Programming the LEGO™ Mindstorms Robot Using C++, Code::Blocks, nxtOSEK and the nxtSIM

CPlusVEBot: C++ Virtual Environment for Programming Robots

Amy Delman
Lawrence Goetz
Mikhail Kunin
Yedidyah Langsam
Theodore Raphan

Brooklyn College
City University of New York

Version 1.2

Copyright © 2010

Table of Contents

Programming the LEGO™ Mindstorms Robot[1] Using C++, Code::Blocks[2], nxtOSEK[3] and the nxtSim[4]

Section 1: Introduction

Lego Robot systems are wheeled robots that can be assembled in different configurations and can be made to move and sense things through various sensors (e.g., touch, light, sound, etc.) as they move. Lego first marketed the Lego Mindstorms RCX, which was based on a Hitachi H8/300 Microprocessor. We developed an integrated system that is being used to teach C++[5], which had incorporated the use of the Lego Mindstorms RCX. Lego has upgraded their Mindstorms system to a more powerful system known as the NXT robot, which is based on an ARM processor. We have now developed an NXT-based system for teaching students C/C++.

The purpose of this tutorial is to give students an understanding of how to program the NXT robot using C/C++. You will be introduced to the commands that will control wheel motion and sensing through touch and light. This type of programming is especially important for science and engineering students because programming does not just involve writing code to perform calculations and output results on a computer screen, but also involves process control. Many computer applications require the writing of programs to sense the environment and perform a task based on the information that is sensed. These issues come up in multimedia applications, including game playing and, of course, robotics. By learning to program a robot, you will be introduced to the concept of how programs can be written to interact with the environment.

As this is the first course in programming, the programs that you will write will be simple, but will introduce you to basic mechanisms of sensing and control using C/C++.

[1] http://www.lego.com/
[2] http://www.codeblocks.org/
[3] http://brickos.sourceforge.net/
[4] http://hoenicke.ath.cx/rcx/brickemu.html
[5] http://eilat.sci.brooklyn.cuny.edu/cis1_5/Programming%20the%20LEGO.pdf

Section 2: Basic Concepts

We will first consider some basic concepts on the relationships among Programs and Compilers, Bios and Firmware, Operating Systems, Virtual Machines, and Integrated Development Environments (IDE's), which are critical for understanding the system and how programs for controlling the robot are developed.

Programs and Compilers:

A computer *program* is a set of instructions that can be followed by the computer to perform a given task. These instructions may be written in any number of languages, such as C++, Java, or Fortran, etc. These languages are known as high-level languages and are a convenient way for the programmer to implement an algorithm used to solve a problem. A program written in a high-level language is known as the *source code*. Unfortunately, the computer can only understand a difficult to follow low-level language, consisting of zeros and ones, known as *machine code*. A *compiler* translates the source code into an intermediate form known as *object code*, which is then combined with various pre-written routines contained in libraries to form the executable file (machine code) which can be *executed* (or *run*) on the computer. In order to distinguish the various files from one another, they are assigned extensions. Thus, source code written in C++ may have the *.cpp* extension, the output of the compiler the *.obj* extension and the executable may be assigned the *.exe* extension (Figure 1).

The system we describe for programming the NXT robots is nxtOSEK, which utilizes the GNU ARM cross-compiler to compile code written in C/C++ for the NXT robot. Under nxtOSEK, the compiler translates the C++ source code into a form understood by the robot, which is saved in a file having the *.rxe* extension

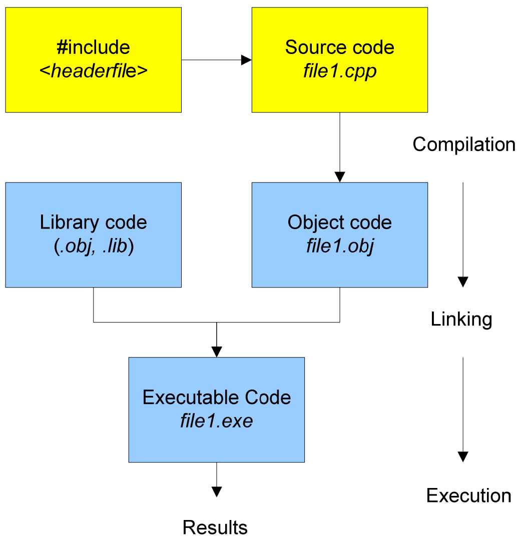


Figure 1

BIOS and Firmware:

All computers must follow a set of basic instructions, which start up the computer, allow for communication between the devices in your computer (such as your hard drive and graphics card) and the rest of the system hardware. This software is known as the *BIOS*, or Basic Input/Output System. The *BIOS* is stored in a semi-permanent location known as the *CMOS* (Complimentary Metal-Oxide Semiconductor) memory. The term *firmware* is used to describe the software that implements instruction sets that can run any device and is generally stored in the memory of a device. In the computer that runs a robot, the firmware is the program residing in the memory of the robot that interprets and implements the program that instructs the robot to behave in a certain fashion. While there various Firmware systems that have been developed for the NXT Lego robot, we will utilize John Hansen's Enhanced Firmware to run the C/C++ programs that will control the Lego NXT robot (See Section 11.2).

Operating Systems:

An *operating system* (OS) provides the interface between the user, the application programs and the computer's hardware, Figure 2.

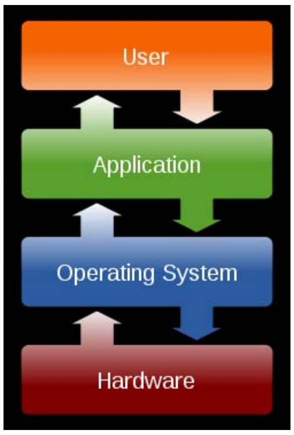


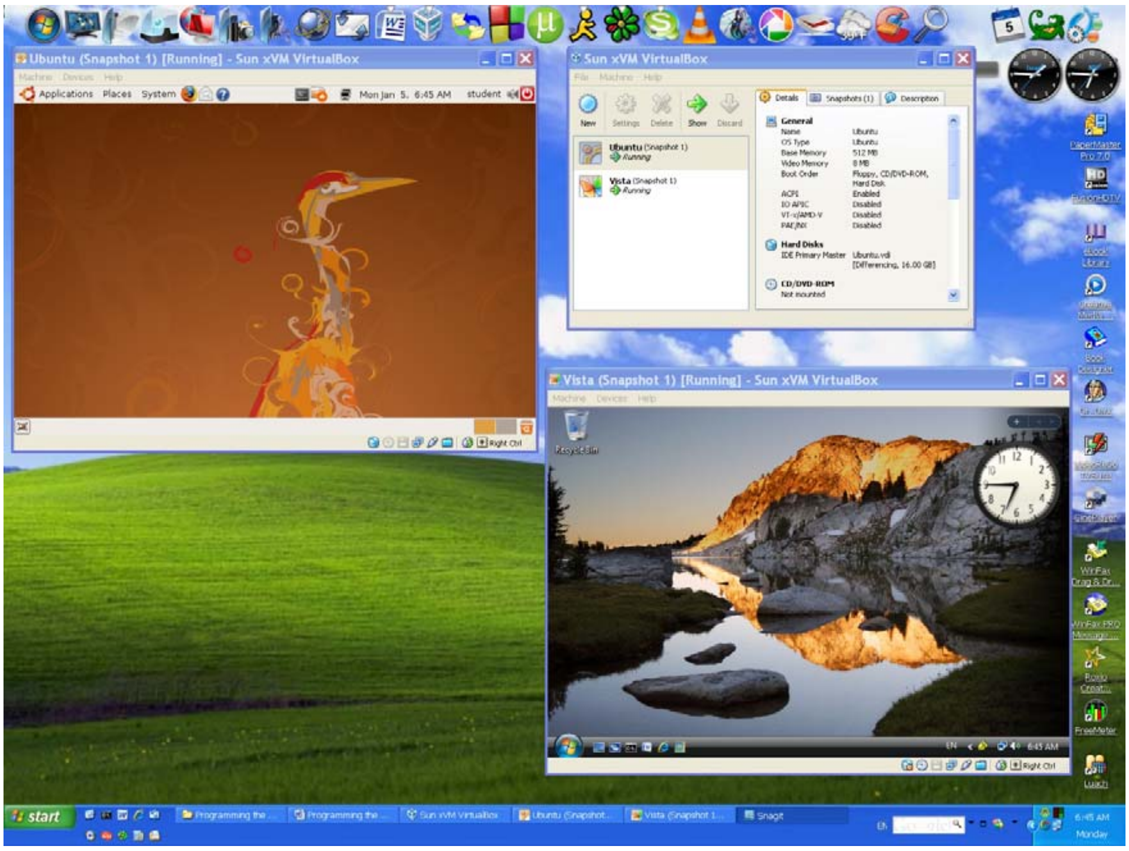
Figure 2

All programs that you use or write are managed by the operating system that is installed on your computer. Operating systems with which you may be familiar include: Windows XP, Windows Vista, Windows 7, Mac OS and Ubuntu Linux.

Virtual Machines:

Most users select a single operating system, often pre-installed on their computer at the time of purchase. Thus IBM compatible – Intel based computers often have Windows Vista or Windows XP pre-installed, while Apple – Intel based computers will have Mac OS X as their native operating system. (It is interesting to note, that Mac OS X is actually based upon a Linux variant). In order to take advantage of features and software programs that may be available only on a specific operating system, advanced users often find it useful to be able to use more than a single operating system. One way to accomplish this is by permanently installing two different operating systems using what is known as a dual-boot system. However, a more modern (and more flexible) alternative is known as platform virtualization or as the *virtual machine*. A virtual environment is created by software (such as Oracle's VirtualBox or Microsoft's VirtualPC) that is installed on a *host machine*, which is running a particular host operating system (This could be any of the operating systems described above). This software creates a *virtual machine* on which other operating systems can be installed.

These are known as *guest* operating systems. The guest operating system runs just as if it were installed on a stand-alone hardware platform that has the characteristics of the virtual machine. Typically, many virtual machines can be simulated on a single physical machine, their number limited by the host's hardware resources. Figure 3 is an example of Windows XP hosting two virtual machines: a virtual machine running a Windows Vista guest as well as an independent virtual machine running Ubuntu Linux as a guest, using VirtualBox as the virtualization platform.



General

Name	Ubuntu
OS Type	Ubuntu
Base Memory	512 MB
Video Memory	8 MB
Boot Order	Floppy, CD/DVD-ROM, Hard Disk
ACPI	Enabled
IO APIC	Disabled
VT-x/AMD-V	Disabled
PAE/NX	Disabled

Hard Disks

IDE Primary Master	Ubuntu.vdi [Differencing, 16.00 GB]

CD/DVD-ROM
Not mounted

Figure 3 - Windows XP hosting Vista and Ubuntu Linux

In this example, Windows XP is the host OS while Ubuntu Linux and Windows Vista are the guests, so that the user has the luxury of having the equivalent of three computers for the price of one.

Open Source Software:

Open source software (OSS) refers to software that is developed, tested, or improved through public collaboration and distributed with the idea that the source code must be shared with others, ensuring an open future collaboration.[6] In contrast to proprietary software, OSS may be freely downloaded and used (as long as they include the source code for others to modify and change). All the software that we will be using

[6] http://searchenterpriselinux.techtarget.com/sDefinition/0,,sid39_gci214343,00.html

in this manual is of the OSS variety. Because of the nature of the public collaboration, discussion forums are available for the user (and developer) who need further information. Students are urged to follow the links given in the Appendix for more information.

Section 3: Programming Environment

Programs in this course will be developed using an *Integrated Development Environment* (IDE) called *Code::Blocks***,** whose use and structure is described in the *Code::Blocks Student Manual*, by Goetz, Langsam and Raphan, available at the following site:

http://www.sci.brooklyn.cuny.edu/~goetz/codeblocks/codeblocks-instructions.pdf.

Briefly, Code::Blocks is an environment that allows you to *build* (*compile and link*) programs that can be run. For running C++ programs under Code::Blocks, a *compiler* will be used to translate the C++ programs which you will write, into an intermediate form of machine language which can be understood by the computer. A *linker* combines your program with other modules that are in a library that are necessary to print, read data and do the calculations that make the program work. To *build* your program is to compile and link the intermediate code and pre-written modules into the final form which can be executed on the computer. In more advanced programs, the linker combines different modules to form one major program. For normal programming, a compiler is used that works for the host system. When working on a PC under Microsoft Windows, the compiler will translate the program to work with the processor that is in the computer as well as generating the appropriate code to interact with the Windows environment. On a Mac, it will translate the program to the processor on the computer and interact with MacOS X. Similar translations will occur for computers running other operating systems such as Linux.

To control the NXT robot, it will be necessary to translate the program into a language the robot understands. This will require a different type of compiler, which is called a *cross-compiler*. This *cross-compiler* has been incorporated into the Code::Blocks IDE environment provided by the CPlusVEBot described in this manual. Thus, Code::Blocks can be used to cross-compile robot programs and translate them into the language that the robot microprocessor can run and make the robot perform.

Once the robot program has been compiled, it will then be necessary to download the program into the robot's control system, which we will refer to as the "*Brick*." This is done either through a wired USB (Universal Serial Bus) connection from the host computer or via a Bluetooth connection. Before a program can be downloaded to the robot microprocessor, however, the Lego firmware (which will reside in the memory of the Brick), must be downloaded. So, running a robot program requires six steps:

1. Download the firmware (John Hansen's Enhanced NXT Firmware) from the host computer to the Brick over the USB connection.
2. Write a program using the Code::Blocks IDE.
3. Build the program (cross-compile and link) using the Code::Blocks IDE.
4. Download the built program to the Brick.

5. Open the program on the Brick.
6. Push the *run* button on the Brick and watch the robot do its thing.

As mentioned in the introduction, the different stages of compilation and creating executable code generate files that have different extensions.

Extension	Purpose
.rxe	*filename.rxe* – compiled nxtOSEK C++ code
.c	*filename.c* – Source code written in C
.cpp	*filename.cpp* – Source code written in C++
.o or .obj	*filename.o* – Object code produced by the C++ compiler
.exe	*filename.exe* – Executable file produced by the linker which may be run on the computer

Table 1

Section 4: Real Brick and its Simulator

Since not everyone has the resources to buy and build a real Lego Robot, we developed a simulator, which we call *nxtSIM*. This simulator is a system that runs on a computer and executes the instructions of the real Brick, but on your computer. Using *nxtSIM* you can write programs for the Brick, and have them run just as if you had the real Brick. This helps debug programs that you will write so that you can then have them run on the real robot.

The Brick NXT Simulator has been designed to function from within a Linux environment. Whether using a Windows PC with the XP or Vista/Windows 7 operating system, or a Mac PC using OS X, all the programs necessary to program the real or simulated Brick have been incorporated within an Ubuntu Linux client hosted on a VirtualBox virtualized PC. This will provide a common environment for the entire class.

Section 5: The Lego Robot and the NXT Brick

One of the many robots that can be built using the Lego NXT Robot system is shown in Figure 4.

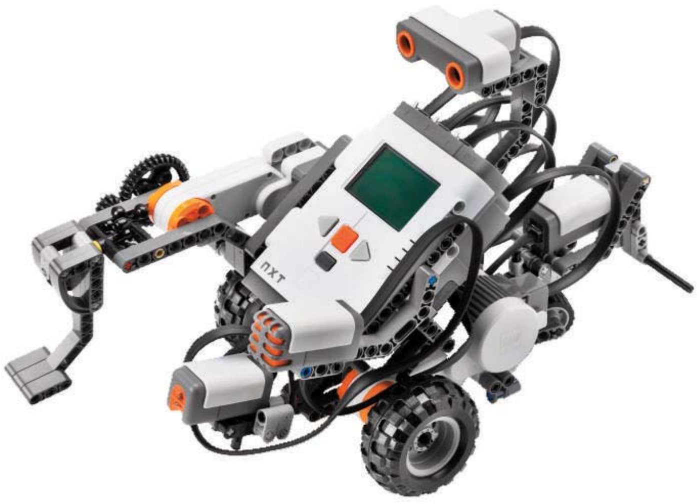


Figure 4 - A Lego NXT Roverbot with various sensors and motors.

The brain of the Lego Robot is the NXT Brick), which contains an ARM AT91SAM7S256 microprocessor. There are also four sensor ports (Sensor 1, Sensor 2, Sensor 3, and Sensor 4) and three motor ports (A, B, C). The motors and sensors can be independently controlled by the program. It also has a LED display so that messages can be sent to the robot and displayed. Programs can be downloaded to the Brick via the Lego USB port or through a Bluetooth interface that establishes a proprietary link between the robot and host computer.

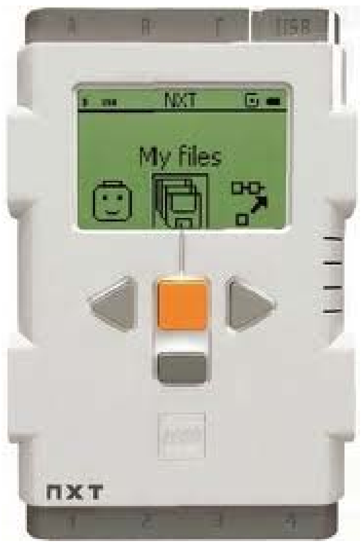


Figure 5 - Lego NXT Brick

nxtSIM – Simulator for NXT Brick:

The nxtSIM was designed to mimic the actual NXT Brick and is shown in Figure 6.

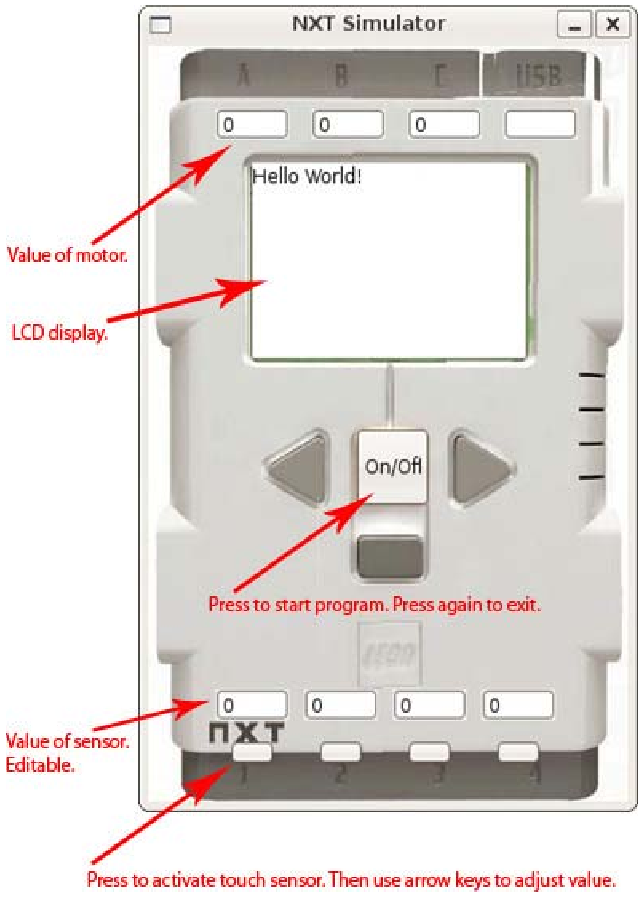

NXT Simulator
A
B
C
USB
0
0
0
Hello World!
Value of motor.
LCD display.
On/Off
Press to start program. Press again to exit.
0
0
0
0
Value of sensor. Editable.
NXT
1
2
3
4
Press to activate touch sensor. Then use arrow keys to adjust value.

Figure 6 - nxtSIM

The values sent to the motor ports are displayed in the boxes at the top of the display (currently displaying a value of 0 for all three motor ports). The sensor ports (shown at the bottom of the display) may be modified by clicking on a button underneath the sensor and using the LEFT/RIGHT and UP/DOWN arrow keys (LEFT/RIGHT change in steps of 10 and UP/DOWN keys change the values in steps of 1). Pressing Q, W, E, R, represent the four sensor inputs. Holding down one of these keys will simulate a touch sensor being pressed. When the ON button is pressed, the program for the simulator runs. Pressing the ON/OFF button again will close the simulator. The center panel is a window that simulates the brick’s LCD.

We will have more to say about the simulator below.

Section 6: Installation

6.1. System requirements[7]

In order to run VirtualBox on your machine, you need:

- Reasonably powerful **x86 hardware.** Any recent Intel or AMD processor should do. (Note: You cannot run VirtualBox on a PowerPC based Mac.)
- **Memory.** You will need at least 512 MB of RAM (but probably more, and the more the better). Basically, you will need whatever your host operating system needs to run comfortably, plus the amount that the guest operating system needs. So you probably won't enjoy the experience much with less than 1 GB of RAM. Two or more GB of RAM will be more than adequate on an XP host, while 3-GB of RAM will be more than adequate for a Vista/Windows 7 Host.
- **Hard disk space.** While VirtualBox itself takes up little space on the disk (a typical installation will only need about 30 MB of hard disk space), the virtual machines will require fairly huge files on disk to represent their own hard disk storage. So, to install Windows XP, for example, you will need a file that will easily grow to several GB in size. For our system you will need a minimum of 2 GB free space. The VMDK file (containing the virtual media disk) is just under 2 GB. Technically the VMDK file has a maximum of 16 GB so that a total of 20GB of free space would be more than adequate for future growth.

6.2. Installation

The installation of the system while lengthy is quite easy to do. It contains three steps:

a. Copying the installation files to your disk
b. Installing VirtualBox
c. Installing Ubuntu Linux

Step a: Copying the installation files to your disk (Windows based PC)

Insert the Installation Disk (available from your instructor) into your computer.

After a short period of time, on a Windows PC you will see a screen similar to the following.

If you are using a Mac please go to page 14.

