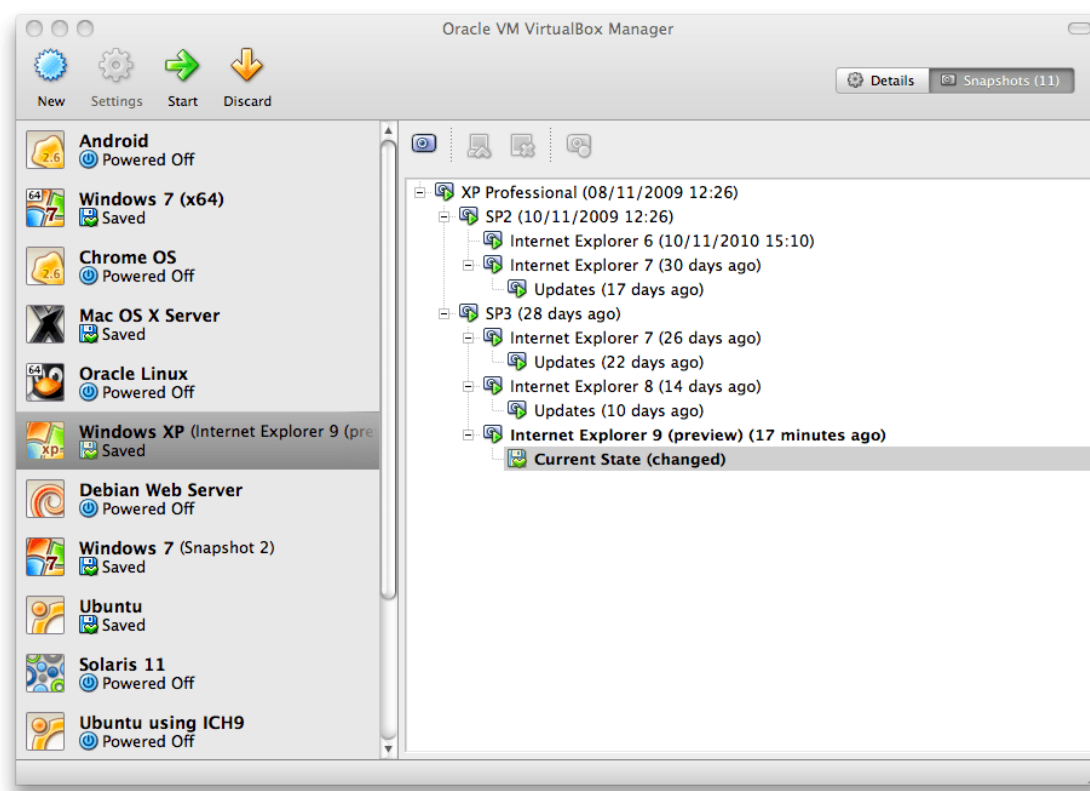
There are three operations related to snapshots:

1. You can **take a snapshot**. This makes a copy of the machine's current state, to which you can go back at any given time later.

- If your VM is currently running, select "Take snapshot" from the "Machine" pull-down menu of the VM window.
- If your VM is currently in either the "saved" or the "powered off" state (as displayed next to the VM in the VirtualBox main window), click on the "Snapshots" tab on the top right of the main window, and then
 - either on the small camera icon (for "Take snapshot") or
 - right-click on the "Current State" item in the list and select "Take snapshot" from the menu.

In any case, a window will pop up and ask you for a snapshot name. This name is purely for reference purposes to help you remember the state of the snapshot. For example, a useful name would be "Fresh installation from scratch, no Guest Additions", or "Service Pack 3 just installed". You can also add a longer text in the "Description" field if you want.

Your new snapshot will then appear in the snapshots list. Underneath your new snapshot, you will see an item called "Current state", signifying that the current state of your VM is a variation based on the snapshot you took earlier. If you later take another snapshot, you will see that they will be displayed in sequence, and each subsequent snapshot is derived from an earlier one:



VirtualBox imposes no limits on the number of snapshots you can take. The only practical limitation is disk space on your host: each snapshot stores the state of the virtual machine and thus occupies some disk space. (See the next section for details on what exactly is stored in a snapshot.)

2. You can **restore a snapshot** by right-clicking on any snapshot you have taken in the list of snapshots. By restoring a snapshot, you go back (or forward) in time: the current state of the machine is lost, and the machine is restored to the exact state it was in when the snapshot was taken.^[4]

Note

Restoring a snapshot will affect the virtual hard drives that are connected to your VM, as the entire state of the virtual hard drive will be reverted as well. This means also that all files that have been created since the snapshot and all other file changes *will be lost*. In order to prevent such data loss while still making use of the snapshot feature, it is possible to add a second hard drive in "write-through" mode using the `VBoxManage` interface and use it to store your data. As write-through hard

drives are *not* included in snapshots, they remain unaltered when a machine is reverted. See [Section 5.4, "Special image write modes"](#) for details.

To avoid losing the current state when restoring a snapshot, you can create a new snapshot before the restore.

By restoring an earlier snapshot and taking more snapshots from there, it is even possible to create a kind of alternate reality and to switch between these different histories of the virtual machine. This can result in a whole tree of virtual machine snapshots, as shown in the screenshot above.

3. You can also **delete a snapshot**, which will not affect the state of the virtual machine, but only release the files on disk that VirtualBox used to store the snapshot data, thus freeing disk space. To delete a snapshot, right-click on it in the snapshots tree and select "Delete". As of VirtualBox 3.2, snapshots can be deleted even while a machine is running.

Note

Whereas taking and restoring snapshots are fairly quick operations, deleting a snapshot can take a considerable amount of time since large amounts of data may need to be copied between several disk image files. Temporary disk files may also need large amounts of disk space while the operation is in progress.

There are some situations which cannot be handled while a VM is running, and you will get an appropriate message that you need to perform this snapshot deletion when the VM is shut down.

1.10.2. Snapshot contents

Think of a snapshot as a point in time that you have preserved. More formally, a snapshot consists of three things:

- It contains a complete copy of the VM settings, including the hardware configuration, so that when you restore a snapshot, the VM settings are restored as well. (For example, if you changed the hard disk configuration or the VM's system settings, that change is undone when you restore the snapshot.)

The copy of the settings is stored in the machine configuration, an XML text file, and thus occupies very little space.

- The complete state of all the virtual disks attached to the machine is preserved. Going back to a snapshot means that all changes that had been made to the machine's disks -- file by file, bit by bit -- will be undone as well. Files that were since created will disappear, files that were deleted will be restored, changes to files will be reverted.

(Strictly speaking, this is only true for virtual hard disks in "normal" mode. As mentioned above, you can configure disks to behave differently with snapshots; see [Section 5.4, "Special image write modes"](#). Even more formally and technically correct, it is not the virtual disk itself that is restored when a snapshot is restored. Instead, when a snapshot is taken, VirtualBox creates differencing images which contain only the changes since the snapshot were taken, and when the snapshot is restored, VirtualBox throws away that differencing image, thus going back to the previous state. This is both faster and uses less disk space. For the details, which can be complex, please see [Section 5.5, "Differencing images"](#).)

Creating the differencing image as such does not occupy much space on the host disk initially, since the differencing image will initially be empty (and grow dynamically later with each write operation to the disk). The longer you use the machine after having created the snapshot, however, the more the differencing image will grow in size.

- Finally, if you took a snapshot while the machine was running, the memory state of the machine is also saved in the snapshot (the same way the memory can be saved when you close the VM

window). When you restore such a snapshot, execution resumes at exactly the point when the snapshot was taken.

The memory state file can be as large as the memory size of the virtual machine and will therefore occupy quite some disk space as well.

1.11. Virtual machine configuration

When you select a virtual machine from the list in the Manager window, you will see a summary of that machine's settings on the right.

Clicking on the "Settings" button in the toolbar at the top brings up a detailed window where you can configure many of the properties of the selected VM. But be careful: even though it is possible to change all VM settings after installing a guest operating system, certain changes might prevent a guest operating system from functioning correctly if done after installation.

Note

The "Settings" button is disabled while a VM is either in the "running" or "saved" state. This is simply because the settings dialog allows you to change fundamental characteristics of the virtual computer that is created for your guest operating system, and this operating system may not take it well when, for example, half of its memory is taken away from under its feet. As a result, if the "Settings" button is disabled, shut down the current VM first.

VirtualBox provides a plethora of parameters that can be changed for a virtual machine. The various settings that can be changed in the "Settings" window are described in detail in [Chapter 3, Configuring virtual machines](#). Even more parameters are available with the VirtualBox command line interface; see [Chapter 8, VBoxManage](#).