[7] http://www.virtualbox.org:80/wiki/End-user_documentation

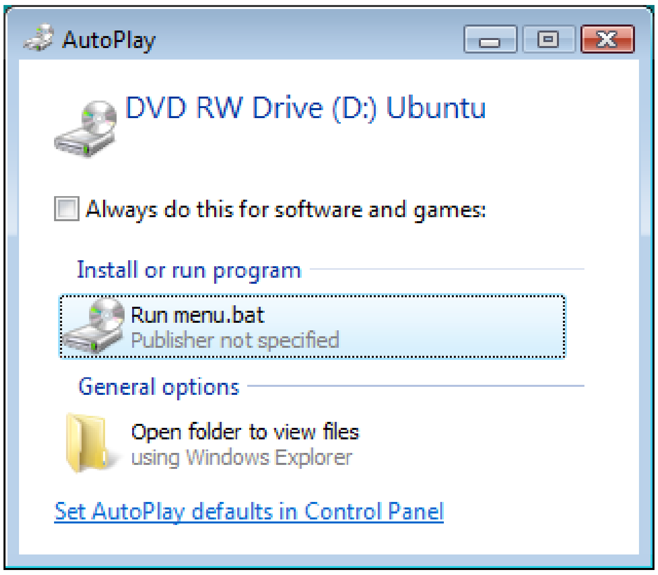
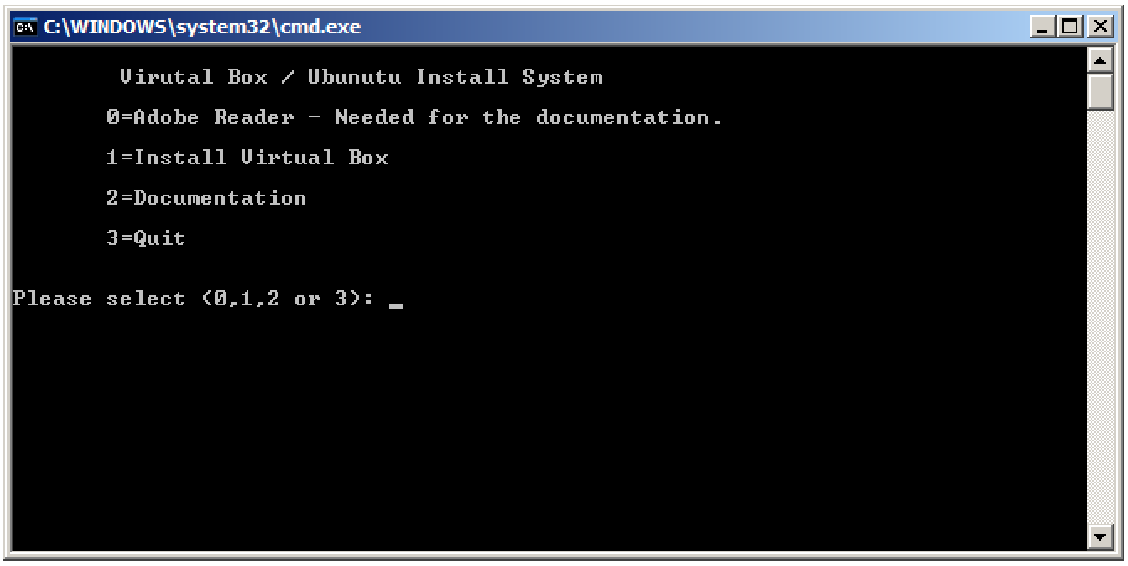


Figure 7

Select **Run Menu.bat.** The screen on the next page will appear.



```
C:\WINDOWS\system32\cmd.exe

        Virutal Box / Ubunutu Install System
       0=Adobe Reader - Needed for the documentation.
       1=Install Virtual Box
       2=Documentation
       3=Quit

Please select (0,1,2 or 3): _
```

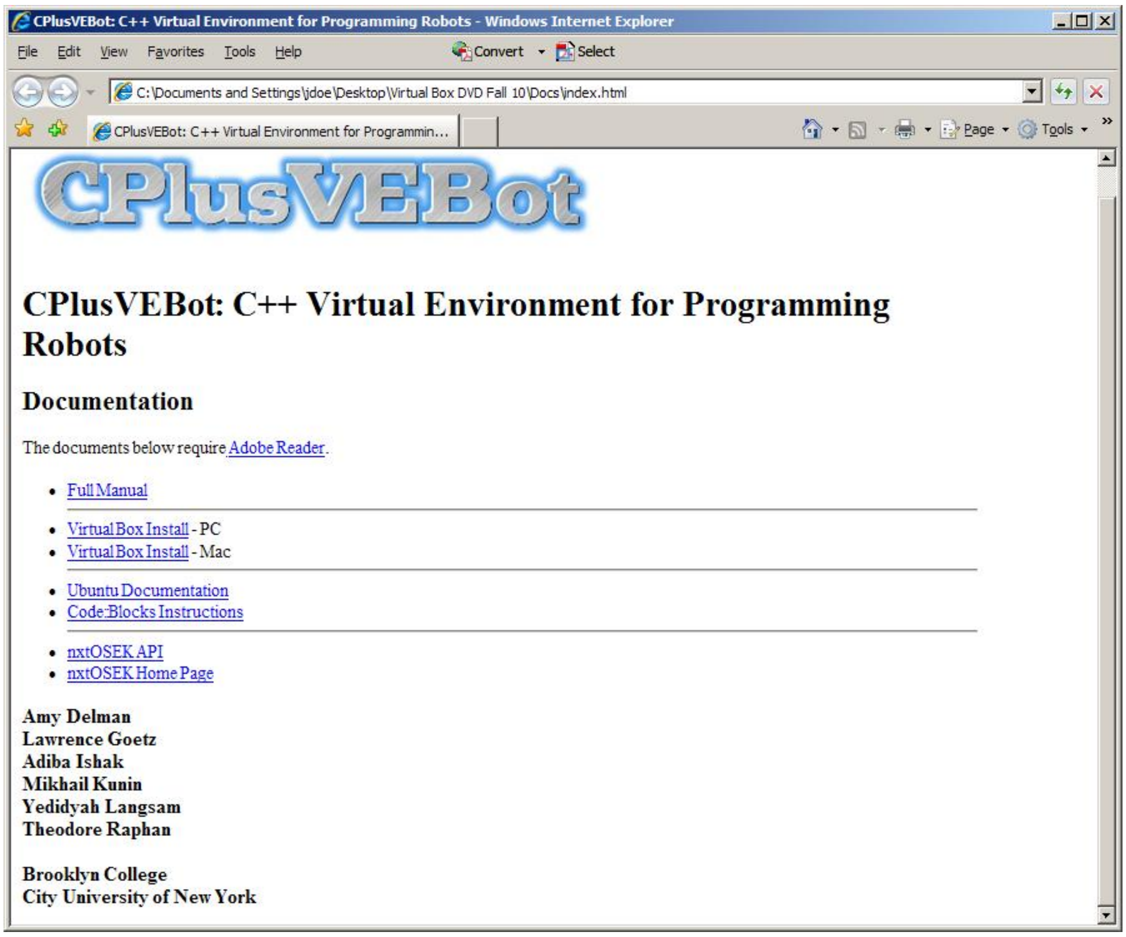
Figure 8

Select *choice 1* in order to install VirtualBox .

To read the documentation, select *choice 2* then *Enter*.

To quit, select *choice 3* and then *Enter*.

The DVD contains some useful documentation for you to read. Upon selecting *choice 3* the following Documentation Screen appears:


CPlusVEBot: C++ Virtual Environment for Programming Robots - Windows Internet Explorer
File Edit View Favorites Tools Help
Convert Select
C:\Documents and Settings\jdoe\Desktop\Virtual Box DVD Fall 10\Docs\index.html
CPlusVEBot: C++ Virtual Environment for Programmin...
Page Tools
CPlusVEBot

CPlusVEBot: C++ Virtual Environment for Programming Robots

Documentation

The documents below require Adobe Reader.

- Full Manual
- Virtual Box Install - PC
- Virtual Box Install - Mac
- Ubuntu Documentation
- Code:Blocks Instructions
- nxtOSEK API
- nxtOSEK Home Page

Amy Delman
Lawrence Goetz
Adiba Ishak
Mikhail Kunin
Yedidyah Langsam
Theodore Raphan

Brooklyn College
City University of New York

Figure 9

If you are installing VirtualBox on a Windows based PC please skip to page 16.

Step a: Copying the installation files to your disk (MAC OS X)

If you are using a Windows based PC please go to page 11.

Insert the Installation Disk (available from your instructor) into your computer.

After a short period of time, on a MAC OS X you will see a screen similar to the following.

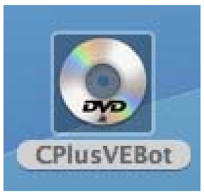


Figure 10

Click on the DVD icon.

You will then see the following screen:

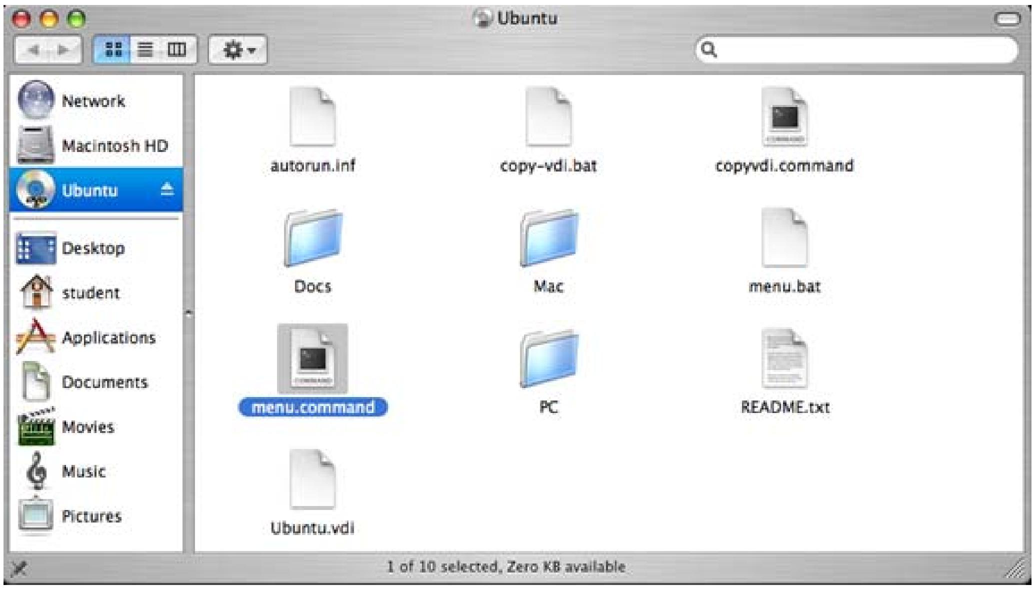
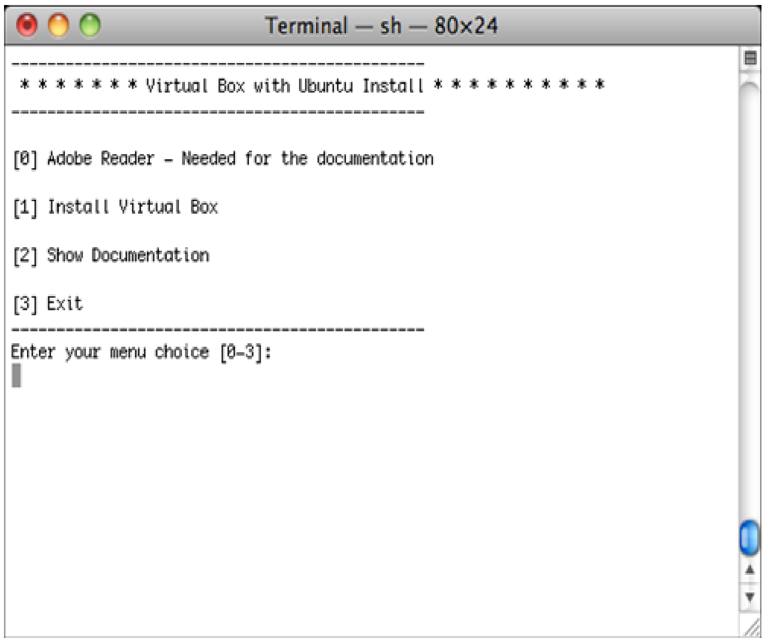


Figure 11

Double click on the file called *menu.command* and you will see the screen below



```
----------------------------------------------
 * * * * * * * Virtual Box with Ubuntu Install * * * * * * * * * *
----------------------------------------------

[0] Adobe Reader - Needed for the documentation

[1] Install Virtual Box

[2] Show Documentation

[3] Exit
----------------------------------------------
Enter your menu choice [0-3]:
```

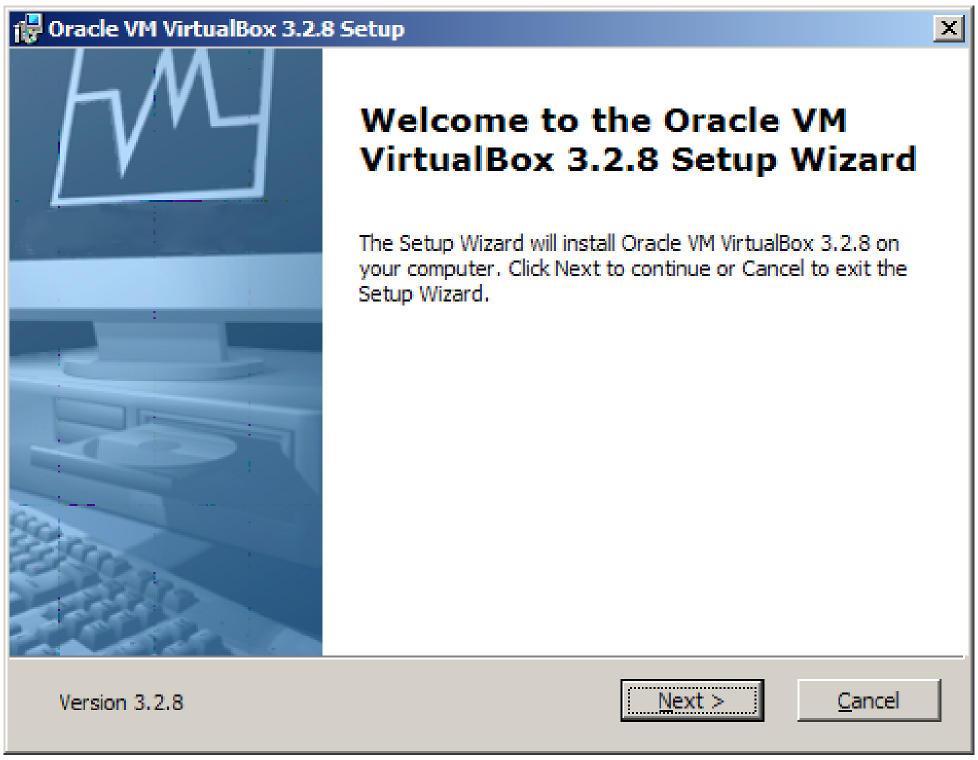
Figure 12

Select menu option 1 to install VirtualBox and proceed to *Step b: Installing VirtualBox (on a Mac)* on page 26.

Step b: Installing VirtualBox (on a PC)

If you are installing VirtualBox on an Apple Mac please skip to page 26.

You will see the following screen. Follow the steps of the installation wizard as shown below:


Oracle VM VirtualBox 3.2.8 Setup

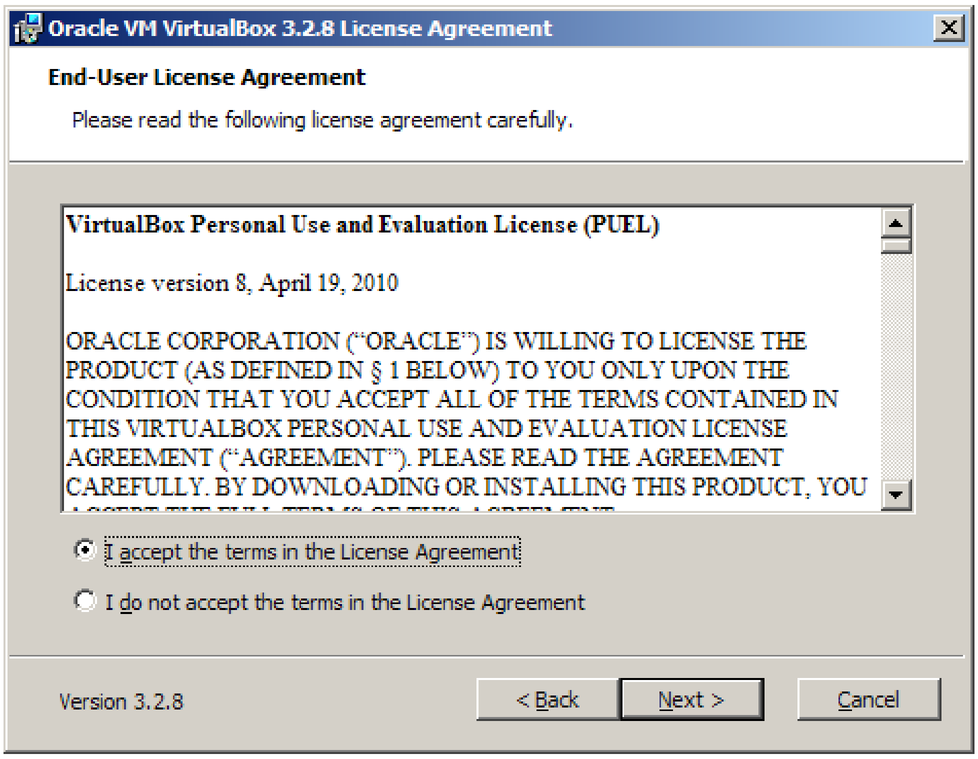
Welcome to the Oracle VM VirtualBox 3.2.8 Setup Wizard

The Setup Wizard will install Oracle VM VirtualBox 3.2.8 on your computer. Click Next to continue or Cancel to exit the Setup Wizard.

Version 3.2.8

Figure 13

Click **Next** to contine.


Oracle VM VirtualBox 3.2.8 License Agreement

End-User License Agreement

Please read the following license agreement carefully.

VirtualBox Personal Use and Evaluation License (PUEL)

License version 8, April 19, 2010

ORACLE CORPORATION ("ORACLE") IS WILLING TO LICENSE THE PRODUCT (AS DEFINED IN § 1 BELOW) TO YOU ONLY UPON THE CONDITION THAT YOU ACCEPT ALL OF THE TERMS CONTAINED IN THIS VIRTUALBOX PERSONAL USE AND EVALUATION LICENSE AGREEMENT ("AGREEMENT"). PLEASE READ THE AGREEMENT CAREFULLY. BY DOWNLOADING OR INSTALLING THIS PRODUCT, YOU

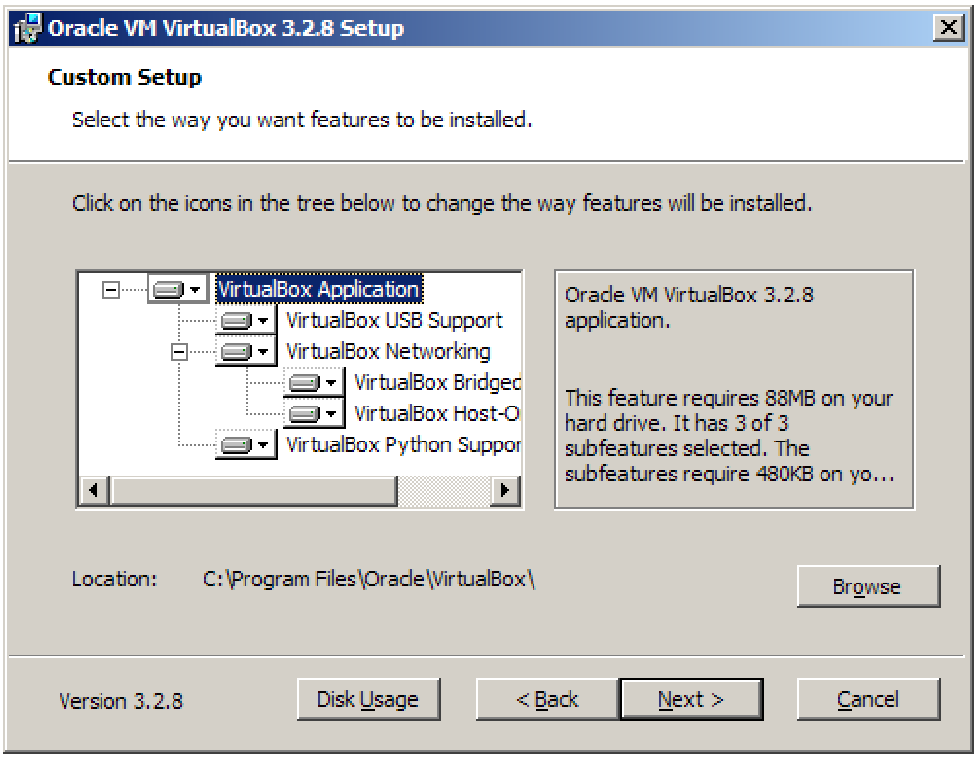
[✓ I accept the terms in the License Agreement]

I do not accept the terms in the License Agreement

Version 3.2.8

Figure 14

Accept the agreement and select **Next**.


Oracle VM VirtualBox 3.2.8 Setup

Custom Setup

Select the way you want features to be installed.

Click on the icons in the tree below to change the way features will be installed.

- VirtualBox Application
  - VirtualBox USB Support
  - VirtualBox Networking
    - VirtualBox Bridged
    - VirtualBox Host-O
  - VirtualBox Python Suppor

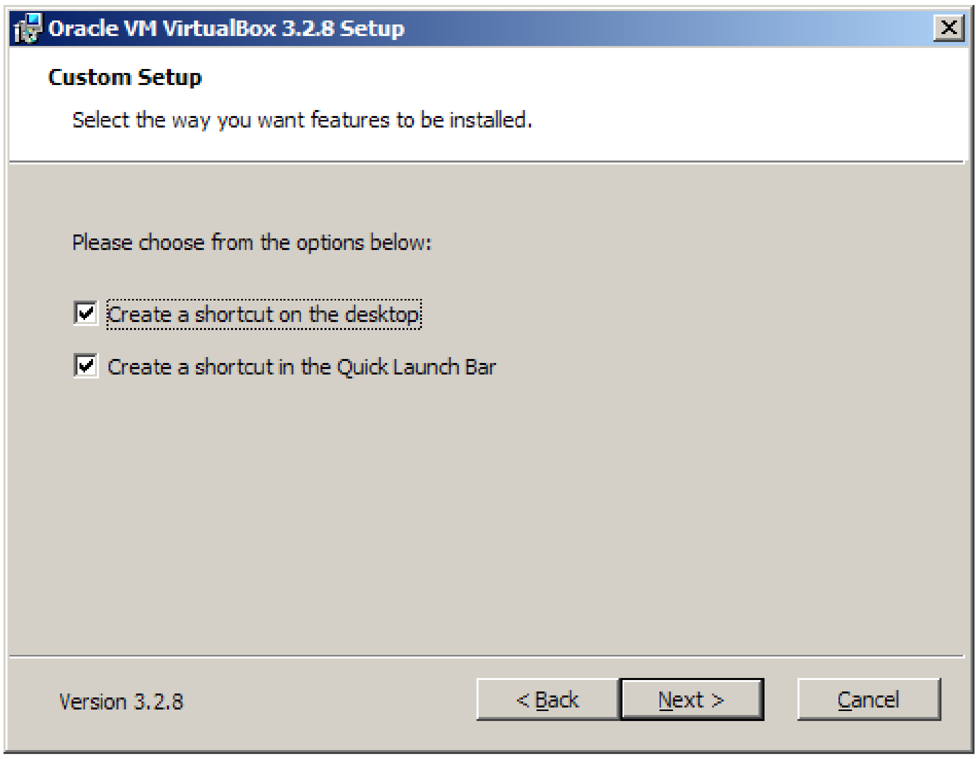
Oracle VM VirtualBox 3.2.8 application.

This feature requires 88MB on your hard drive. It has 3 of 3 subfeatures selected. The subfeatures require 480KB on yo...

Location: C:\Program Files\Oracle\VirtualBox\

Figure 15

Leave the defaults set and click **Next**.


Oracle VM VirtualBox 3.2.8 Setup
Custom Setup
Select the way you want features to be installed.

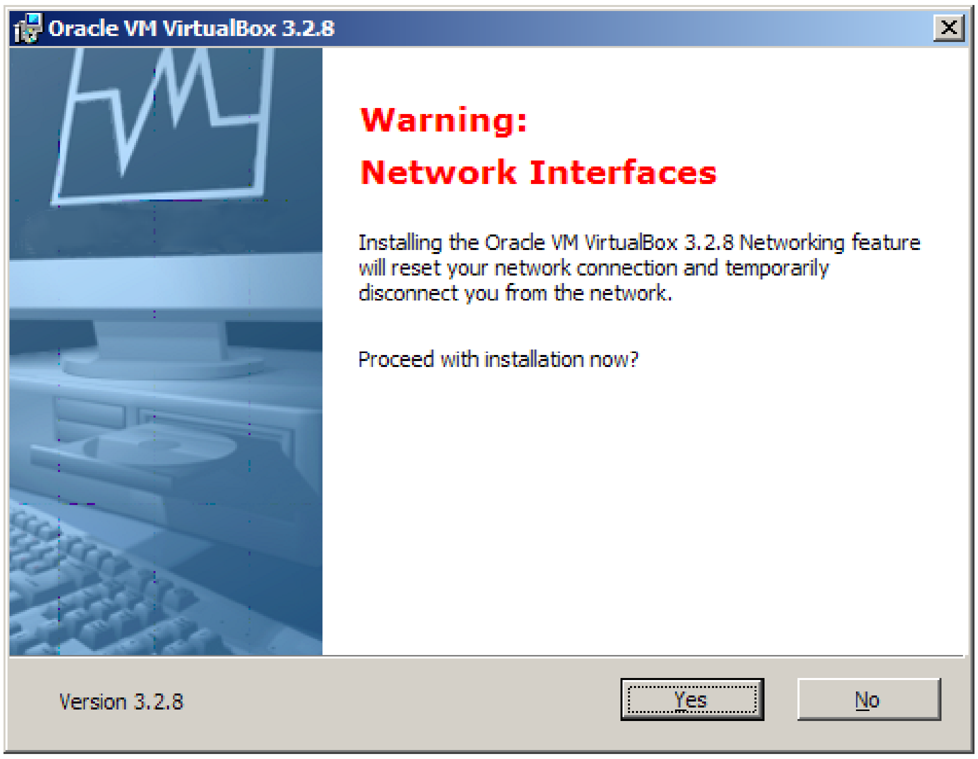
Please choose from the options below:

- [✓] Create a shortcut on the desktop
- [✓] Create a shortcut in the Quick Launch Bar

Version 3.2.8
< Back
Next >
Cancel

Figure 16

Leave the defaults set and click **Next**.