1.12. Removing virtual machines

To remove a virtual machine which you no longer need, right-click on it in the Manager's VM list select "Remove" from the context menu that comes up.

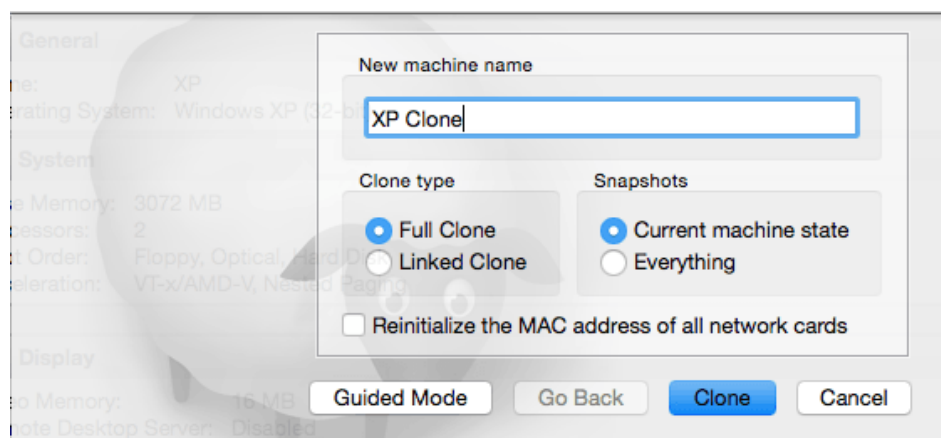
A confirmation window will come up that allows you to select whether the machine should only be removed from the list of machines or whether the files associated with it should also be deleted.

The "Remove" menu item is disabled while a machine is running.

1.13. Cloning virtual machines

To experiment with a VM configuration, test different guest OS levels or to simply backup a VM, VirtualBox can create a full or a linked copy of an existing VM.^[5]

A wizard will guide you through the clone process:



This wizard can be invoked from the context menu of the Manager's VM list (select "Clone") or the "Snapshots" view of the selected VM. First choose a new name for the clone. When you select **Reinitialize the MAC address of all network cards** every network card get a new MAC address assigned. This is useful when both, the source VM and the cloned VM, have to operate on the same network. If you leave this unchanged, all network cards have the same MAC address like the one in the source VM. Depending on how you invoke the wizard you have different choices for the cloning operation. First you need to decide if the clone should be linked to the source VM or a fully independent clone should be created:

- **Full clone:** In this mode all depending disk images are copied to the new VM folder. The clone can fully operate without the source VM.
- **Linked clone:** In this mode new differencing disk images are created where the parent disk images are the source disk images. If you selected the current state of the source VM as clone point, a new snapshot will be created implicitly.

After selecting the clone mode, you need to decide about what exactly should be cloned. You can always create a clone of the current state only or all. When you select all, the current state and in addition all snapshots are cloned. Have you started from a snapshot which has additional children, you can also clone the current state and all children. This creates a clone starting with this snapshot and includes all child snapshots.

The clone operation itself can be a lengthy operation depending on the size and count of the attached disk images. Also keep in mind that every snapshot has differencing disk images attached, which need to be cloned as well.

The "Clone" menu item is disabled while a machine is running.

For how to clone a VM at the command line, please see [Section 8.9, "VBoxManage clonevm"](#).

1.14. Importing and exporting virtual machines

VirtualBox can import and export virtual machines in the industry-standard Open Virtualization Format (OVF).^[6]

OVF is a cross-platform standard supported by many virtualization products which allows for creating ready-made virtual machines that can then be imported into a virtualizer such as VirtualBox. VirtualBox makes OVF import and export easy to access and supports it from the Manager window as well as its command-line interface. This allows for packaging so-called **virtual appliances**: disk images together with configuration settings that can be distributed easily. This way one can offer complete ready-to-use software packages (operating systems with applications) that need no configuration or installation except for importing into VirtualBox.

Note

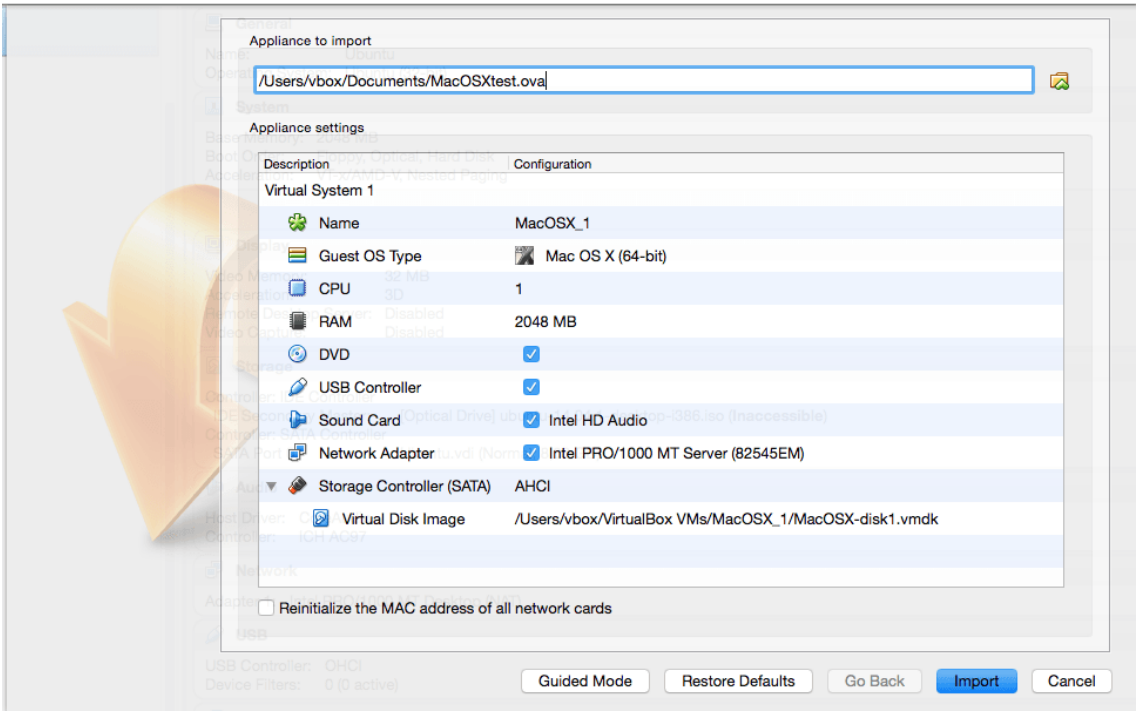
The OVF standard is complex, and support in VirtualBox is an ongoing process. In particular, no guarantee is made that VirtualBox supports all appliances created by other virtualization software. For a list of known limitations, please see [Chapter 14, Known limitations](#).

Appliances in OVF format can appear in two variants:

1. They can come in several files, as one or several disk images, typically in the widely-used VMDK format (see [Section 5.2, "Disk image files \(VDI, VMDK, VHD, HDD\)"](#)) and a textual description file in an XML dialect with an `.ovf` extension. These files must then reside in the same directory for VirtualBox to be able to import them.
2. Alternatively, the above files can be packed together into a single archive file, typically with an `.ova` extension. (Such archive files use a variant of the TAR archive format and can therefore be unpacked outside of VirtualBox with any utility that can unpack standard TAR files.)

To **import** an appliance in one of the above formats, simply double-click on the OVF/OVA file.^[7] Alternatively, select "File" → "Import appliance" from the Manager window. In the file dialog that comes up, navigate to the file with either the .ovf or the .ova file extension.

If VirtualBox can handle the file, a dialog similar to the following will appear:



This presents the virtual machines described in the OVF file and allows you to change the virtual machine settings by double-clicking on the description items. Once you click on "**Import**", VirtualBox will copy the disk images and create local virtual machines with the settings described in the dialog. These will then show up in the Manager's list of virtual machines.

Note that since disk images tend to be big, and VMDK images that come with virtual appliances are typically shipped in a special compressed format that is unsuitable for being used by virtual machines directly, the images will need to be unpacked and copied first, which can take a few minutes.

For how to import an image at the command line, please see [Section 8.10, "VBoxManage import"](#).

Conversely, to **export** virtual machines that you already have in VirtualBox, select "File" → "Export appliance". A different dialog window shows up that allows you to combine several virtual machines into an OVF appliance. Then, select the target location where the target files should be stored, and the conversion process begins. This can again take a while.

For how to export an image at the command line, please see [Section 8.11, "VBoxManage export"](#).