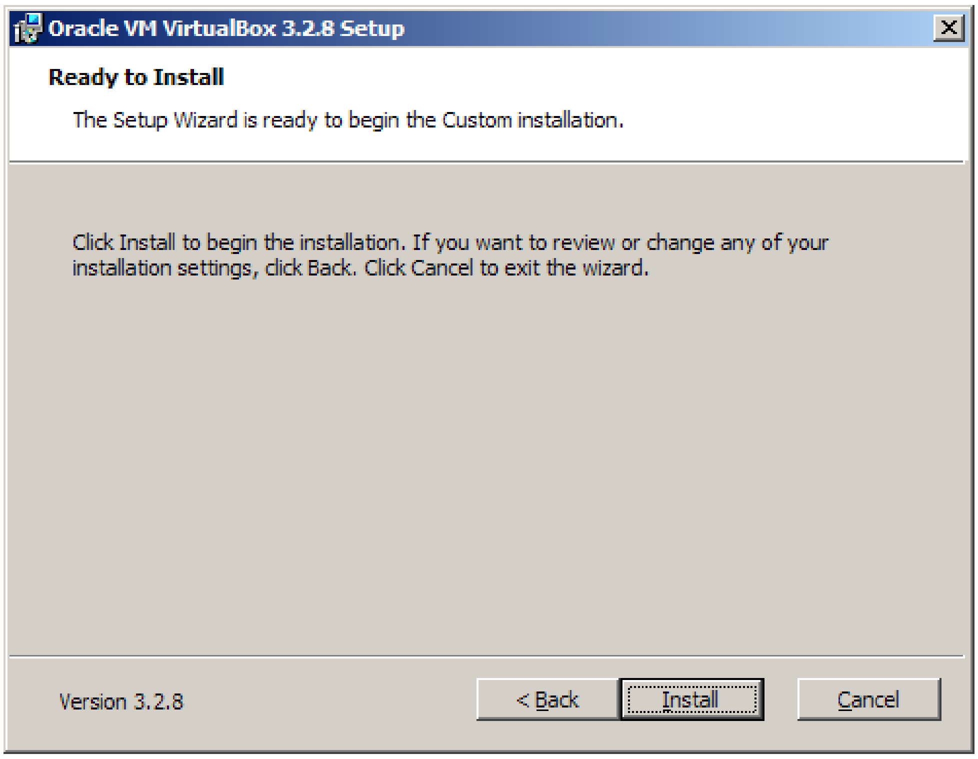
Warning:
Network Interfaces

Installing the Oracle VM VirtualBox 3.2.8 Networking feature will reset your network connection and temporarily disconnect you from the network.

Proceed with installation now?

Figure 17

Click **Yes** to install the Network Interfaces.


Oracle VM VirtualBox 3.2.8 Setup

Ready to Install

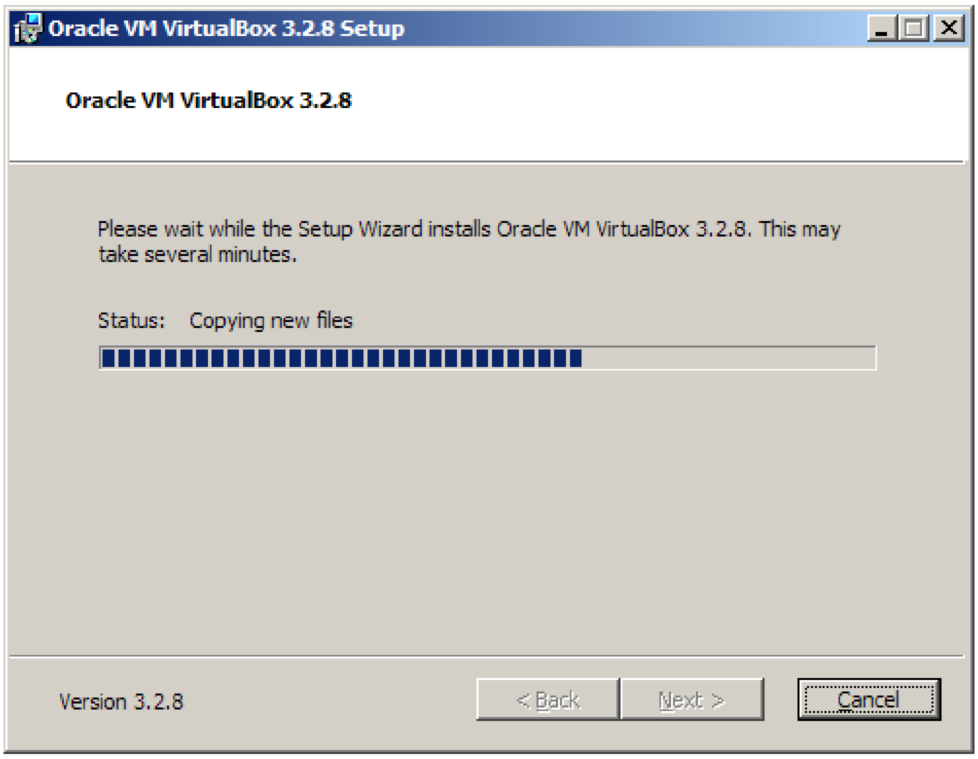
The Setup Wizard is ready to begin the Custom installation.

Click Install to begin the installation. If you want to review or change any of your installation settings, click Back. Click Cancel to exit the wizard.

Version 3.2.8

Figure 18

Click **Install** to install the program.


Oracle VM VirtualBox 3.2.8 Setup

Oracle VM VirtualBox 3.2.8

Please wait while the Setup Wizard installs Oracle VM VirtualBox 3.2.8. This may take several minutes.

Status: Copying new files

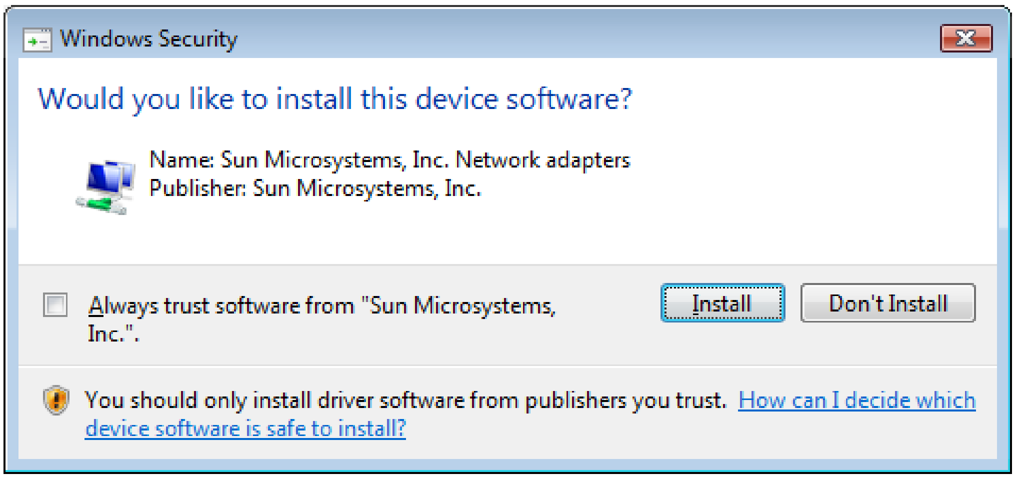
Version 3.2.8

Figure 19

Please wait.

If any windows ask for your permission to continue, please allow them to install.

If you get a window like these, select to **Install** this driver software anyway.



Windows Security

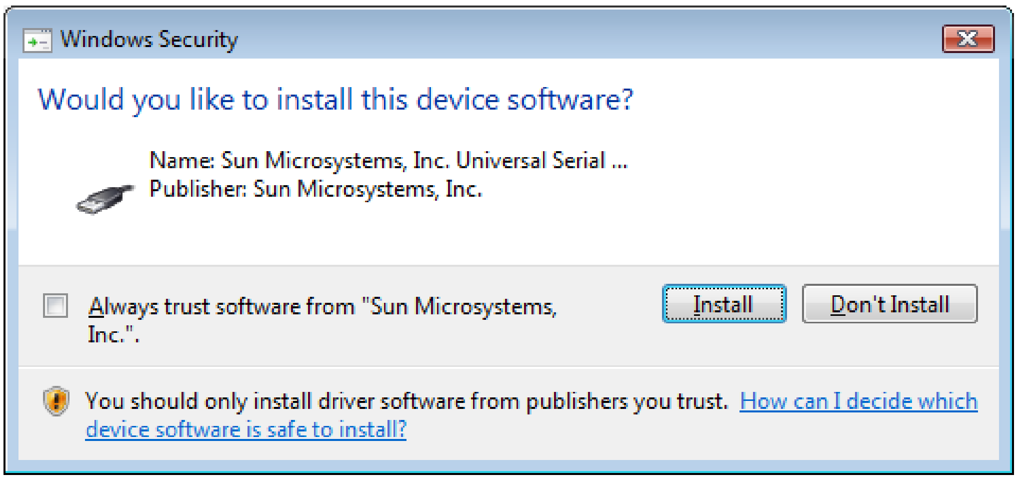
Would you like to install this device software?

Name: Sun Microsystems, Inc. Network adapters
Publisher: Sun Microsystems, Inc.

Always trust software from "Sun Microsystems, Inc.".

You should only install driver software from publishers you trust. How can I decide which device software is safe to install?

Figure 20



Windows Security

Would you like to install this device software?

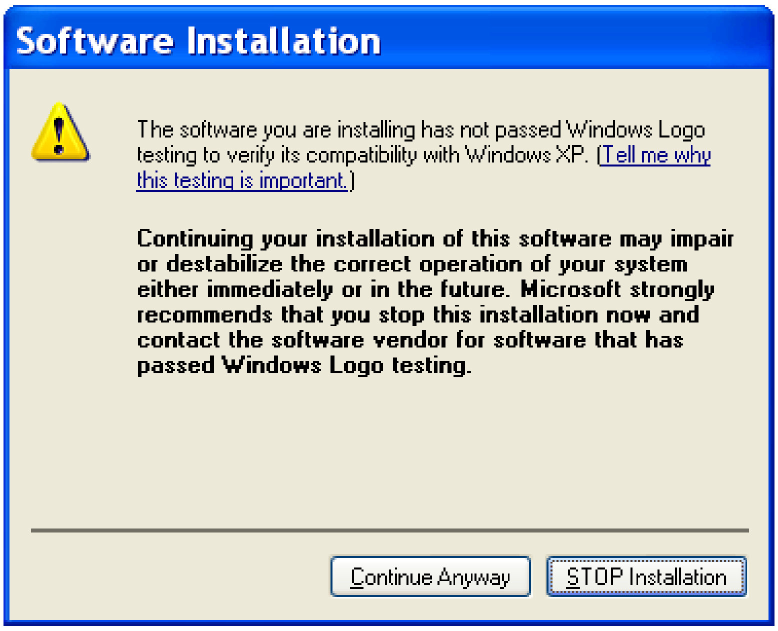
Name: Sun Microsystems, Inc. Universal Serial ...
Publisher: Sun Microsystems, Inc.

Always trust software from "Sun Microsystems, Inc.".

You should only install driver software from publishers you trust. How can I decide which device software is safe to install?

Figure 21

In Windows XP, please select to **Continue Anyway** when questioned.

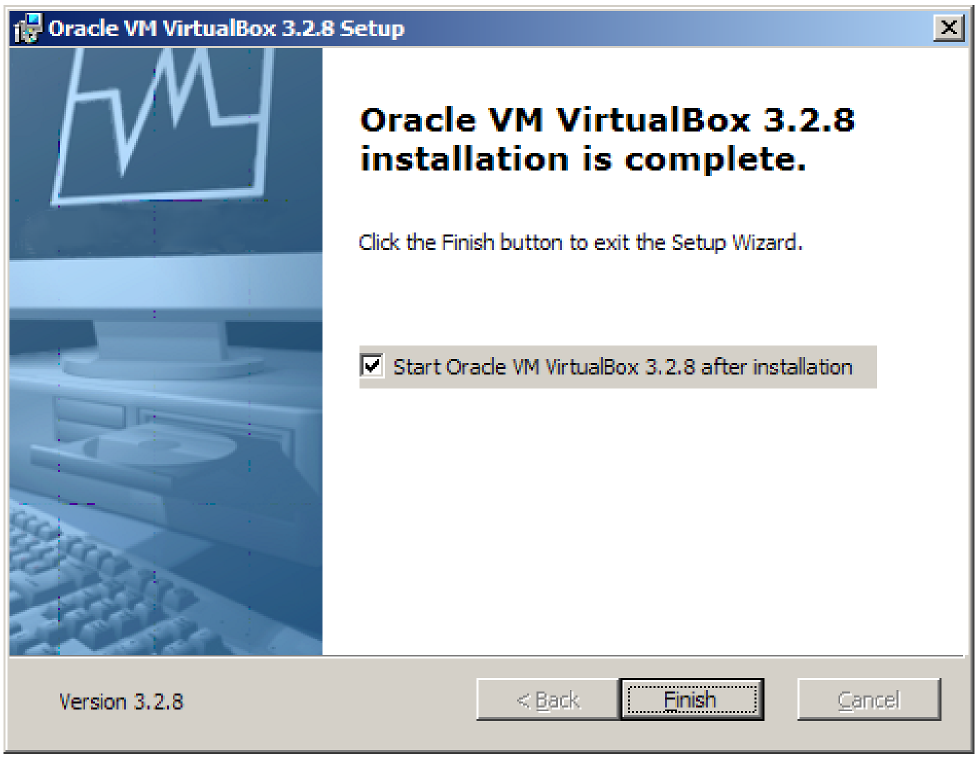


Software Installation

The software you are installing has not passed Windows Logo testing to verify its compatibility with Windows XP. (Tell me why this testing is important.)

Continuing your installation of this software may impair or destabilize the correct operation of your system either immediately or in the future. Microsoft strongly recommends that you stop this installation now and contact the software vendor for software that has passed Windows Logo testing.

Figure 22



Oracle VM VirtualBox 3.2.8 installation is complete.

Click the Finish button to exit the Setup Wizard.

☑ Start Oracle VM VirtualBox 3.2.8 after installation

Version 3.2.8

Figure 23

Leave '*Start Oracle xVM VirtualBox after installation*' checked and press **Finish** to exit.

VirtualBox has been installed!

Proceed to Step c 'Installing the Ubuntu Client' on page 32.

Step b: Installing VirtualBox (on a Mac)

If you are installing VirtualBox on a PC please go to page 16.

You will see the following screen. Follow the steps of the installation wizard as shown below:

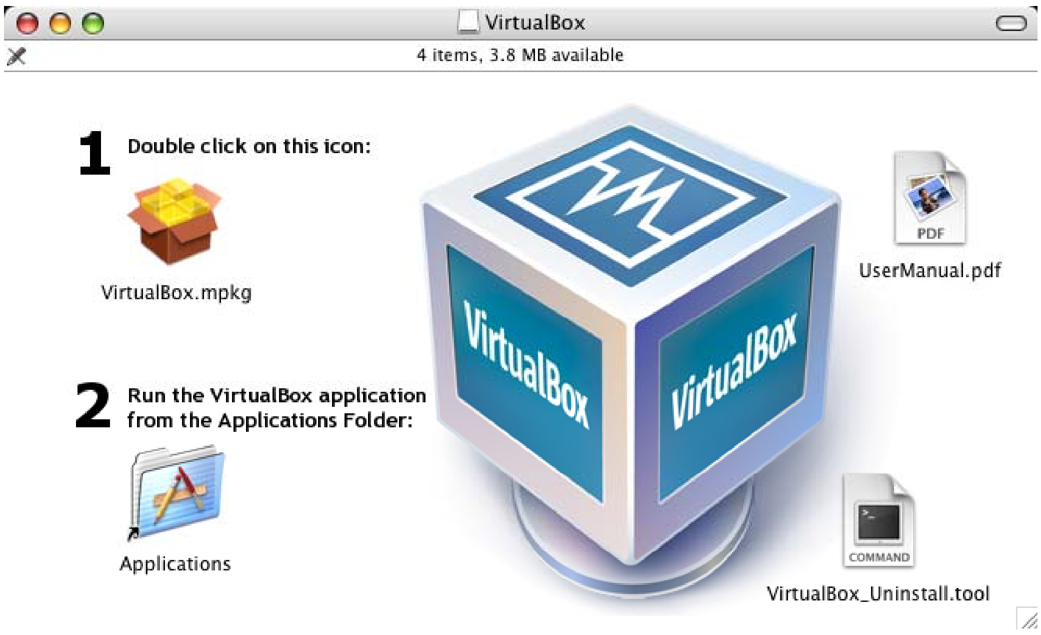
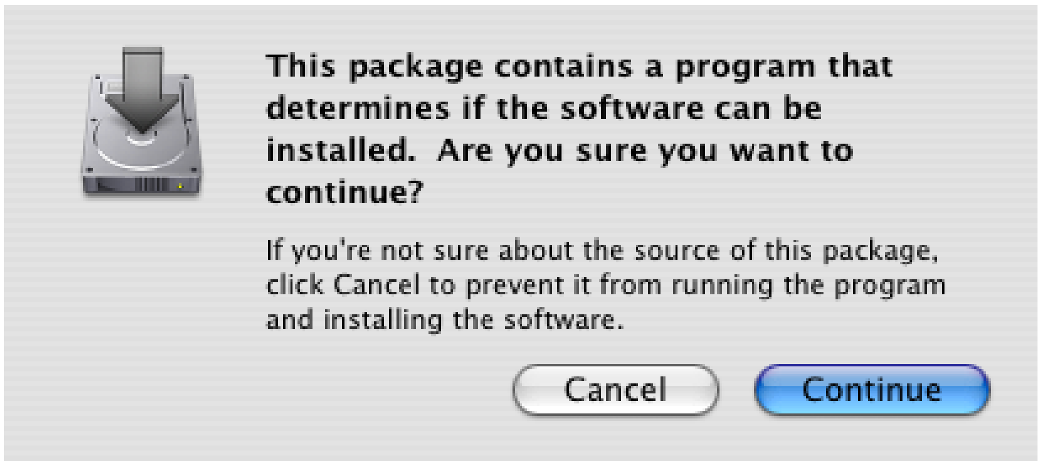

VirtualBox
4 items, 3.8 MB available
1 Double click on this icon:
VirtualBox.mpkg
2 Run the VirtualBox application from the Applications Folder:
Applications
UserManual.pdf
VirtualBox_Uninstall.tool

Figure 24

Double click the **box icon** from step 1.

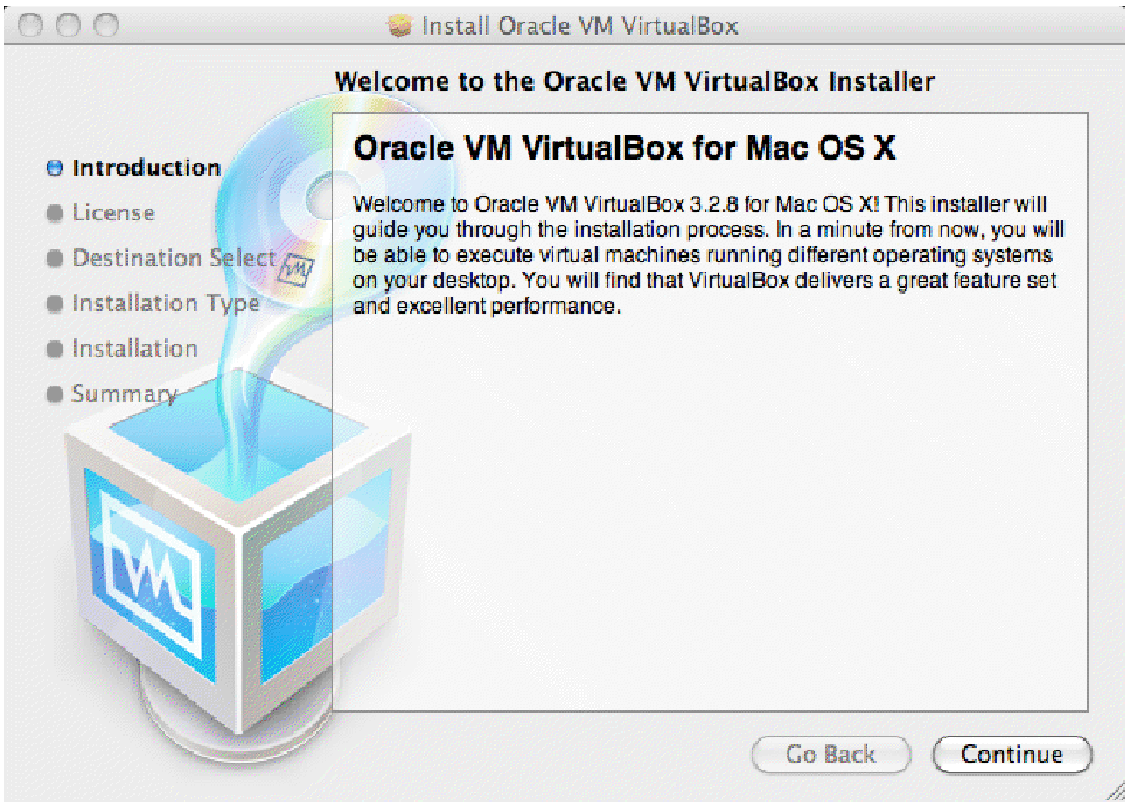


This package contains a program that determines if the software can be installed. Are you sure you want to continue?

If you're not sure about the source of this package, click Cancel to prevent it from running the program and installing the software.

Figure 25

Press **Continue**



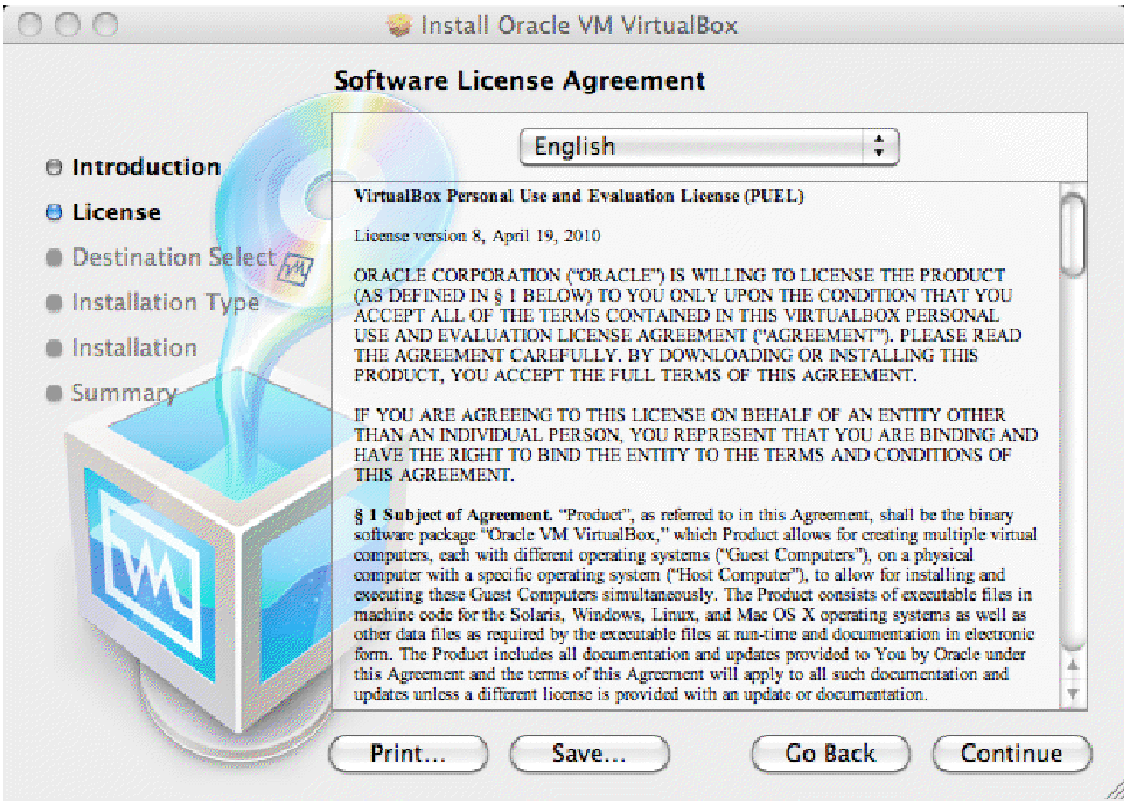
Welcome to the Oracle VM VirtualBox Installer

Oracle VM VirtualBox for Mac OS X

Welcome to Oracle VM VirtualBox 3.2.8 for Mac OS X! This installer will guide you through the installation process. In a minute from now, you will be able to execute virtual machines running different operating systems on your desktop. You will find that VirtualBox delivers a great feature set and excellent performance.