Note

OVF cannot describe snapshots that were taken for a virtual machine. As a result, when you export a virtual machine that has snapshots, only the current state of the machine will be exported, and the disk images in the export will have a "flattened" state identical to the current state of the virtual machine.

1.15. Global Settings

The global settings dialog can be reached through the **File** menu, selecting the **Preferences...** item. It offers a selection of settings which apply to all virtual machines of the current user or in the case of **Extensions** to the entire system:

1. **General** Enables the user to specify the default folder/directory for VM files, and the VRDP Authentication Library.
2. **Input** Enables the user to specify the Host Key. It identifies the key that toggles whether the cursor

is in the focus of the VM or the Host operating system windows (see [Section 1.8.2, “Capturing and releasing keyboard and mouse”](#)) and which is also used to trigger certain VM actions (see [Section 1.8.3, “Typing special characters”](#))

3. **Update** Enables the user to specify various settings for Automatic Updates.
4. **Language** Enables the user to specify the GUI language.
5. **Display** Enables the user to specify the screen resolution, and its width and height.
6. **Network** Enables the user to configure the details of Host Only Networks.
7. **Extensions** Enables the user to list and manage the installed extension packages.
8. **Proxy** Enables the user to configure a HTTP Proxy Server.

1.16. Alternative front-ends

As briefly mentioned in [Section 1.3, “Features overview”](#), VirtualBox has a very flexible internal design that allows for using multiple interfaces to control the same virtual machines. To illustrate, you can, for example, start a virtual machine with the VirtualBox Manager window and then stop it from the command line. With VirtualBox's support for the Remote Desktop Protocol (RDP), you can even run virtual machines remotely on a headless server and have all the graphical output redirected over the network.

In detail, the following front-ends are shipped in the standard VirtualBox package:

1. `VirtualBox` is the VirtualBox Manager. This graphical user interface uses the Qt toolkit; most of this User Manual is dedicated to describing it. While this is the easiest to use, some of the more advanced VirtualBox features are kept away from it to keep it simple.
2. `VBoxManage` is our command-line interface for automated and very detailed control of every aspect of VirtualBox. It is described in [Chapter 8, *VBoxManage*](#).
3. `VBoxSDL` is an alternative, simple graphical front-end with an intentionally limited feature set, designed to only display virtual machines that are controlled in detail with `VBoxManage`. This is interesting for business environments where displaying all the bells and whistles of the full GUI is not feasible. `VBoxSDL` is described in [Section 9.1, “VBoxSDL, the simplified VM displayer”](#).
4. Finally, `VBoxHeadless` is yet another front-end that produces no visible output on the host at all, but can act as a RDP server if the VirtualBox Remote Desktop Extension (VRDE) is installed and enabled for the VM. As opposed to the other graphical interfaces, the headless front-end requires no graphics support. This is useful, for example, if you want to host your virtual machines on a headless Linux server that has no X Window system installed. For details, see [Section 7.1.2, “VBoxHeadless, the remote desktop server”](#).

If the above front-ends still do not satisfy your particular needs, it is possible to create yet another front-end to the complex virtualization engine that is the core of VirtualBox, as the VirtualBox core neatly exposes all of its features in a clean API; please refer to [Chapter 11, *VirtualBox programming interfaces*](#).

[1] Support for 64-bit Windows was added with VirtualBox 1.5. Support for Windows XP was removed with VirtualBox 5.0.

[2] Preliminary Mac OS X support (beta stage) was added with VirtualBox 1.4, full support with 1.6. Mac OS X 10.4 (Tiger) support was removed with VirtualBox 3.1. Support for Mac OS X 10.7 (Lion) and earlier was removed with VirtualBox 5.0. Support for Mac OS X 10.8 (Mountain Lion) was removed with VirtualBox 5.1.

[3] Support for 64-bit Linux was added with VirtualBox 1.4.

[4] Both the terminology and the functionality of restoring snapshots has changed with VirtualBox 3.1. Before that version, it was only possible to go back to the very last snapshot taken -- not earlier ones, and the operation was called "Discard current state" instead of "Restore last snapshot". The limitation has been lifted with version 3.1. It is now possible to restore *any* snapshot, going backward and forward in time.

[5] Cloning support was introduced with VirtualBox 4.1.

[6] OVF support was originally introduced with VirtualBox 2.2 and has seen major improvements with every version since.

[7] Starting with version 4.0, VirtualBox creates file type associations for OVF and OVA files on your host operating system.