Figure 26

Press **Continue**



VirtualBox Personal Use and Evaluation License (PUEL)

License version 8, April 19, 2010

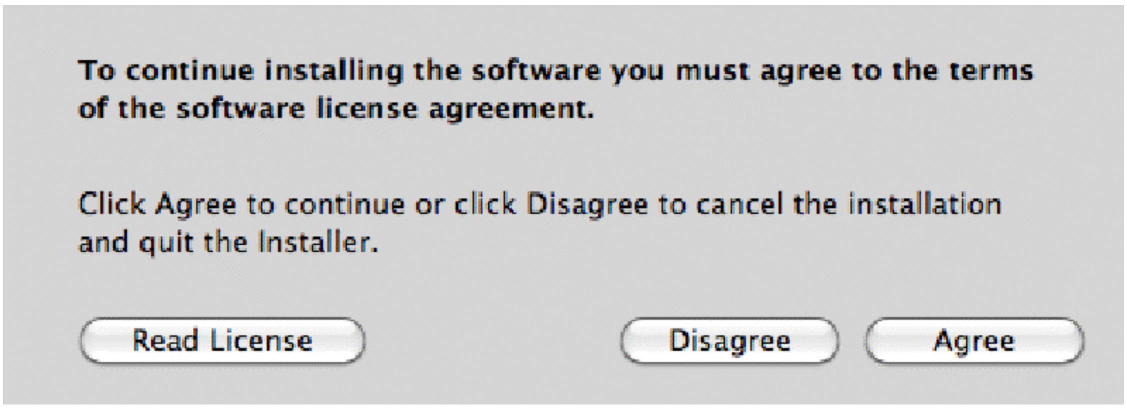
ORACLE CORPORATION ("ORACLE") IS WILLING TO LICENSE THE PRODUCT (AS DEFINED IN § 1 BELOW) TO YOU ONLY UPON THE CONDITION THAT YOU ACCEPT ALL OF THE TERMS CONTAINED IN THIS VIRTUALBOX PERSONAL USE AND EVALUATION LICENSE AGREEMENT ("AGREEMENT"). PLEASE READ THE AGREEMENT CAREFULLY. BY DOWNLOADING OR INSTALLING THIS PRODUCT, YOU ACCEPT THE FULL TERMS OF THIS AGREEMENT.

IF YOU ARE AGREEING TO THIS LICENSE ON BEHALF OF AN ENTITY OTHER THAN AN INDIVIDUAL PERSON, YOU REPRESENT THAT YOU ARE BINDING AND HAVE THE RIGHT TO BIND THE ENTITY TO THE TERMS AND CONDITIONS OF THIS AGREEMENT.

§ 1 Subject of Agreement. "Product", as referred to in this Agreement, shall be the binary software package "Oracle VM VirtualBox," which Product allows for creating multiple virtual computers, each with different operating systems ("Guest Computers"), on a physical computer with a specific operating system ("Host Computer"), to allow for installing and executing these Guest Computers simultaneously. The Product consists of executable files in machine code for the Solaris, Windows, Linux, and Mac OS X operating systems as well as other data files as required by the executable files at run-time and documentation in electronic form. The Product includes all documentation and updates provided to You by Oracle under this Agreement and the terms of this Agreement will apply to all such documentation and updates unless a different license is provided with an update or documentation.

Figure 27

Press **Continue**

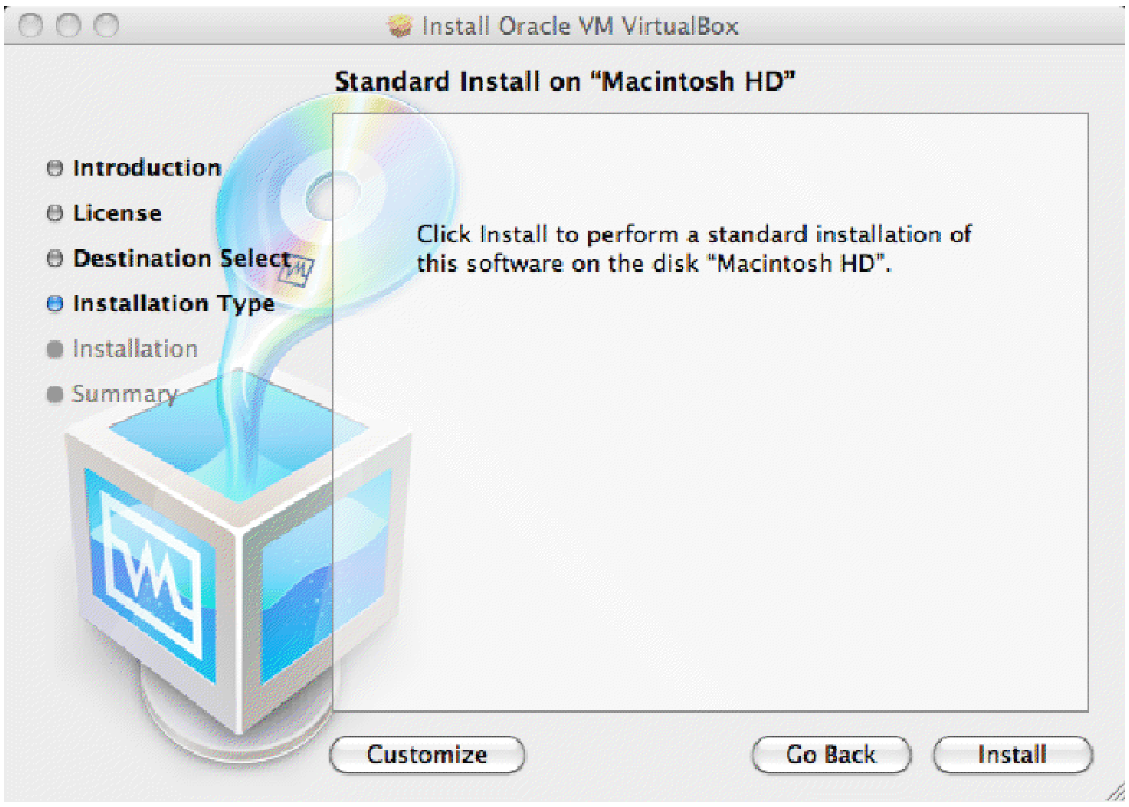


To continue installing the software you must agree to the terms of the software license agreement.

Click Agree to continue or click Disagree to cancel the installation and quit the Installer.

Figure 28

Press **Agree**

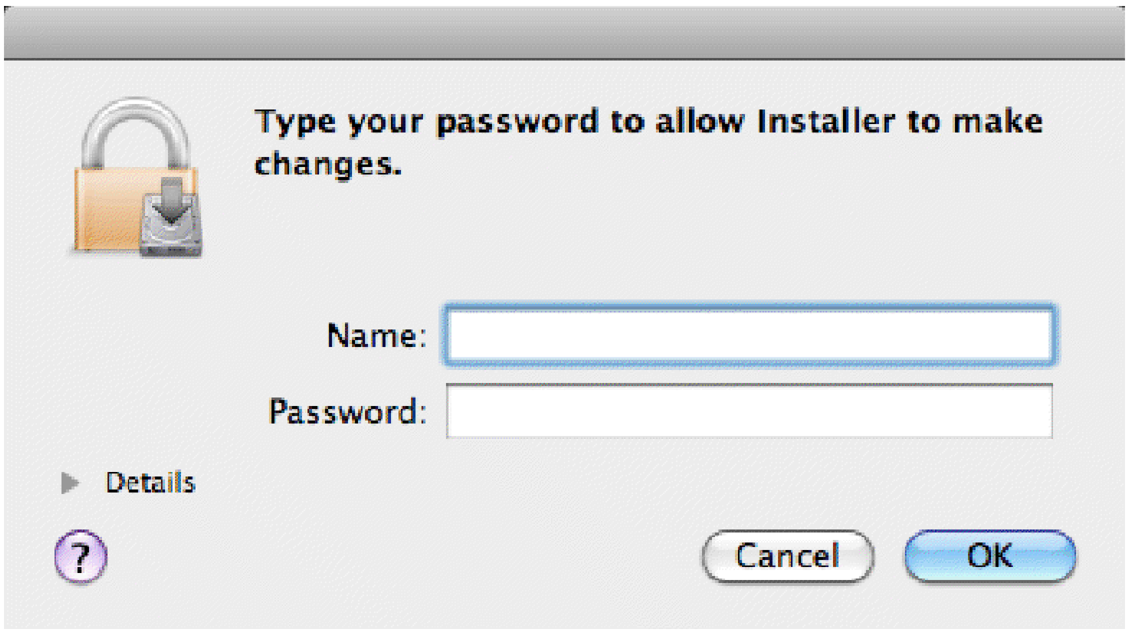


Standard Install on "Macintosh HD"

Click Install to perform a standard installation of this software on the disk "Macintosh HD".

Figure 29

Press **Install**

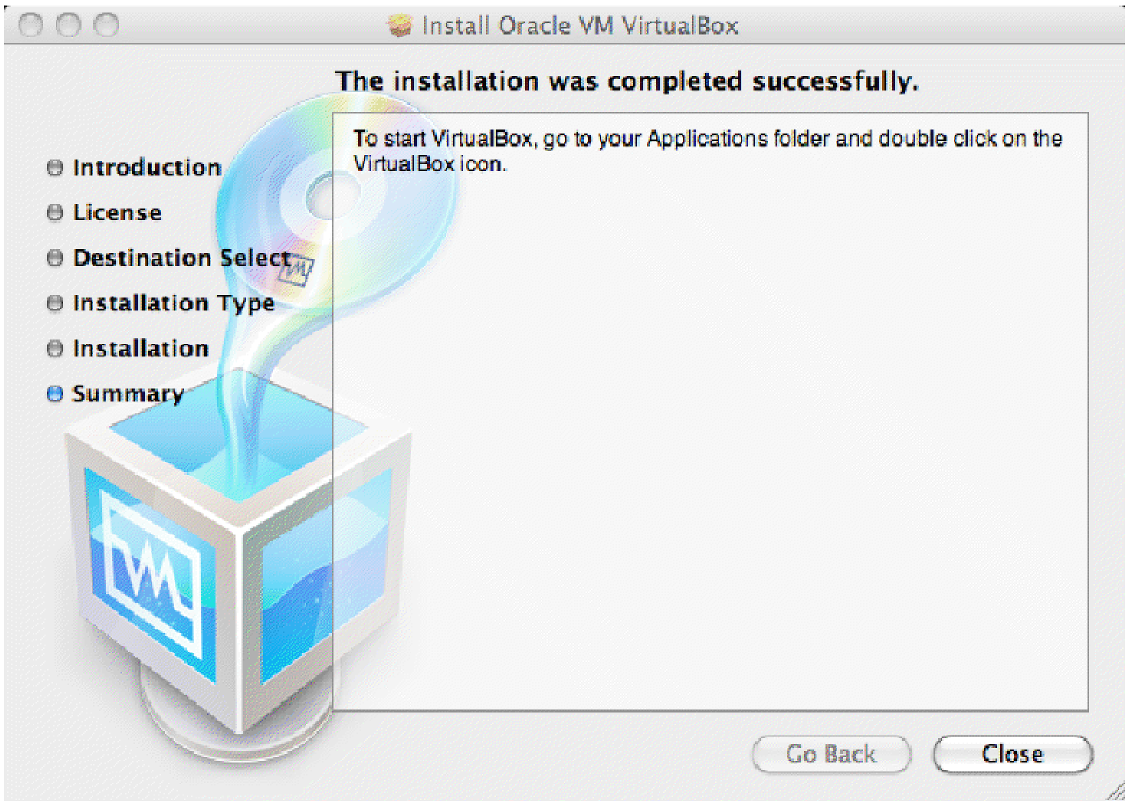

Type your password to allow Installer to make changes.

Name:

Password:

Figure 30

Enter your administrator’s name and password. And then press **OK**.



The installation was completed successfully.

To start VirtualBox, go to your Applications folder and double click on the VirtualBox icon.

Figure 31

Press **Close** to exit the install.

Return to the installation menu, Figure 12 and select option #2. This will copy the Ubuntu client to your computer. Note: It may take 10 minutes or more for the copy to finish. Please have patience.

When the copy concludes, return to the Applications window, as shown below, and double-click on the VirtualBox icon.

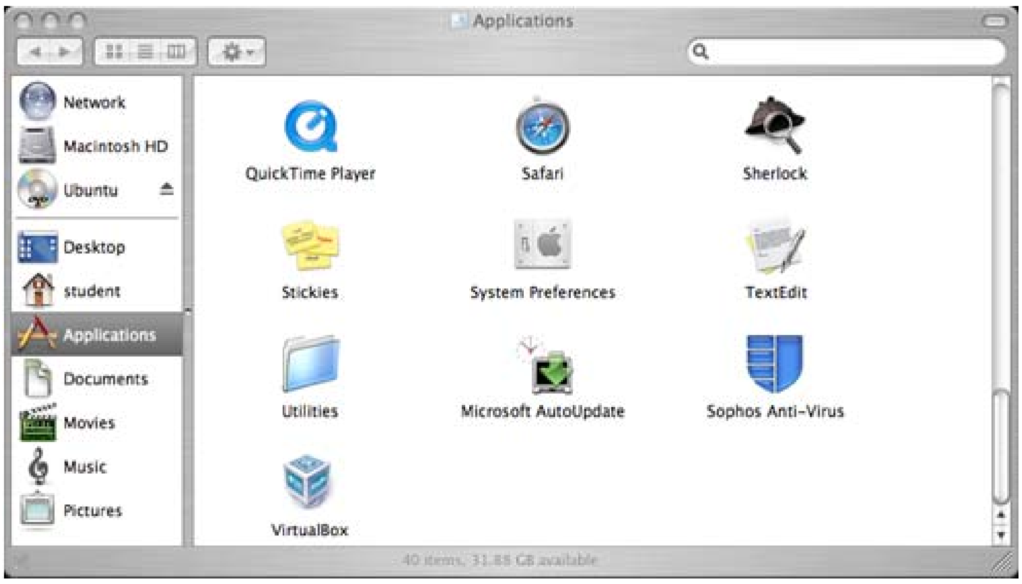


Figure 32

Go to the **Applications** menu to launch VirtualBox and continue with the next section (*Step c 'Installing the Ubuntu Client'*.)

Step c: Installing the Ubuntu Client

You are now ready to start the VirtualBox Console and install the Ubuntu Linux client on your virtual machine. Although we have used Windows Vista screenshots to illustrate the installation, the steps are the same on Windows XP/Windows 7 and Mac OS X.

On a Windows PC, start VirtualBox by clicking on its icon.

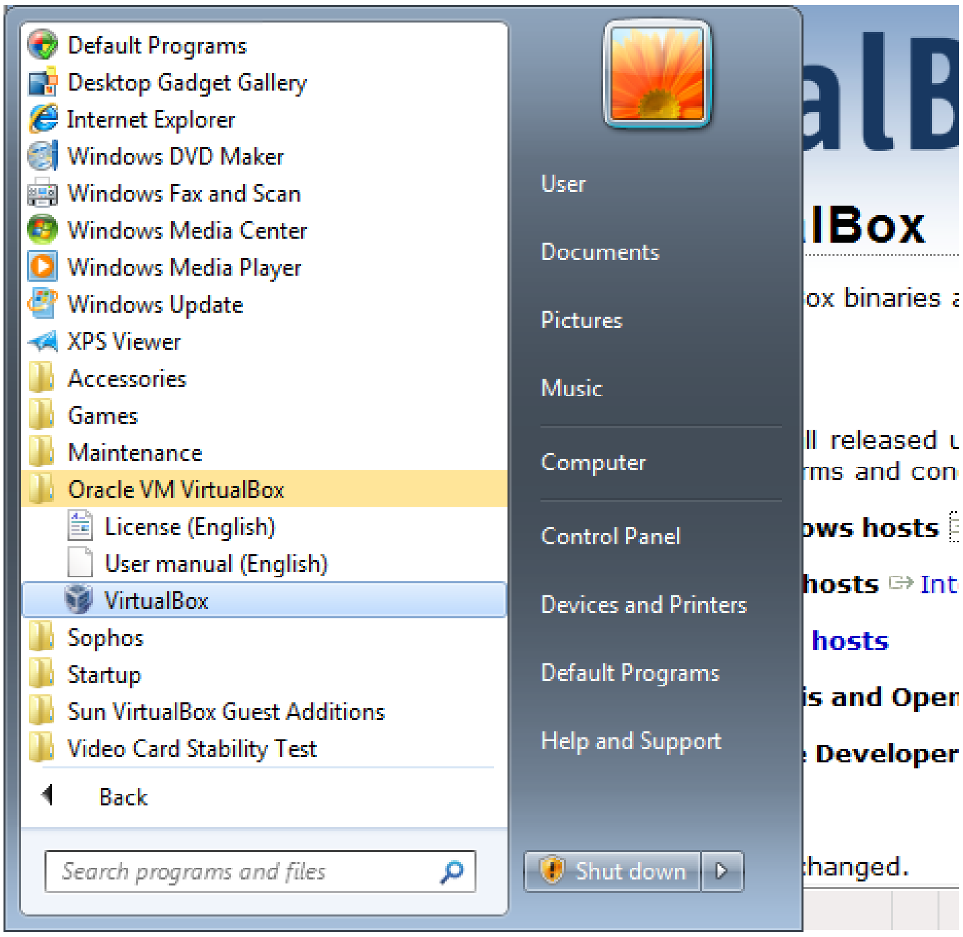
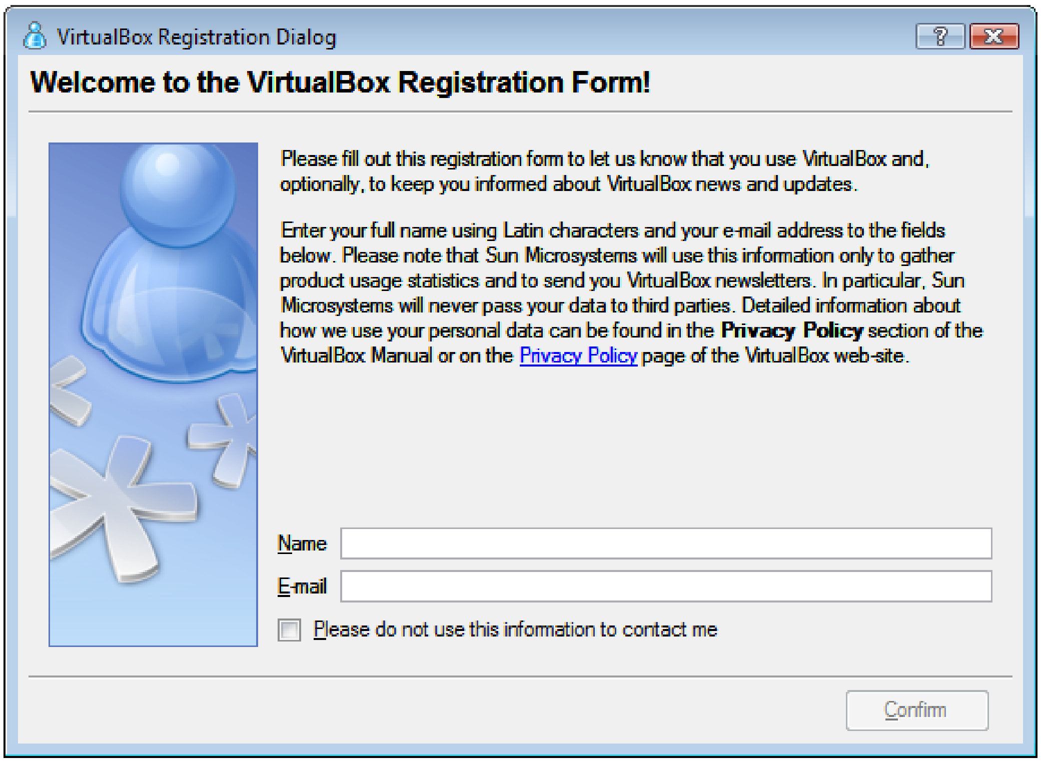


Figure 33

On a Mac OS X double-click on the VirtualBox icon in the Applications window (Figure 32.)

Follow the steps on the next page.



Welcome to the VirtualBox Registration Form!

Please fill out this registration form to let us know that you use VirtualBox and, optionally, to keep you informed about VirtualBox news and updates.

Enter your full name using Latin characters and your e-mail address to the fields below. Please note that Sun Microsystems will use this information only to gather product usage statistics and to send you VirtualBox newsletters. In particular, Sun Microsystems will never pass your data to third parties. Detailed information about how we use your personal data can be found in the **Privacy Policy** section of the VirtualBox Manual or on the Privacy Policy page of the VirtualBox web-site.

Name
E-mail
Please do not use this information to contact me

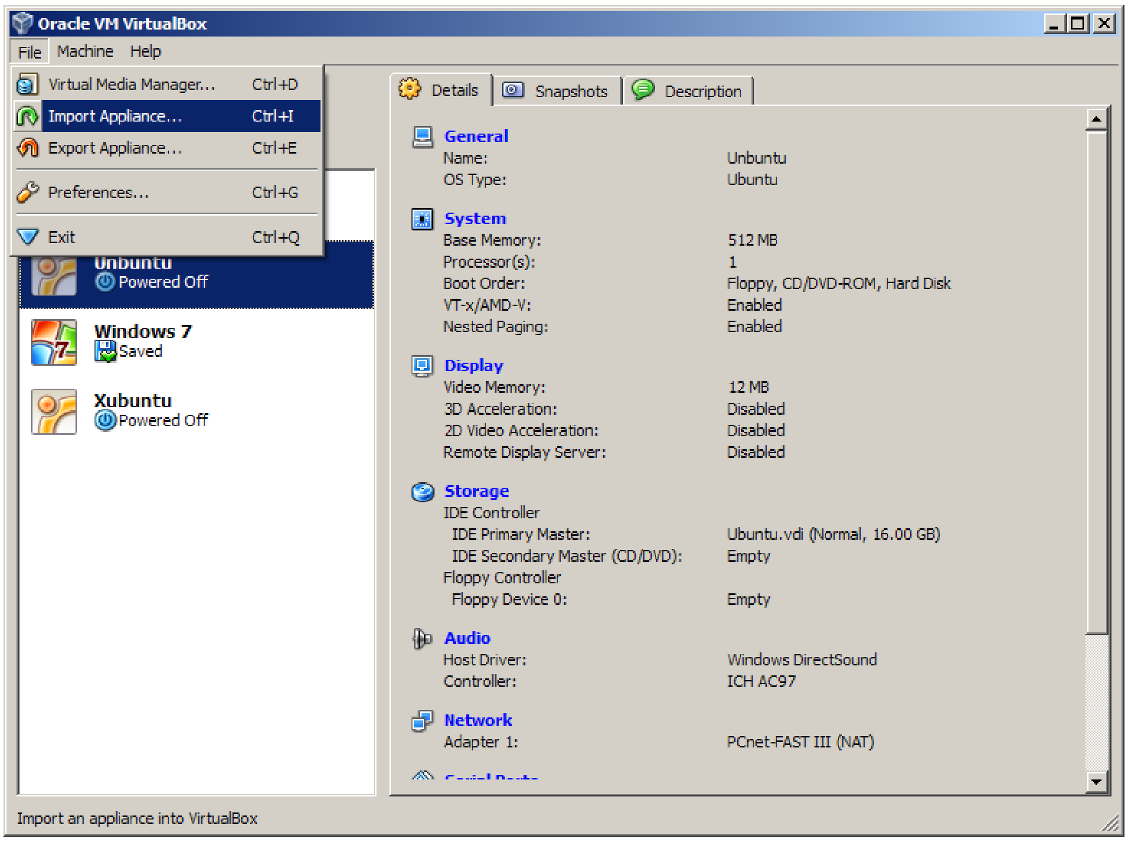
Figure 34

You may register VirtualBox if you so desire. If you prefer not to, you may click the **red X** to close the VirtualBox registration dialog window.

You need to add a Virtual Machine to VirtualBox. When a Virtual Machine is transferred from one computer to another, it is referred to as an Appliance. The Appliance you will be installing has a disk with existing data on it. The installation DVD contains a virtual disk containing the Ubuntu Operating System with support for Code::Blocks and nxtOSEK.

We now wish to setup VirtualBox to use the Appliance contained on the DVD.

Select *File/Import Appliance* from the VirtualBox console, as shown below.



General
Name: Unbuntu
OS Type: Ubuntu

System
Base Memory: 512 MB
Processor(s): 1
Boot Order: Floppy, CD/DVD-ROM, Hard Disk
VT-x/AMD-V: Enabled
Nested Paging: Enabled

Display
Video Memory: 12 MB
3D Acceleration: Disabled
2D Video Acceleration: Disabled
Remote Display Server: Disabled

Storage
IDE Controller
IDE Primary Master: Ubuntu.vdi (Normal, 16.00 GB)
IDE Secondary Master (CD/DVD): Empty
Floppy Controller
Floppy Device 0: Empty

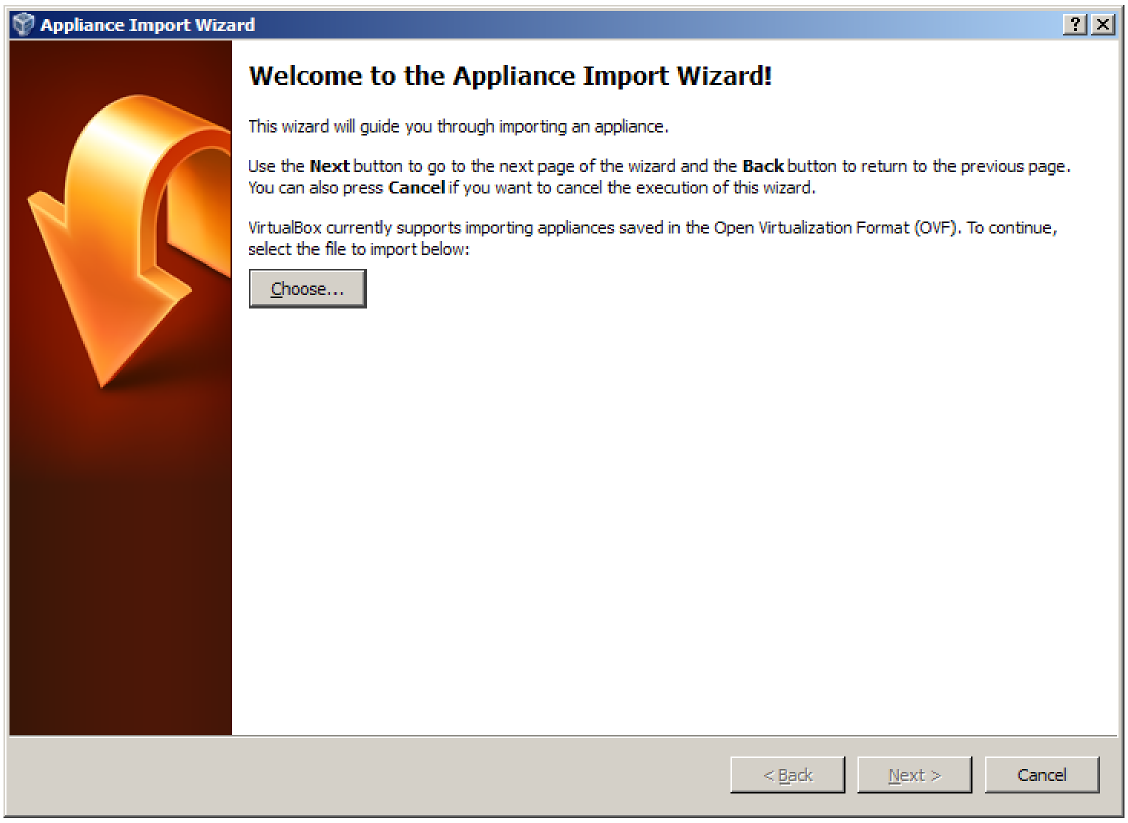
Audio
Host Driver: Windows DirectSound
Controller: ICH AC97

Network
Adapter 1: PCnet-FAST III (NAT)

Import an appliance into VirtualBox

Figure 35

Select the **Choose** button (middle on the left) as shown below.



Welcome to the Appliance Import Wizard!

This wizard will guide you through importing an appliance.

Use the **Next** button to go to the next page of the wizard and the **Back** button to return to the previous page. You can also press **Cancel** if you want to cancel the execution of this wizard.

VirtualBox currently supports importing appliances saved in the Open Virtualization Format (OVF). To continue, select the file to import below:

Figure 36

Double-click on Ubuntu.ovf file in the Appliance directory (as shown in Figure 37)

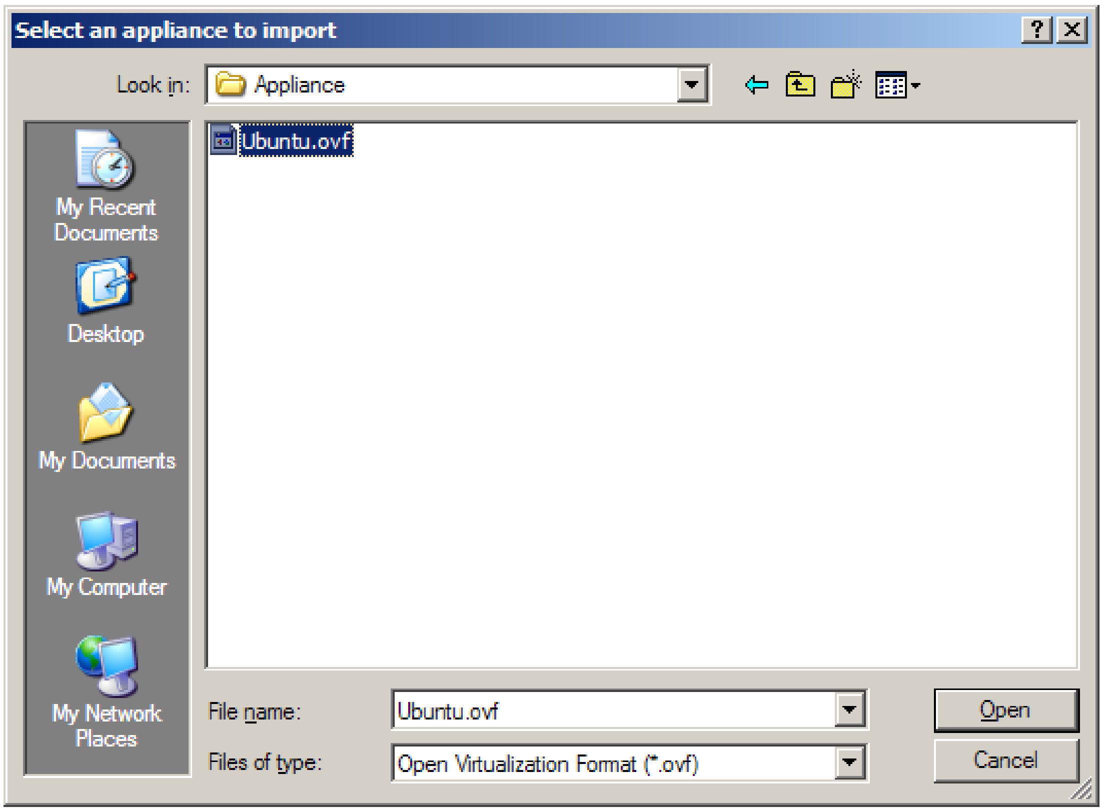
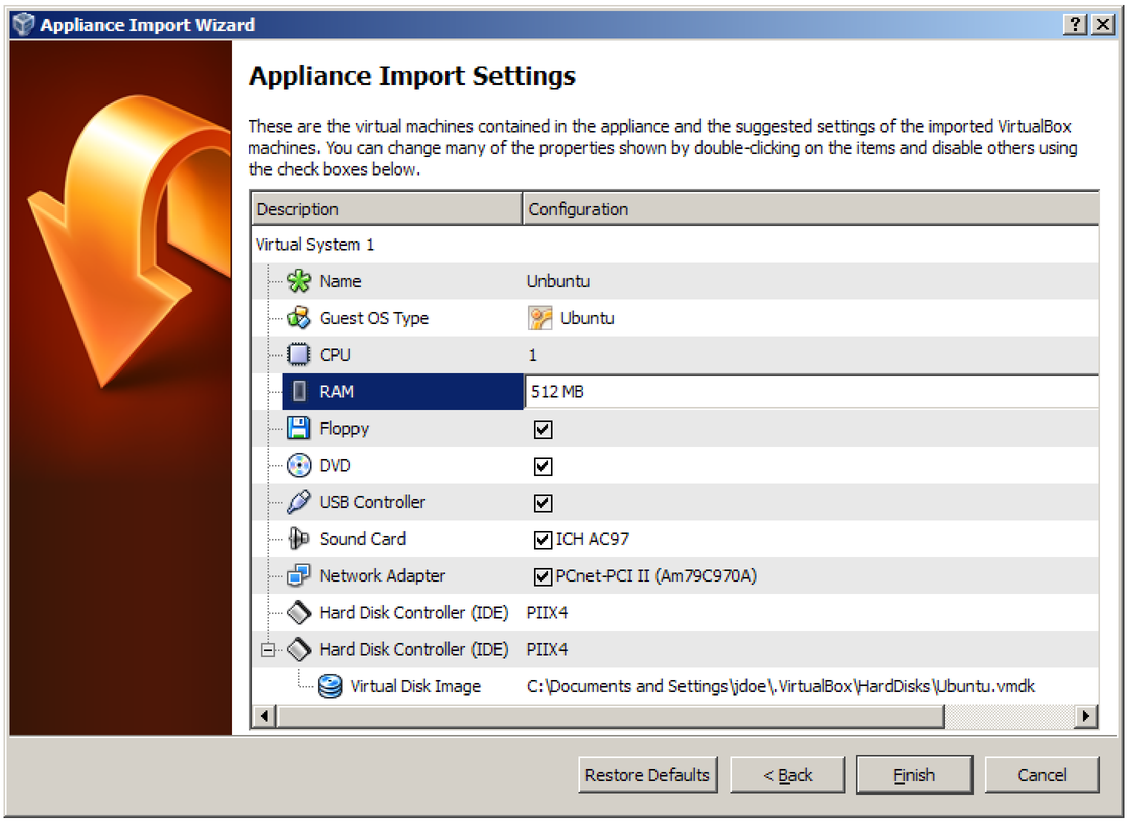


Figure 37

The default is to have **512** MB for the amount of RAM for the virtual machine. If you have only 512 of real RAM in your system (which incidentally is quite low), select **256** MB to be used for the virtual machine. See the screenshot below. To change the amount of RAM, double click the Configuration Value and change the number.



Appliance Import Settings

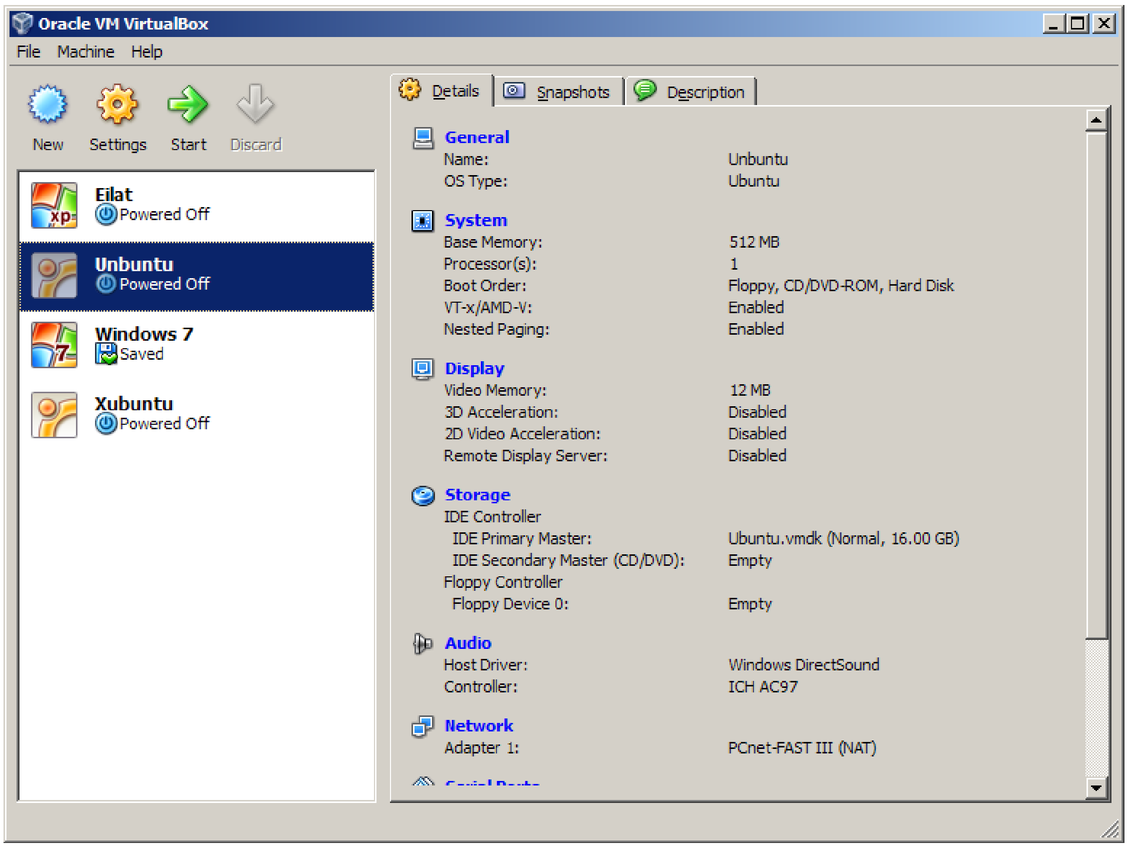
These are the virtual machines contained in the appliance and the suggested settings of the imported VirtualBox machines. You can change many of the properties shown by double-clicking on the items and disable others using the check boxes below.

Description	Configuration
Virtual System 1	
Name	Unbuntu
Guest OS Type	Ubuntu
CPU	1
RAM	512 MB
Floppy	✓
DVD	✓
USB Controller	✓
Sound Card	✓ ICH AC97
Network Adapter	✓ PCnet-PCI II (Am79C970A)
Hard Disk Controller (IDE)	PIIX4
Hard Disk Controller (IDE)	PIIX4
Virtual Disk Image	C:\Documents and Settings\jdoe\.VirtualBox\HardDisks\Ubuntu.vmdk

Figure 38

Select **Finish,** to begin the install. It will take several minutes and the Appliance is setup.

Congratulations! You have now completed the installation of VirtualBox and the Ubuntu client.



General

Name:	Unbuntu
OS Type:	Ubuntu

System

Base Memory:	512 MB
Processor(s):	1
Boot Order:	Floppy, CD/DVD-ROM, Hard Disk
VT-x/AMD-V:	Enabled
Nested Paging:	Enabled

Display

Video Memory:	12 MB
3D Acceleration:	Disabled
2D Video Acceleration:	Disabled
Remote Display Server:	Disabled

Storage

IDE Controller	
IDE Primary Master:	Ubuntu.vmdk (Normal, 16.00 GB)
IDE Secondary Master (CD/DVD):	Empty
Floppy Controller	
Floppy Device 0:	Empty

Audio

Host Driver:	Windows DirectSound
Controller:	ICH AC97

Network

Adapter 1:	PCnet-FAST III (NAT)

Figure 39

You have successfully built a system for Ubuntu!

- Select "*Start*" (the green arrow) to begin using the system.
- To login to the system, the username and password are both: *student*. (The root password is also *student*.)
- As you run a virtual machine, you may get various message windows from VirtualBox. If you wish you may check the box to disable those messages.

6.3. How to Uninstall the Software

To uninstall from Windows go to the *Control Panel* and then choose *Add/Remove Programs*. From the choices that appear select *remove VirtualBox*.

To uninstall from Macs, run the installer again and then double click on the *Uninstall* icon.

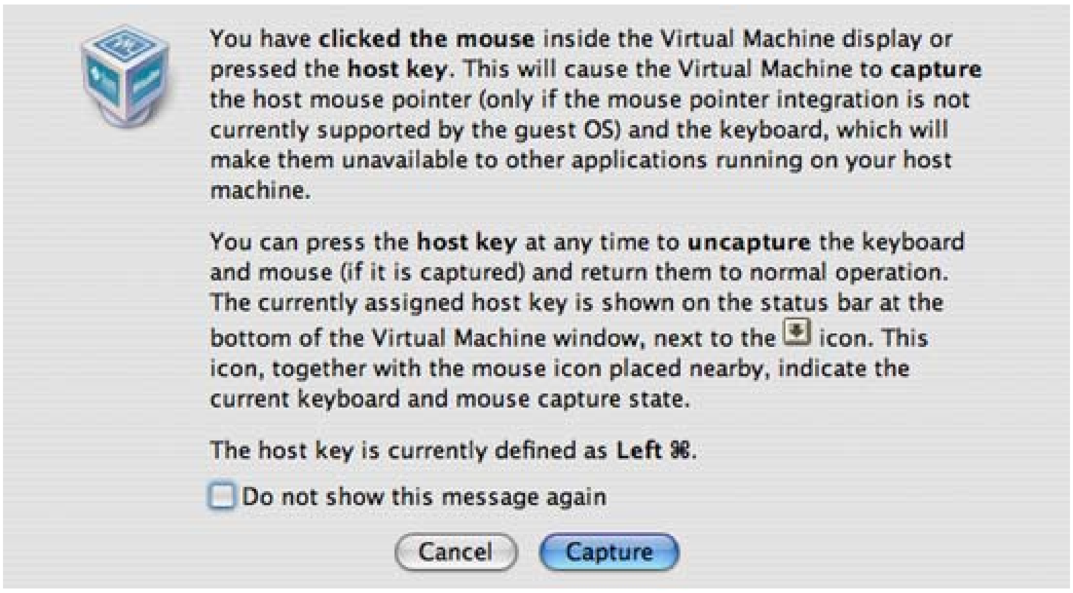
In both cases, find the location of the .*VMDK* file and then delete it.

Section 7: First Use

Once you have completed the installation, you are ready to begin programming for the Lego Robot. In the following examples we will use screen shots from a Windows XP host, however, the first screen shots for those using a Mac OS X host will be comparable. Once you begin using the Ubuntu Linux guest, the screen shots will be identical on all systems.

Special note for Max OS X users of VirtualBox

As you are no doubt aware you on a single button Mac you '*right-click*' by holding down the *Apple key* and simultaneously pressing the mouse button. However, while in VirtualBox, the *Apple key* is considered to be a special key. In order not to be annoyed by the following screen:



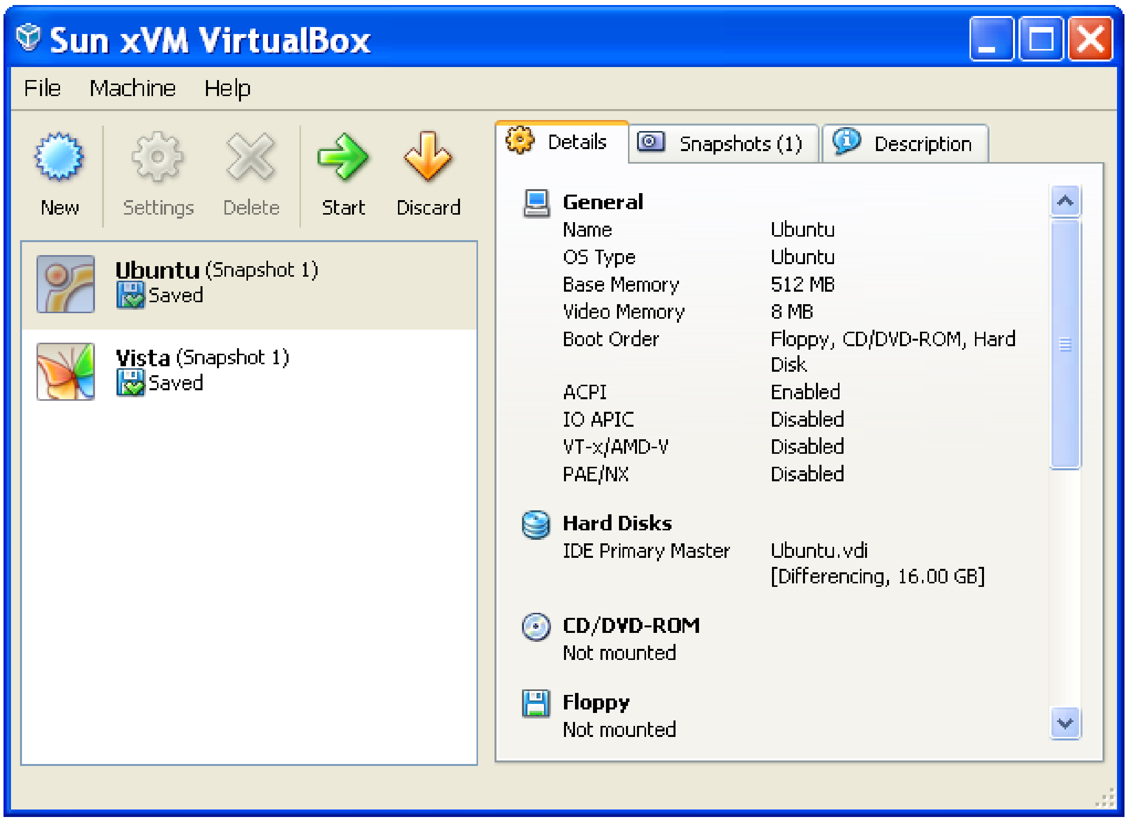
You have **clicked the mouse** inside the Virtual Machine display or pressed the **host key**. This will cause the Virtual Machine to **capture** the host mouse pointer (only if the mouse pointer integration is not currently supported by the guest OS) and the keyboard, which will make them unavailable to other applications running on your host machine.

You can press the **host key** at any time to **uncapture** the keyboard and mouse (if it is captured) and return them to normal operation. The currently assigned host key is shown on the status bar at the bottom of the Virtual Machine window, next to the icon. This icon, together with the mouse icon placed nearby, indicate the current keyboard and mouse capture state.

The host key is currently defined as **Left ⌘**.

Check the "*Do not show this message again*" option.

1. Start the VirtualBox software (click on the VirtualBox icon on your *desktop* or on the menu). The resulting window is shown below (Figure 40).



General

Name	Ubuntu
OS Type	Ubuntu
Base Memory	512 MB
Video Memory	8 MB
Boot Order	Floppy, CD/DVD-ROM, Hard Disk
ACPI	Enabled
IO APIC	Disabled
VT-x/AMD-V	Disabled
PAE/NX	Disabled

Hard Disks

IDE Primary Master	Ubuntu.vdi [Differencing, 16.00 GB]

CD/DVD-ROM
Not mounted

Floppy
Not mounted

Figure 40 - Sun VirtualBox console

Your screen may appear somewhat differently, depending on the configuration of your hardware and the number of guest operating systems installed.

2. Start the Ubuntu Linux guest by clicking on the appropriately named icon in the console. After some time you will be asked to enter the username and your password. These have both been set to '*student*' (all lowercase). After some additional time and appropriate sound effects you will be confronted with the Ubuntu Linux desktop, shown below (Figure 41).

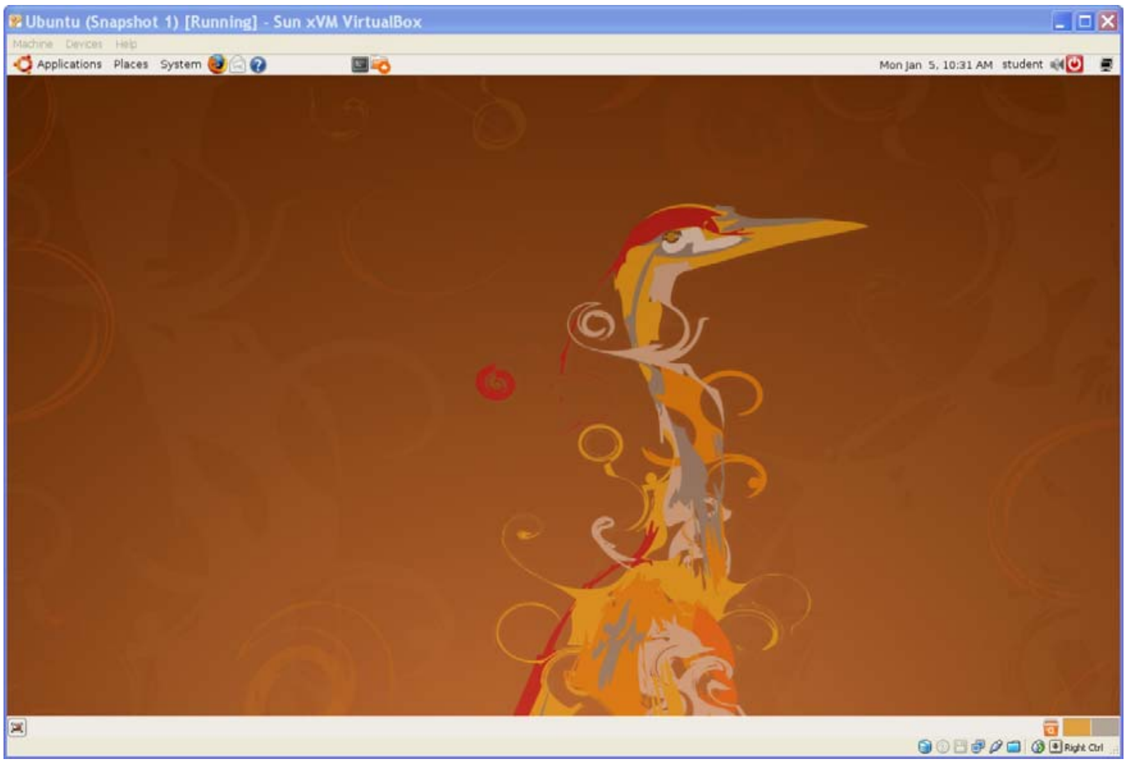
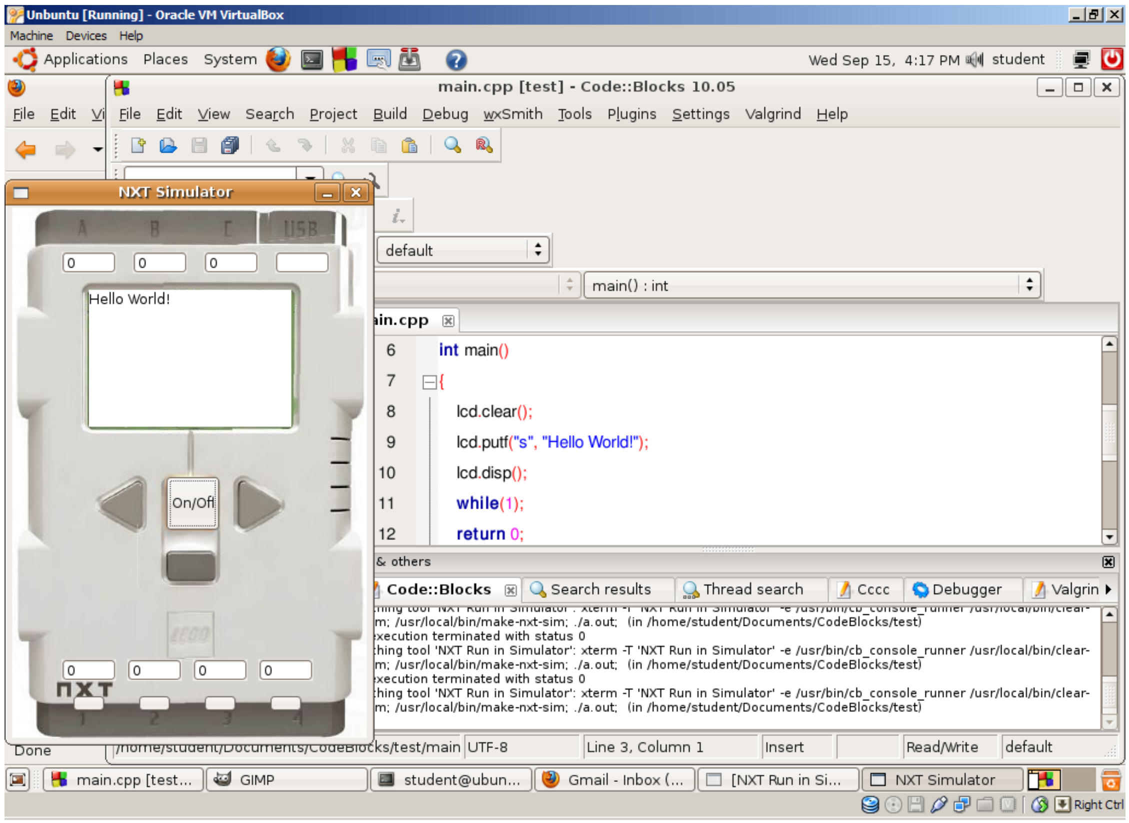

Ubuntu (Snapshot 1) [Running] - Sun xVM VirtualBox
Machine Devices Help
Applications Places System
Mon Jan 5, 10:31 AM student
Right Ctrl

Figure 41 - Ubuntu Linux Desktop

You may at this point open the window full screen since from this point on we will be working only within Ubuntu. (Of course you may adjust the window to whatever size is convenient in order for you to work in the host and the guest simultaneously. Isn't virtualization wonderful?!)

3. Start the Code::Blocks IDE. (Using the mouse click on *Applications/Programming/CodeBlocks IDE*)

4. If you wish to use the simulator, you must first have a source code program in Code::Blocks, which can then be run under the simulator by clicking on Tools/Run in Simulator (See Fig. 48).


Unbuntu [Running] - Oracle VM VirtualBox
Machine Devices Help
Applications Places System
Wed Sep 15, 4:17 PM student
main.cpp [test] - Code::Blocks 10.05
File Edit View Search Project Build Debug wxSmith Tools Plugins Settings Valgrind Help
NXT Simulator
A B C USB
Hello World!
On/Off
default
main() : int
ain.cpp
& others
Code::Blocks Search results Thread search Cccc Debugger Valgrin
Done
UTF-8 Line 3, Column 1 Insert Read/Write default
main.cpp [test... GIMP student@ubun... Gmail - Inbox (... [NXT Run in Si... NXT Simulator
Right Ctrl

```
int main()
{
    lcd.clear();
    lcd.putf("s", "Hello World!");
    lcd.disp();
    while(1);
    return 0;
```

```
m; /usr/local/bin/make-nxt-sim; ./a.out;  (in /home/student/Documents/CodeBlocks/test)
xecution terminated with status 0
hing tool 'NXT Run in Simulator': xterm -T 'NXT Run in Simulator' -e /usr/bin/cb_console_runner /usr/local/bin/clear-
m; /usr/local/bin/make-nxt-sim; ./a.out;  (in /home/student/Documents/CodeBlocks/test)
xecution terminated with status 0
hing tool 'NXT Run in Simulator': xterm -T 'NXT Run in Simulator' -e /usr/bin/cb_console_runner /usr/local/bin/clear-
m; /usr/local/bin/make-nxt-sim; ./a.out;  (in /home/student/Documents/CodeBlocks/test)
```

/home/student/Documents/CodeBlocks/test/main

Figure 42 - Ubuntu Desktop with the Code::Blocks and nxtSIM applications

Press the *ON/OFF* button on the simulator. You should see the output from the program on the LCD screen.

Section 8: The Ubuntu Linux Desktop

Learning to use Linux is an essential component of the education of the programmer. A detailed introduction to Linux is beyond the scope of this manual. Students wishing to learn more about Linux should consult one of the many excellent textbooks and website that are devoted to Linux in general and its Ubuntu variant in particular. In the Appendix we have presented several sources to get you started.

Having said the above, a detailed knowledge of Linux is not necessary for you to program the Lego Robot. In this section we take a brief look at the Ubuntu Linux desktop. In many ways it is similar to the desktop used in Windows XP/Vista or Apple Mac OS X. You select items using a *left-mouse click* and you use *right-mouse* click to open a context sensitive menu from which further selections may be made.

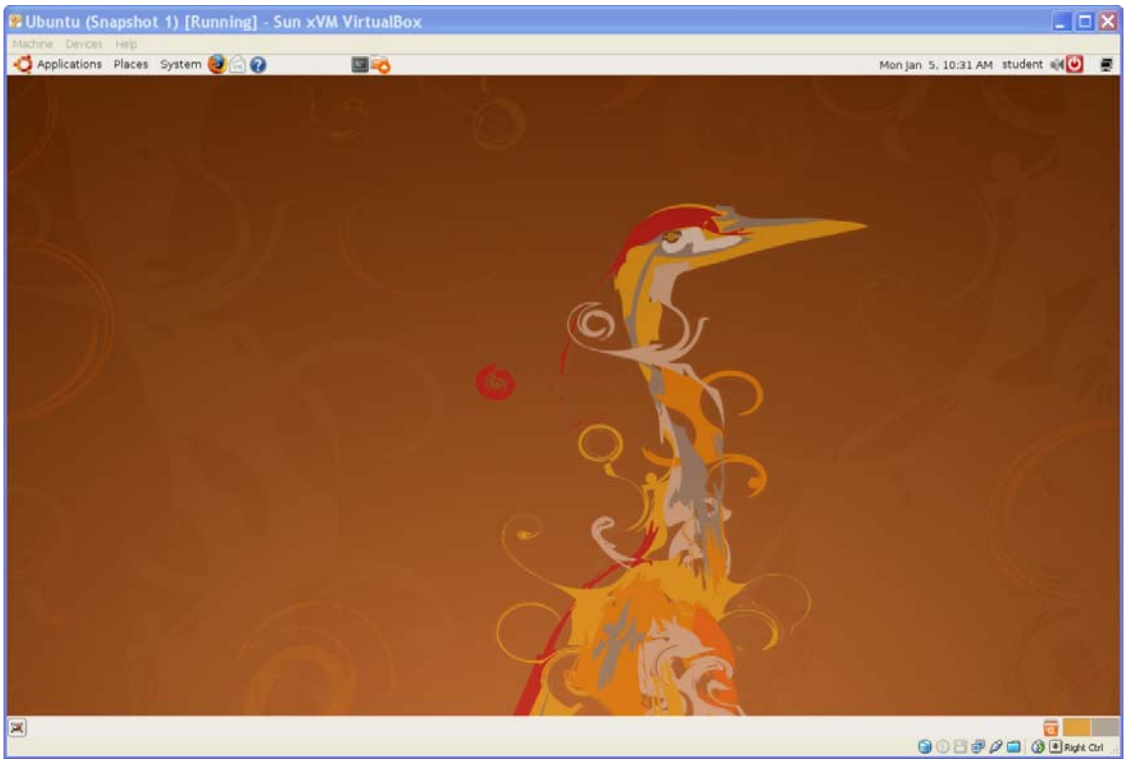


Figure 43 - The Ubuntu Desktop

The desktop is made up of several panels. At the top of the desktop you will see:

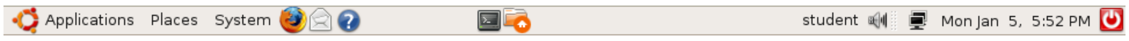


Figure 44 - Ubuntu Desktop Top Panel

From left to right we find:

- *Applications* menu – providing access to the programs installed on your computer
- *Places* menu – providing access to various locations on your computer including files, standard folders, the network as well as system drives.
- *Systems* menu – providing options for configuration of your preferences, system administration and help.
- The middle section contains several program icons allowing access to commonly used applications. You may add additional applications by right-clicking on the panel.
- The right side of the panel contains the current user, a volume control, a network status icon, the date and time, and the power button.
- When updates are available an additional icon appears. Click on that icon and a wizard will guide you through the update process.

Items on the panel may be rearranged and additional items may be added.

At the bottom of the desktop you will see:

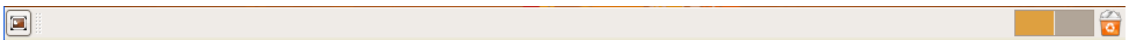


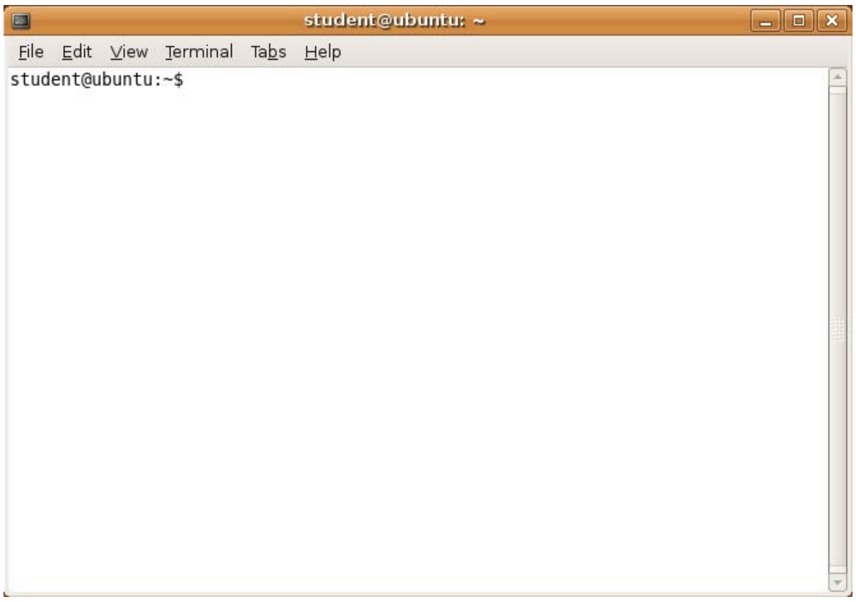
Figure 45 - Ubuntu Desktop Bottom Panel

- The icon on the far left is used to hide and show all windows on the desktop.
- Ubuntu allows the user to have virtual desktops (as if you would have multiple monitors). The squares display, in miniature, the contents of each desktop. You may move back and forth between virtual desktops by clicking the appropriate square.
- At the far right, appears the ubiquitous trash bin.
- As applications are opened, a button will appear for that application on the center portion of this panel.

The central portion of the screen contains your wallpaper (which may be changed) as well as any open application windows.

Additional information about the desktop can be found here: *http://www.techotopia.com/index.php/Ubuntu_Desktop_Essentials* .

For our purposes all interaction with the Ubuntu Linux system will take place using the graphical user interface described above. Advanced Linux users will want to take advantage of the *Linux terminal*, shown below. The terminal may be started by selecting *Applications*/*Accessories*/*Terminal* from the menu on the top panel. Many powerful and complex commands can be issued using the terminal. However, this lies well beyond the scope of this manual. The interested student is urged to consult the resources in Appendix I.



```
student@ubuntu:~$
```

Figure 46 - The Ubuntu Terminal Application

Section 9: How to Shutdown the Ubuntu Client and the VirtualBox Console

1. The Ubuntu guest may be shutdown in one of two ways.

   Method 1 - Press the Ubuntu *Shutdown* button, followed by either *Shutdown* (completely shuts down the Ubuntu virtual machine) or *Hibernate* (saves the current state of the machine before shutting down) on the screen which subsequently appears, *Figure 47*.

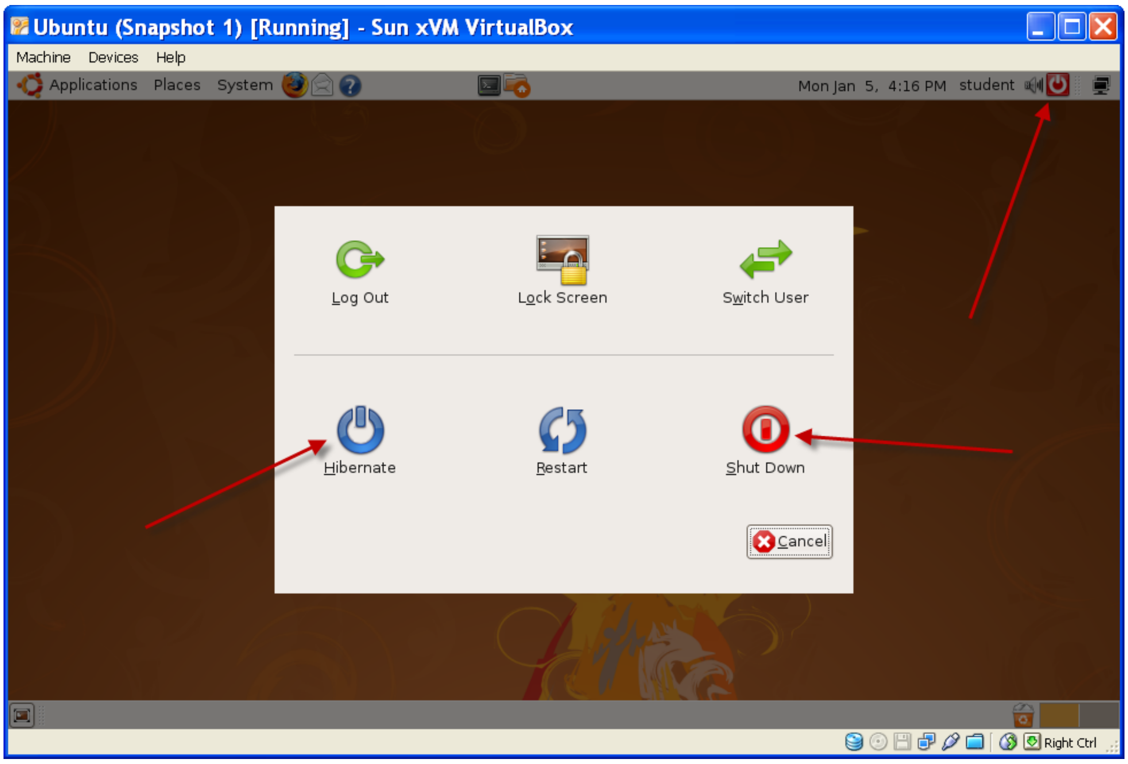
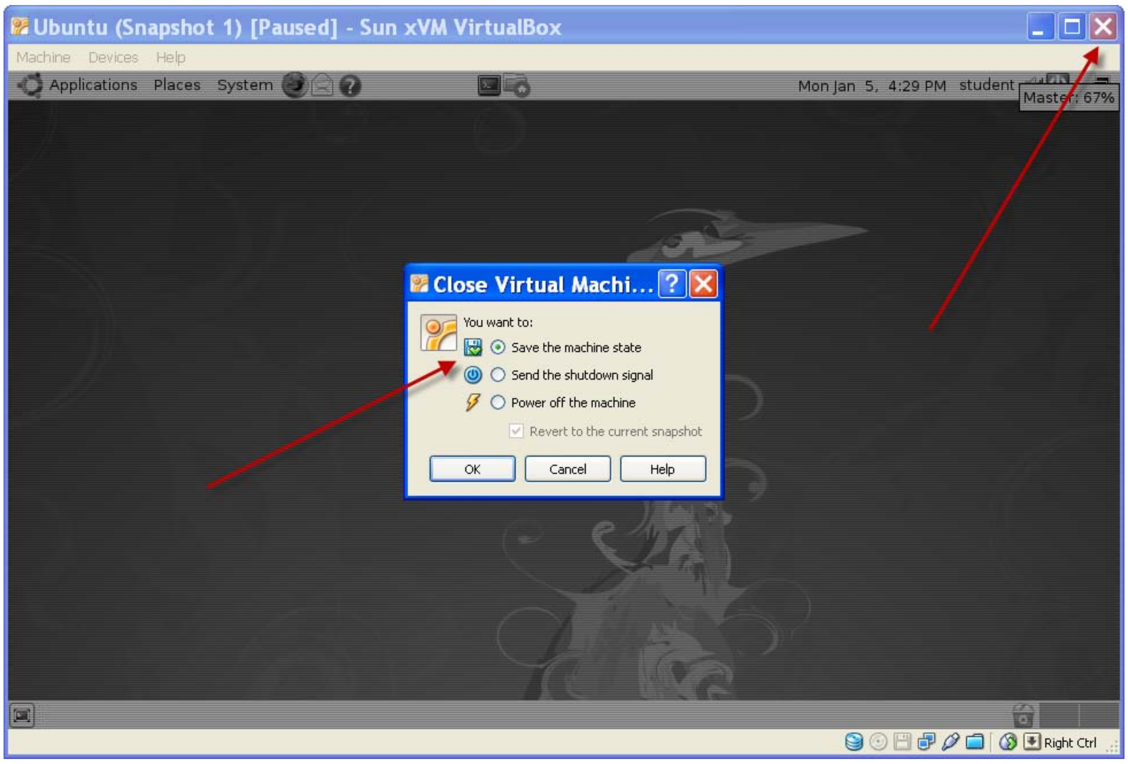

Ubuntu (Snapshot 1) [Running] - Sun xVM VirtualBox
Machine Devices Help
Applications Places System
Mon Jan 5, 4:16 PM student
Log Out
Lock Screen
Switch User
Hibernate
Restart
Shut Down
Cancel
Right Ctrl

Figure 47 - Shutting down Ubuntu (Method 1)

Method 2 – Press the VirtualBox Windows Close button, followed by OK in the dialogue that follows, Figure 48. This will shut down the Ubuntu client, saving its state. The next time you start the Ubuntu client, you will find the virtual machine in the same state as it was when you last shut it down.



You want to:
[✓ Save the machine state]
Send the shutdown signal
Power off the machine
Revert to the current snapshot

Figure 48 - Shutting down Ubuntu (Method 2)

2. Finally close the VirtualBox console.

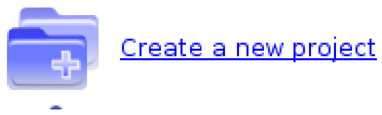
Section 10: First nxtOSEK Program

In this section, we will go through the steps involved in writing a complete C++ program using nxtOSEK for the Lego NXT Robot. We will first test our program on the simulator and then for those who have an actual Lego NXT Robot, we will transfer the program to the Robot. Programming for the Robot is very similar to ordinary programming in C++. You should already be familiar with the Code::Blocks IDE. (You may wish to refresh your memory by reviewing the material in the Code::Blocks manual available here:

http://www.sci.brooklyn.cuny.edu/~goetz/codeblocks/codeblocks-instructions.pdf)

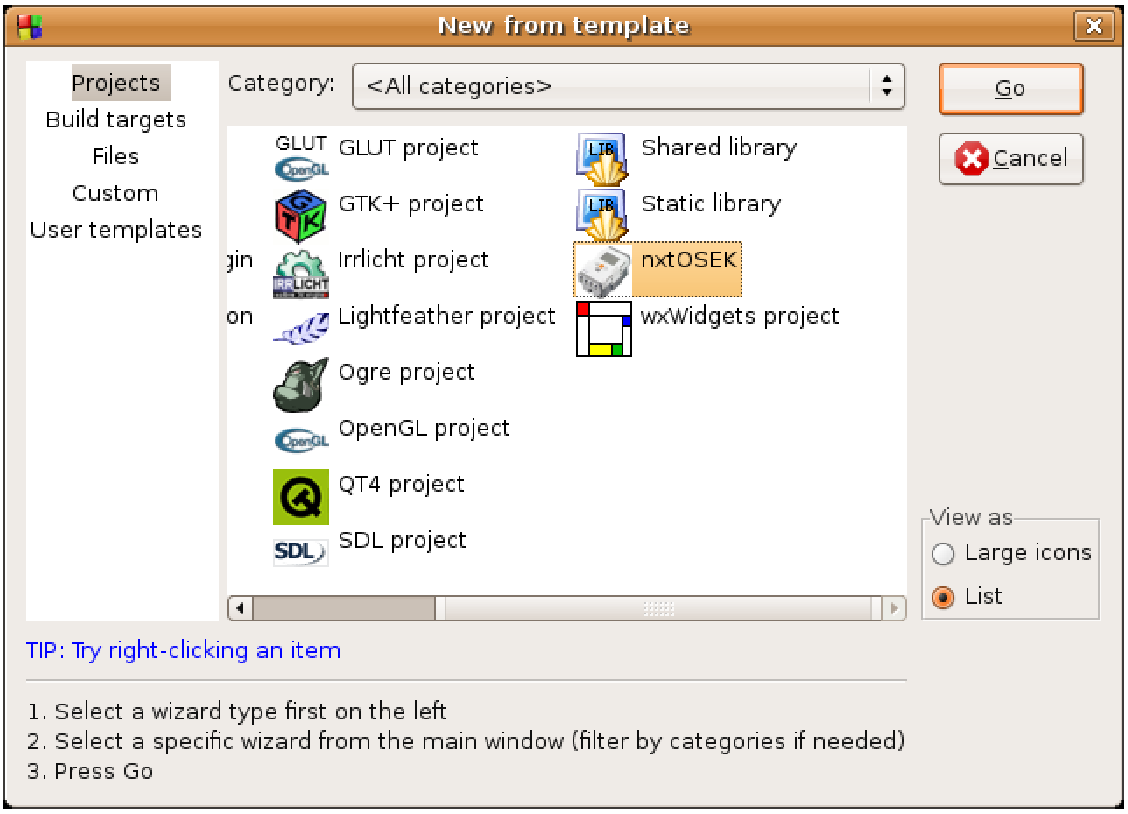
As is traditional, we will write a simple program, called *HelloRobotWorld* to illustrate the basic features of brick programming.

1. Create a new project called *HelloRobotWorld*. Either choose the icon

   from the Code::Blocks IDE opening screen or select *File/New/Project* from the menu.

2. From the resulting screen (Figure 49) select *nxtOSEK*



TIP: Try right-clicking an item

1. Select a wizard type first on the left
2. Select a specific wizard from the main window (filter by categories if needed)
3. Press Go

Figure 49 - Code::Blocks Project Template menu

3. After the usual sequence of Code::Blocks menus, type in the following program as the *main.cpp*:

```
/*   Name: HelloRobotWorld
     Description: A simple program that illustrates
         the features of NxtOSEK.
         The program will:
         1. Display "Hello World" on the LCS
         2. Roll forward and then back
         3. Wait until the either Sensor 1 or
            Sensor 2 is "touched"
*/

// Include files containing the NxtOSEK library functions
// Remember to modify the roboTask.h file as described
   below.
#include "robot.h"
#include "roboTask.h"

Lcd lcd;
Clock c;
```

```
bool running = false;

TouchSensor touch2(PORT_4);
TouchSensor touch1(PORT_1);
Motor motorB(PORT_B);
Motor motorC(PORT_C);

// function prototypes
void hello();
void world();
void roll();

int main()
{
    lcd.clear();        // Clears the display
    hello();            // Display "Hello"
    c.wait(5000);       // Pause for 5 seconds
    world();            // Display "World!"
    c.wait(5000);       // Pause for 5 seconds

    roll();             // Roll forward and backwards

    while(1);           // Run the main()forever
    return 0;
}

// function world displays "Hello" on the LCD
void hello()
{
    lcd.putf("s", "Hello ");
    lcd.disp();
}

// function world displays "World!" on the LCD
void world()
{
    lcd.putf("s", "World!");
    lcd.disp();
}
```

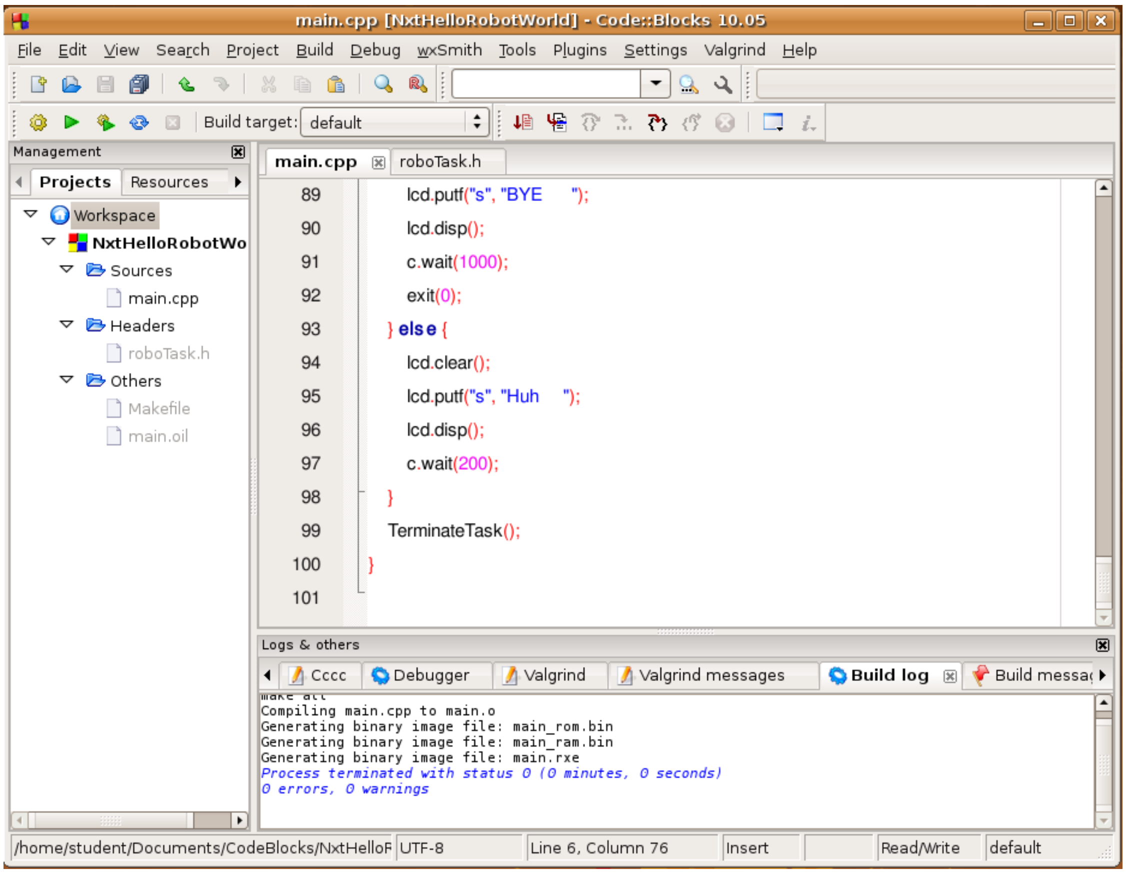
```
// function roll instructs the robot to roll forward
// and then backward.
void roll()
{
    running = true;
    motorB.setPWM(50);
    motorC.setPWM(50);
    c.wait(5000);
    motorB.setPWM(-50);
    motorC.setPWM(-50);
    c.wait(5000);

    // stops the wheels
    motorB.reset();
    motorC.reset();
}

/* The task displays "HUH" until such time as either
   sensor 1 or sensor 2 is activated.
   Display "OUCH" whenever sensor 2 is activated.
   Display "BYE" and end the program when sensor 1 is
   activated.
*/
task1()
{
    if (!running)
        return;
    if (touch1.isPressed()) {
        lcd.clear();
        lcd.putf("s", "OUCH     ");
        lcd.disp();
        c.wait(200);
    } else if (touch2.isPressed()) {
        lcd.clear();
        lcd.putf("s", "BYE      ");
        lcd.disp();
        c.wait(1000);
        exit(0);
    } else {
        lcd.clear();
        lcd.putf("s", "Huh     ");
        lcd.disp();
        c.wait(200);
    }
    TerminateTask();
}
```

4. The *roboTask.h* file is created automatically by the wizard. It is designed to allow up to ten tasks running simultaneously. For every task that your *main*() function defines you must **delete** (or **comment out**) the corresponding task from the header file as shown below.

```
/* This header file is a requirement of the OSEK
   operating system, which generates a number of tasks.
   The number is defined in an OSEK OIL configuration
   file. Therefore, if more tasks are required, you must
   rebuild the system.
  When writing programs that utilize tasks, these must be
   extracted from this header file and placed in your
   main(). The tasks that are extracted must be deleted
   from this file or it will generate a redefinition
   error when the program is compiled.
*/


#ifndef ROBOTASK_H_INCLUDED
#define ROBOTASK_H_INCLUDED

//      /* Task1 */
//      task1()
//      {
//           TerminateTask();
//      } // end task1

/* Task2 */
task2()
{
    TerminateTask();
}

/* Task3 */
task3()
{
    TerminateTask();
}

/* Task4 */
task4()
{
    TerminateTask();
}
```

```
/* Task5 */
task5()
{
    TerminateTask();
}

/* Task6 */
task6()
{
    TerminateTask();
}

/* Task7 */
task7()
{
    TerminateTask();
}

/* Task8 */
task8()
{
    TerminateTask();
}

/* Task9 */
task9()
{
    TerminateTask();
}

/* Task10 */
task10()
{
    TerminateTask();
}

#endif // ROBOTASK_H_INCLUDED
```

5. Use the Code::Blocks IDE to cross compile the program for the Lego Brick. Select *Build/Compile Current File*. If your program contains no errors your screen should look like Figure 50 at the end of the *Build/Compile* process.


main.cpp [NxtHelloRobotWorld] - Code::Blocks 10.05
File Edit View Search Project Build Debug wxSmith Tools Plugins Settings Valgrind Help
Build target: default
Management
Projects Resources
Workspace
NxtHelloRobotWo
Sources
main.cpp
Headers
roboTask.h
Others
Makefile
main.oil
main.cpp roboTask.h
Logs & others
Cccc Debugger Valgrind Valgrind messages Build log Build messa

```
        lcd.putf("s", "BYE     ");
        lcd.disp();
        c.wait(1000);
        exit(0);
    } else {
        lcd.clear();
        lcd.putf("s", "Huh     ");
        lcd.disp();
        c.wait(200);
    }
    TerminateTask();
}
```

```
make all
Compiling main.cpp to main.o
Generating binary image file: main_rom.bin
Generating binary image file: main_ram.bin
Generating binary image file: main.rxe
Process terminated with status 0 (0 minutes, 0 seconds)
0 errors, 0 warnings
```

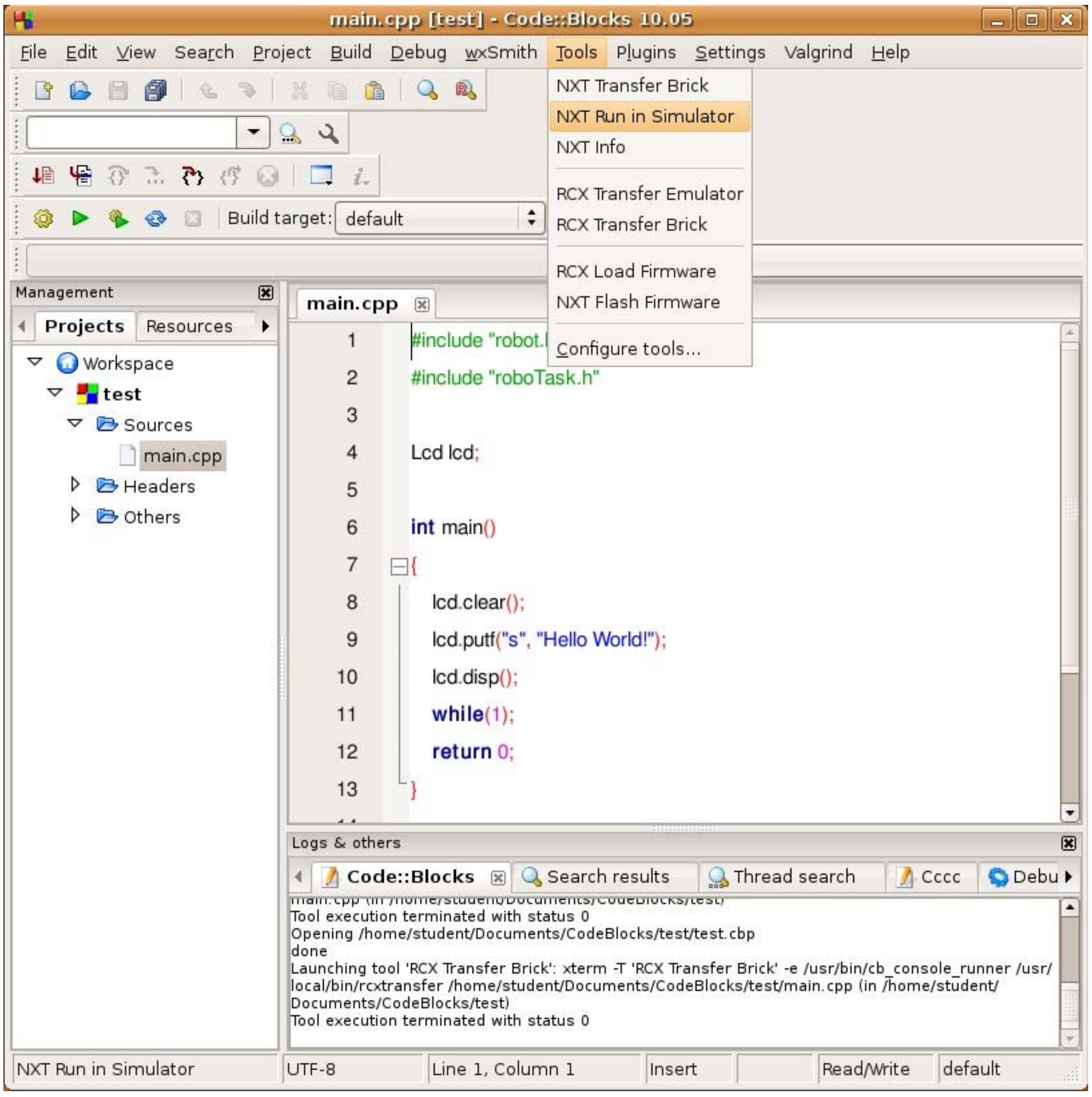
/home/student/Documents/CodeBlocks/NxtHelloF | UTF-8 | Line 6, Column 76 | Insert | Read/Write | default

Figure 50 - Code::Blocks indicating a successful compilation

Be sure not to *Build* (or *Build and Run*) the program (as you might do during ordinary C++ program development. The program we have written is designed to run on the Brick, not on the computer. Building your code in the usual fashion would result in object code and executable code designed for the computer that Code::Blocks is running on and not the computer residing in the Brick. This is certainly not what we want.

Assuming your source code compiled correctly for the Brick emulator, you will find a file called *main.rxe* in your *NxtHelloRobotWorld* project directory. This is the file that would be transferred to the NXT Brick to run on the real robot. This transfer is shown later on.

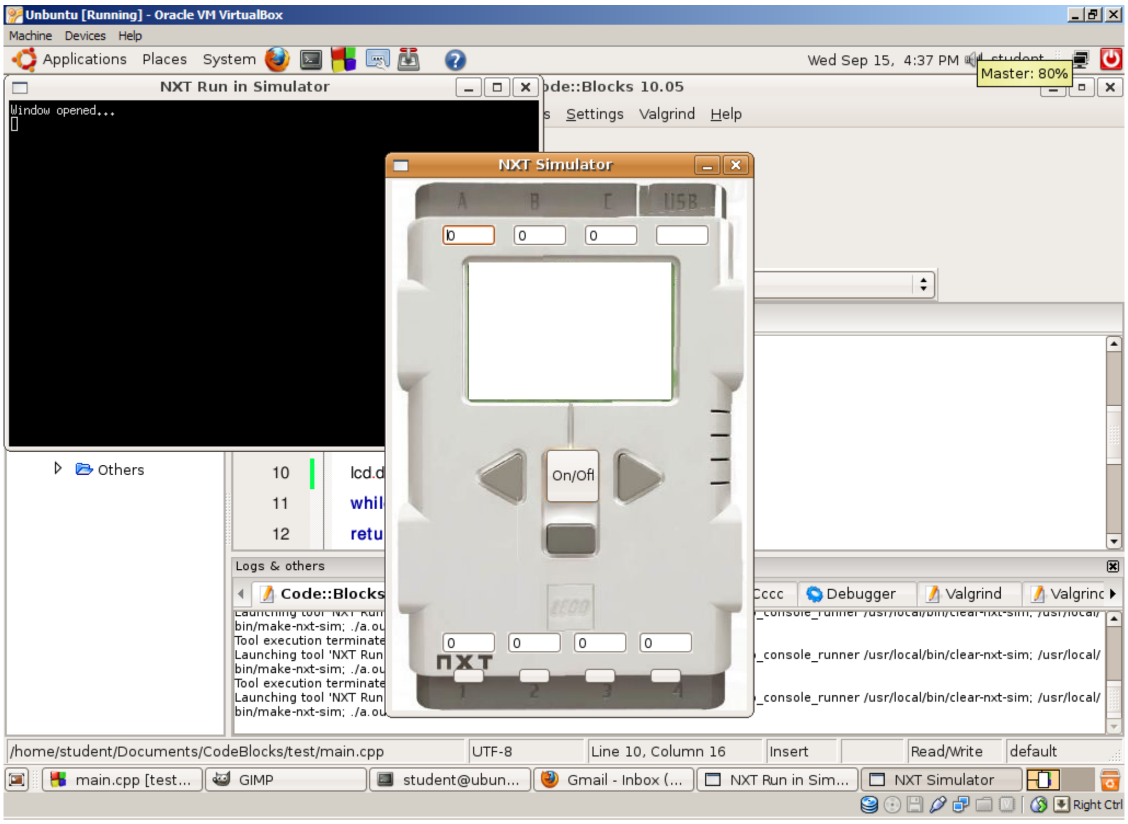
To transfer to the simulator, select Tools/Run in NXT Simulator.



```
#include "robot.h"
#include "roboTask.h"

Lcd lcd;

int main()
{
    lcd.clear();
    lcd.putf("s", "Hello World!");
    lcd.disp();
    while(1);
    return 0;
}
```

```
main.cpp (in /home/student/Documents/CodeBlocks/test/)
Tool execution terminated with status 0
Opening /home/student/Documents/CodeBlocks/test/test.cbp
done
Launching tool 'RCX Transfer Brick': xterm -T 'RCX Transfer Brick' -e /usr/bin/cb_console_runner /usr/
local/bin/rcxtransfer /home/student/Documents/CodeBlocks/test/main.cpp (in /home/student/
Documents/CodeBlocks/test)
Tool execution terminated with status 0
```

Figure 51Transfering the compiled program to the NXT simulator



```
Window opened...
```

```
Launching tool 'NXT Run                          _console_runner /usr/local/bin/clear-nxt-sim; /usr/local/
bin/make-nxt-sim; ./a.ou
Tool execution terminate
Launching tool 'NXT Run                          _console_runner /usr/local/bin/clear-nxt-sim; /usr/local/
bin/make-nxt-sim; ./a.ou
Tool execution terminate
Launching tool 'NXT Run                          _console_runner /usr/local/bin/clear-nxt-sim; /usr/local/
bin/make-nxt-sim; ./a.ou
```

Figure 52 – Simulator loaded with program in memory.

5. Run your Robot by pressing the *Run* button on the NXT Simulator. Your program should display “Hello World”. (Figure 53).

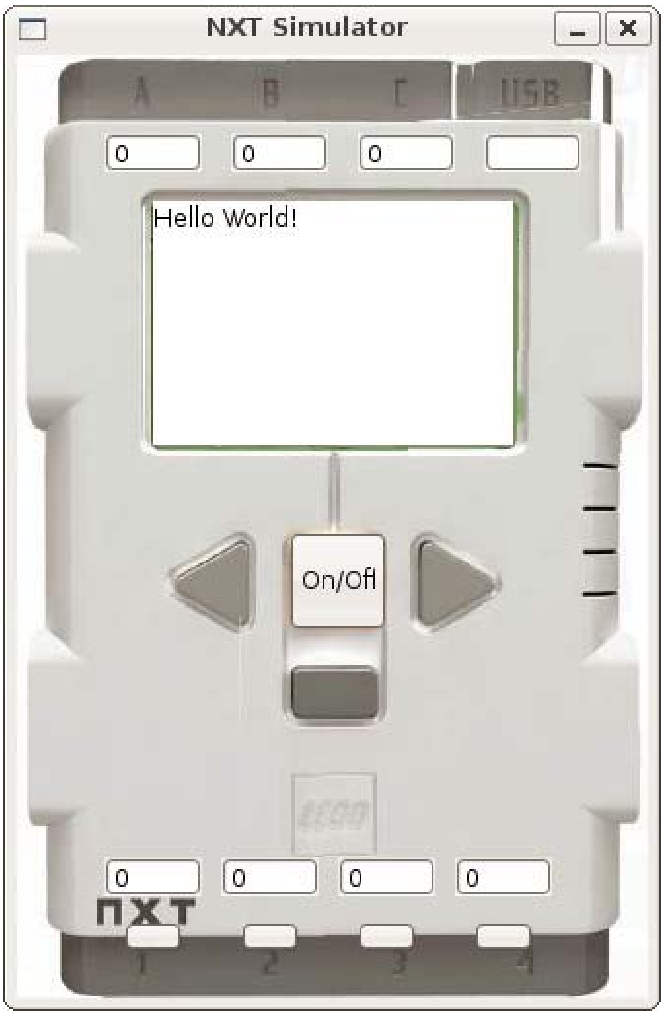


Figure 53 - BrickEMU running program

6. After a short period of time the motor values will be displayed under Port A and Port C. Note that they will switch from 50 to -50 indicating that the robot will be moving forward and then backwards.
7. Pressing the *Q*, *W*, *E*, *R* keys on your keyboard will enter a value into the sensor ports 1, 2, 3, 4 respectively. Note that the either “Huh” “Ouch” or “Bye” will be displayed in the LCD. (You may also enter values directly into the sensor port by selecting the little window above the port numbers and using your left and/or right arrow keys to raise or lower the value “sensed” by the port.)
8. Congratulations! You have just run your first nxtOSEK program on a robot simulator.
9. The next step is to learn how to down load your program into the real robot and watch it do its thing.

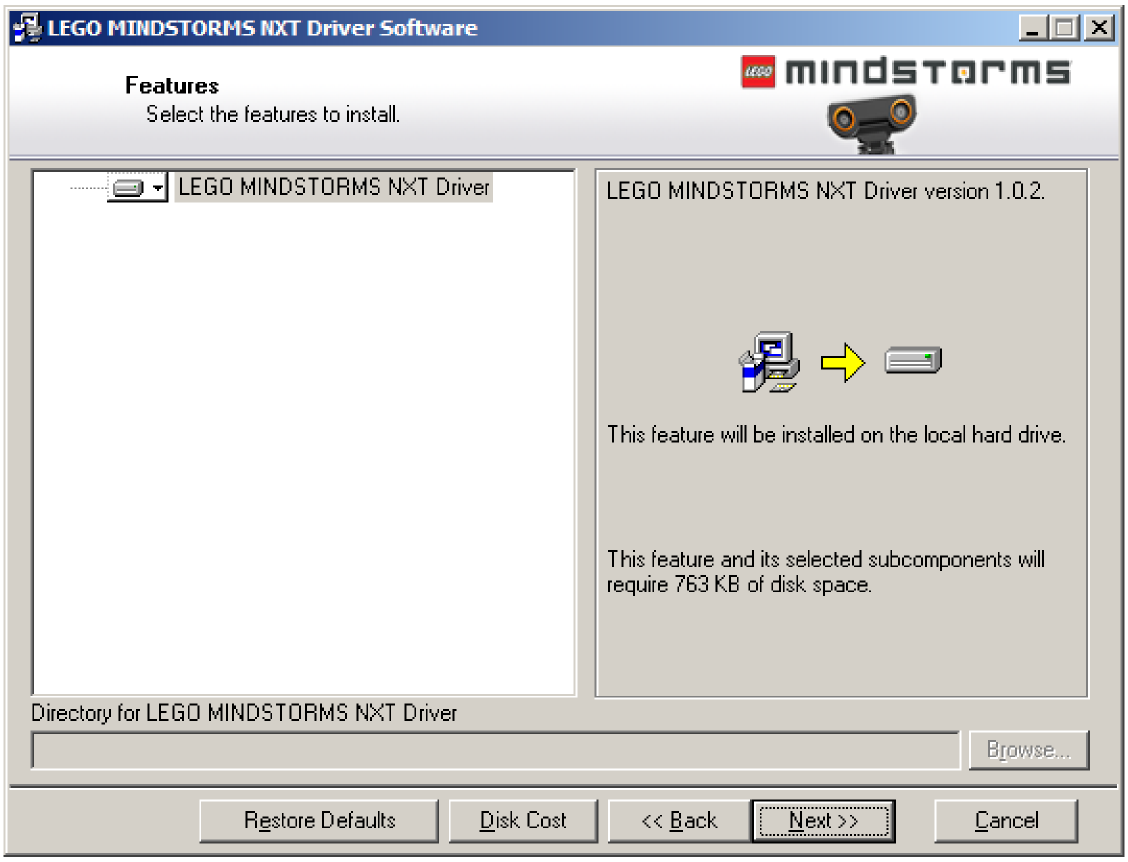
Section 11: Using the Lego NXT Brick

Before using the Lego NXT Brick for the first time it will have to be installed onto both the host and client machines. Follow the follow steps:

11.1 Installing the NXT Driver

Do not plug in the brick until after the drivers have been installed.

i. Go to the "MINDSTORMS NXT Driver v1.02" on the DVD.
ii. Click on the "*Setup*" file


LEGO MINDSTORMS NXT Driver Software

Features
Select the features to install.

LEGO MINDSTORMS NXT Driver

LEGO MINDSTORMS NXT Driver version 1.0.2.

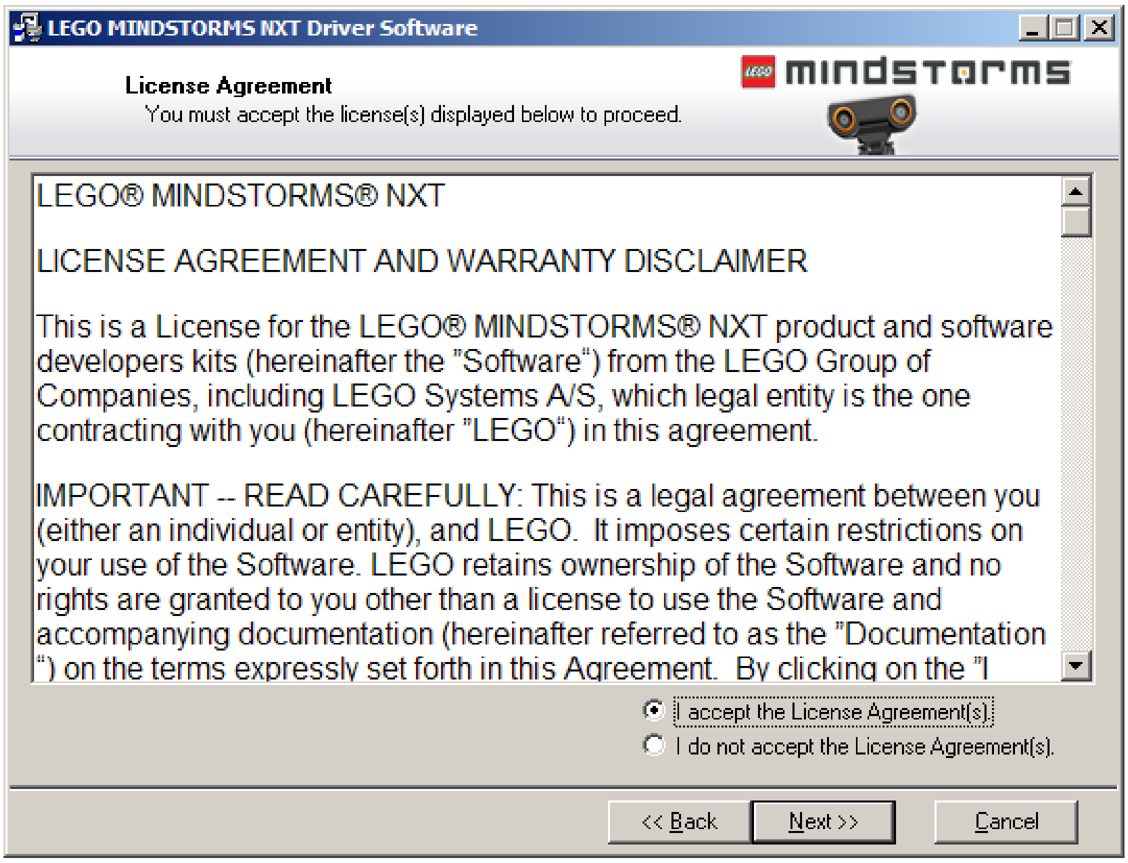
This feature will be installed on the local hard drive.

This feature and its selected subcomponents will require 763 KB of disk space.

Directory for LEGO MINDSTORMS NXT Driver

Figure 54 Installing the MindStorms NXT Driver

iii. Press **Next**



License Agreement

You must accept the license(s) displayed below to proceed.

LEGO® MINDSTORMS® NXT

LICENSE AGREEMENT AND WARRANTY DISCLAIMER

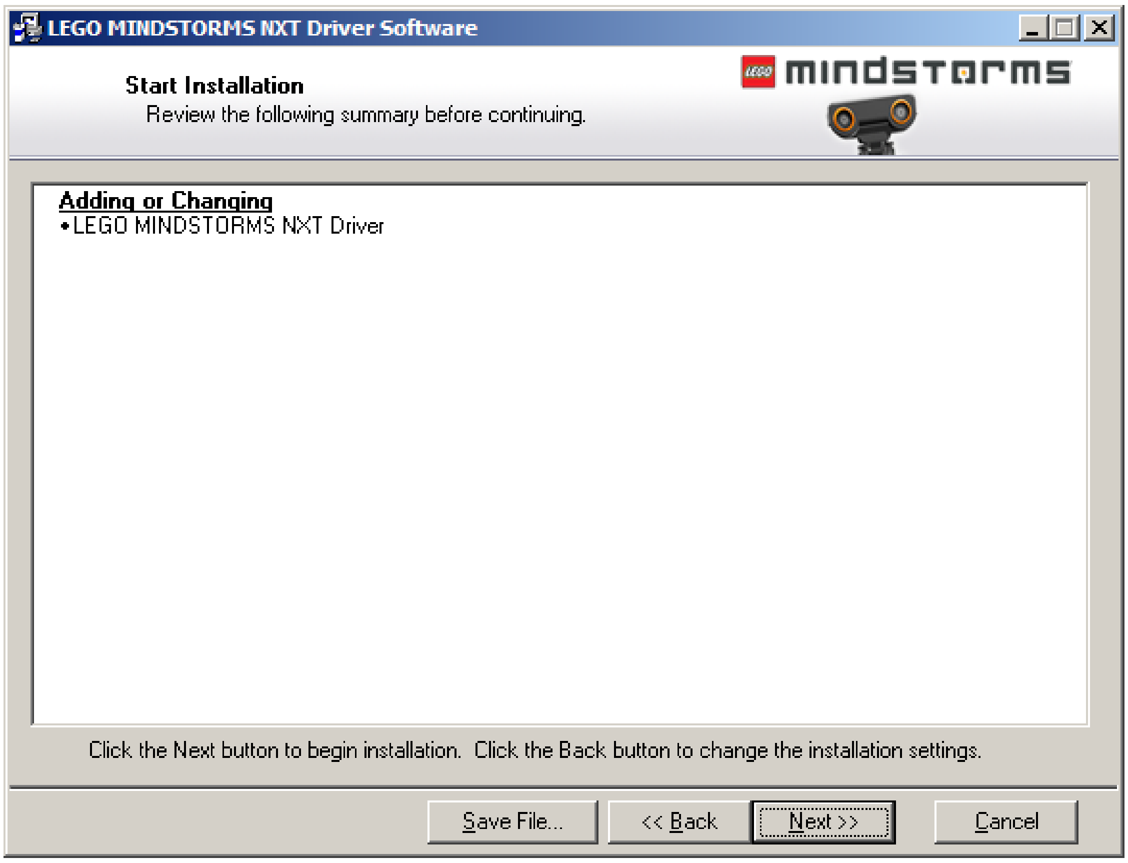
This is a License for the LEGO® MINDSTORMS® NXT product and software developers kits (hereinafter the "Software") from the LEGO Group of Companies, including LEGO Systems A/S, which legal entity is the one contracting with you (hereinafter "LEGO") in this agreement.

IMPORTANT -- READ CAREFULLY: This is a legal agreement between you (either an individual or entity), and LEGO. It imposes certain restrictions on your use of the Software. LEGO retains ownership of the Software and no rights are granted to you other than a license to use the Software and accompanying documentation (hereinafter referred to as the "Documentation") on the terms expressly set forth in this Agreement. By clicking on the "I

[✓ I accept the License Agreement(s).] I do not accept the License Agreement(s).

Figure 55

iv. Accept the agreement. Then press **Next**.



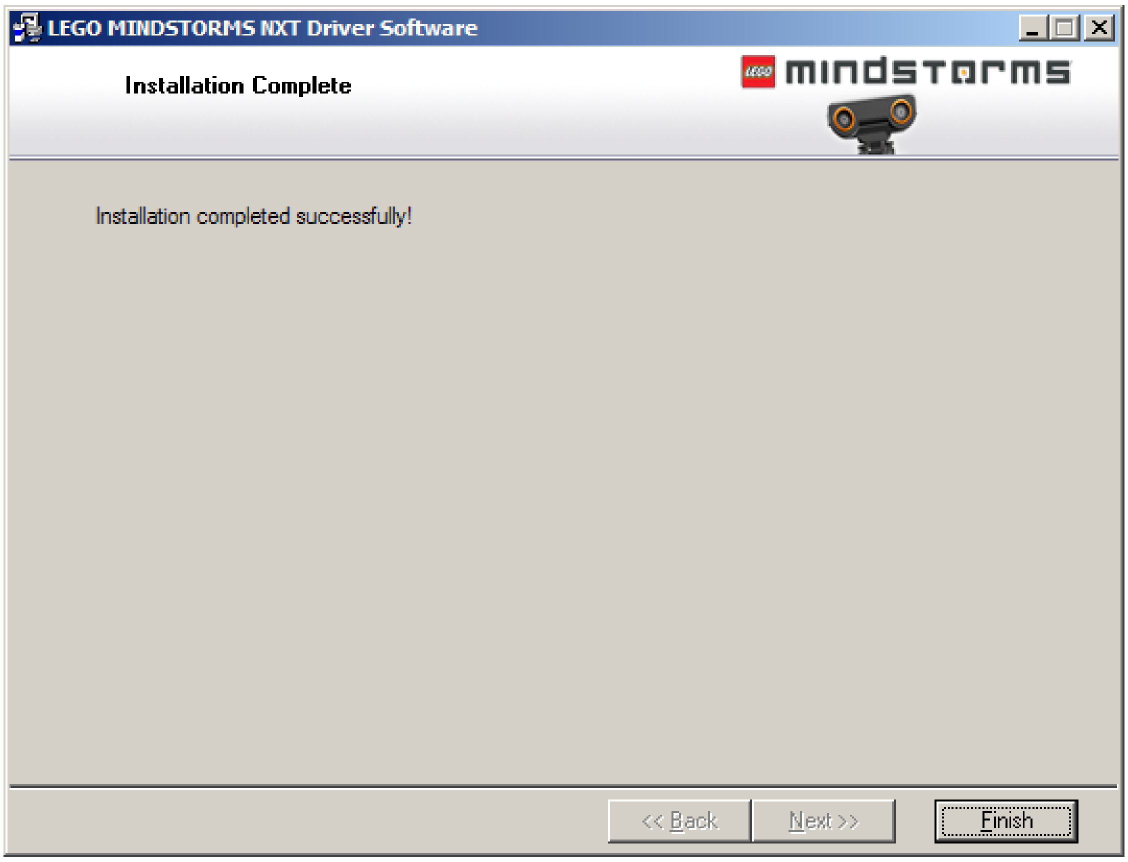
Start Installation
Review the following summary before continuing.

Adding or Changing
• LEGO MINDSTORMS NXT Driver

Click the Next button to begin installation. Click the Back button to change the installation settings.

Figure 56

v. Press **Next**.

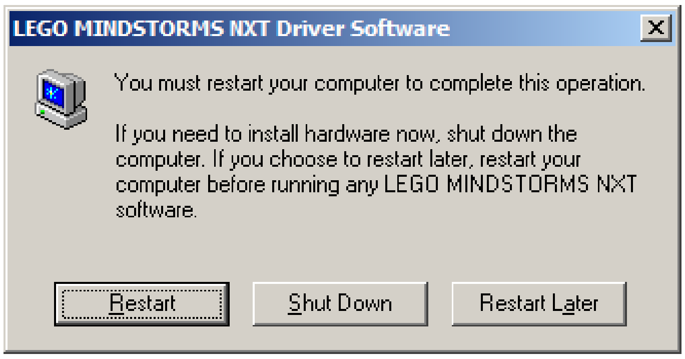

LEGO MINDSTORMS NXT Driver Software

Installation Complete

Installation completed successfully!

Figure 57

vi. When done, press **Finish**.


LEGO MINDSTORMS NXT Driver Software

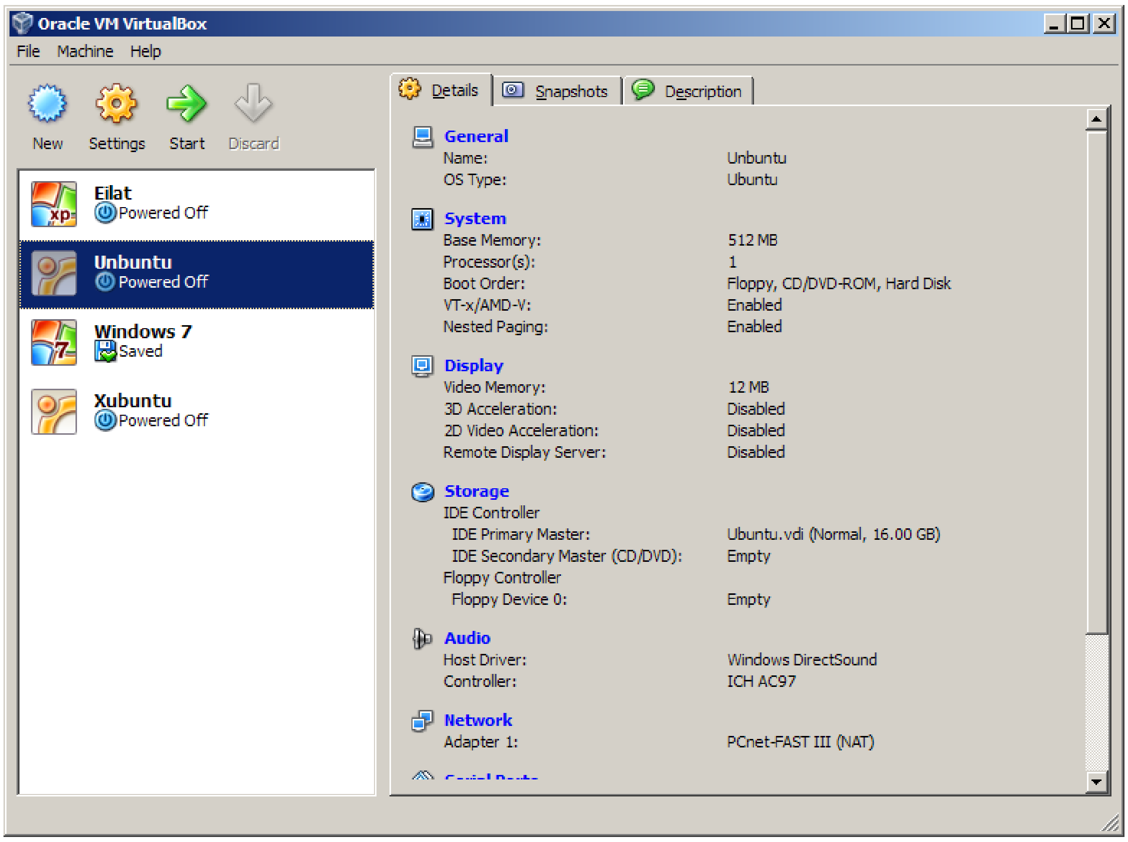
You must restart your computer to complete this operation.

If you need to install hardware now, shut down the computer. If you choose to restart later, restart your computer before running any LEGO MINDSTORMS NXT software.

Figure 58

vii. Select to **Restart** the computer.
viii. After the computer restarts, plug in the brick.
ix. Windows will report that the NXT is connected.

x. Start the *VirtualBox* console. Be sure that the *Ubuntu Virtual Machine* is powered off, and scroll the *Details* window until the *USB* section is visible, as in the screenshot below:



General
Name: Unbuntu
OS Type: Ubuntu

System
Base Memory: 512 MB
Processor(s): 1
Boot Order: Floppy, CD/DVD-ROM, Hard Disk
VT-x/AMD-V: Enabled
Nested Paging: Enabled

Display
Video Memory: 12 MB
3D Acceleration: Disabled
2D Video Acceleration: Disabled
Remote Display Server: Disabled

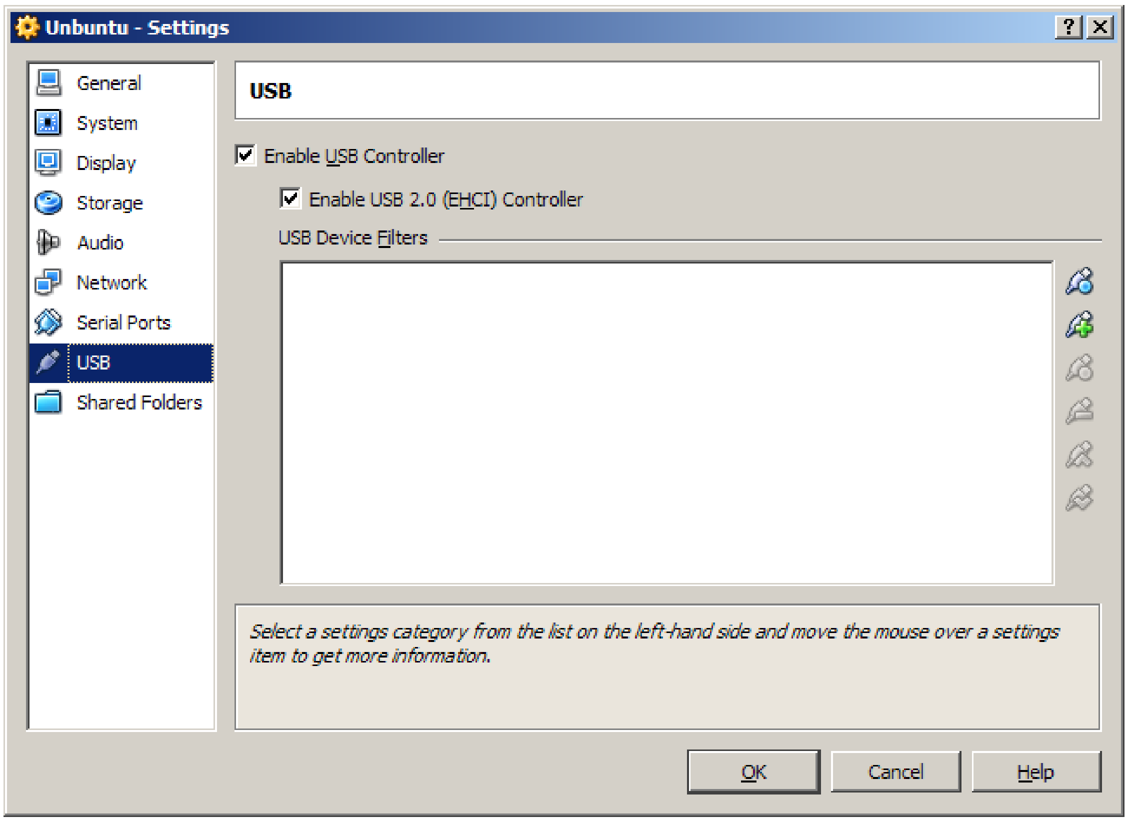
Storage
IDE Controller
IDE Primary Master: Ubuntu.vdi (Normal, 16.00 GB)
IDE Secondary Master (CD/DVD): Empty
Floppy Controller
Floppy Device 0: Empty

Audio
Host Driver: Windows DirectSound
Controller: ICH AC97

Network
Adapter 1: PCnet-FAST III (NAT)

Figure 59

xi. Double-click on the *USB section* (Windows - Figure 60) or the *Ports section* (Mac OS X - Figure 61) and select the 2nd icon on the right to add USB Device Filters to the Virtual machine


Unbuntu - Settings

USB

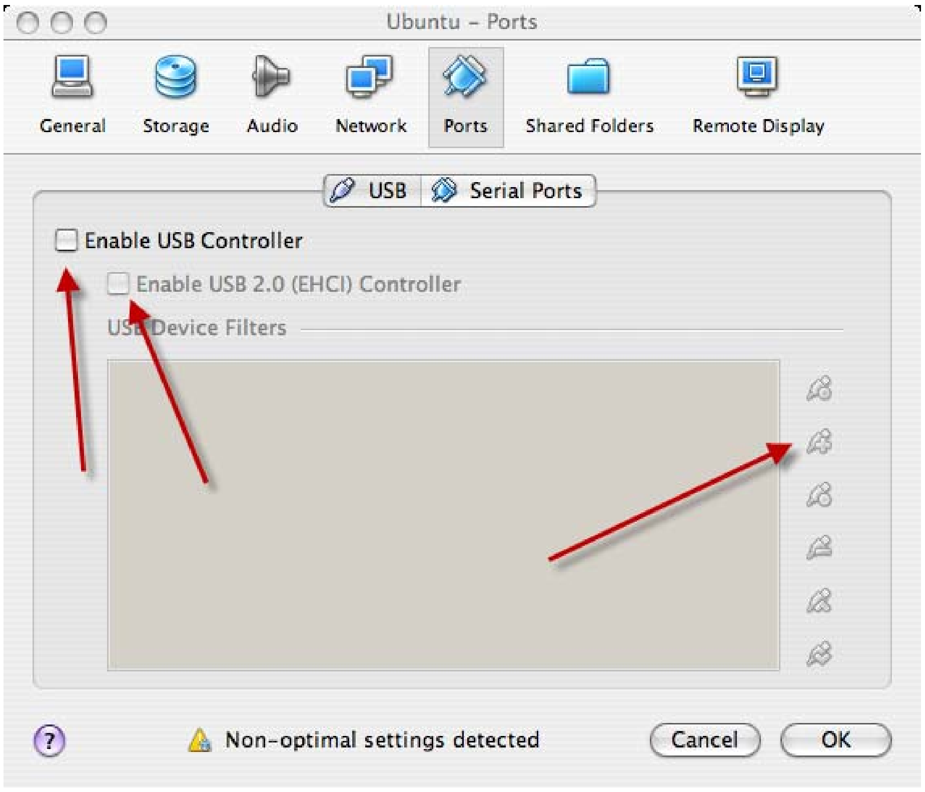
☑ Enable USB Controller

☑ Enable USB 2.0 (EHCI) Controller

USB Device Filters

Select a settings category from the list on the left-hand side and move the mouse over a settings item to get more information.

Figure 60 – VirtualBox (Windows) USB Setup



Enable USB Controller

Enable USB 2.0 (EHCI) Controller

USB Device Filters

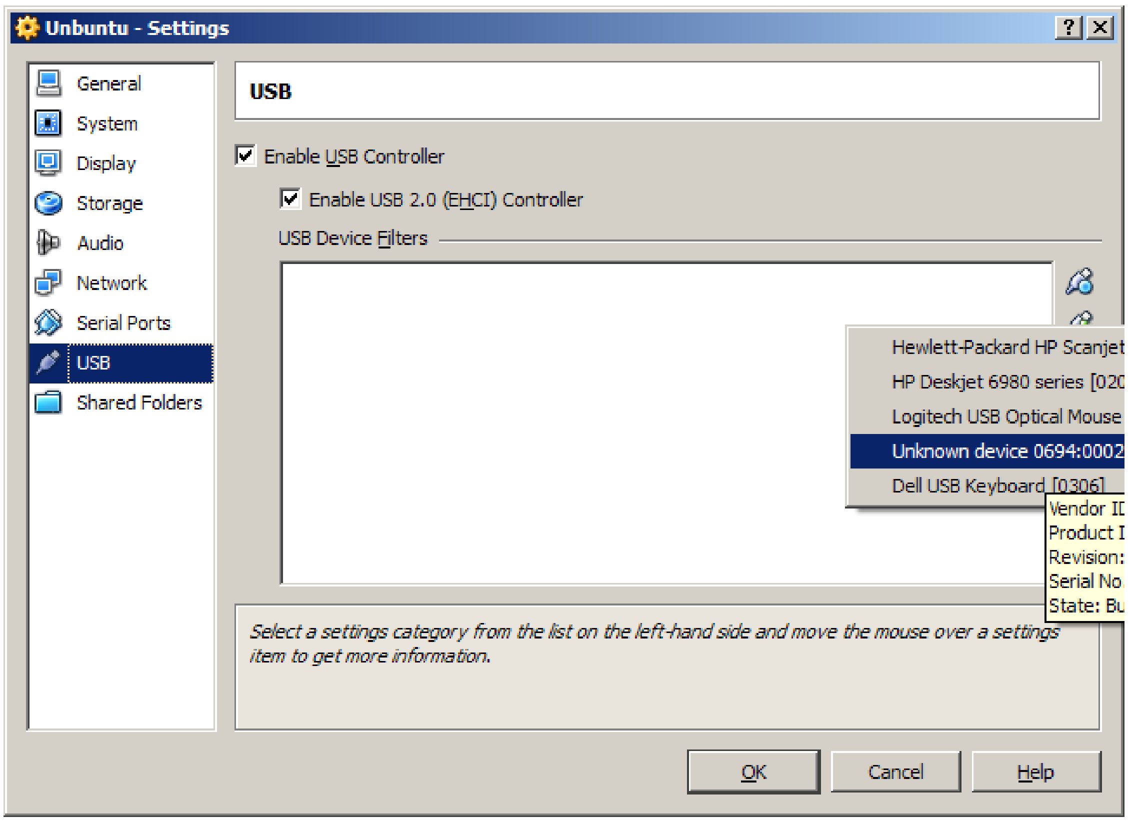
Non-optimal settings detected

Figure 61 – VirtualBox (Mac OS X) USB Setup

xii. Be sure to enable the USB Controller by checking both boxes.

xiii. Choose the *Unknown Device* from the list that appears. You may also wish to add your USB printer (if any) so that it becomes available to the Ubuntu Virtual Machine.

Note: Items selected will become unavailable to the host as long as the client is active. If you print anything from the host, it will be held in the printer queue until you shut down the virtual machine.


Unbuntu - Settings

General
System
Display
Storage
Audio
Network
Serial Ports
USB
Shared Folders

USB

☑ Enable USB Controller

☑ Enable USB 2.0 (EHCI) Controller

USB Device Filters

Hewlett-Packard HP Scanjet
HP Deskjet 6980 series [020
Logitech USB Optical Mouse
Unknown device 0694:0002
Dell USB Keyboard [0306]

Vendor ID
Product I
Revision:
Serial No
State: Bu

Select a settings category from the list on the left-hand side and move the mouse over a settings item to get more information.

OK   Cancel   Help

Figure 62

xiv. Choose **OK** to return to the VirtualBox console.

xv. Start the Ubuntu Virtual Machine, and *right-click* on the USB icon as shown below. In the resultant list of devices place a check mark by clicking to the left of the entry for the *Unknown Device*.

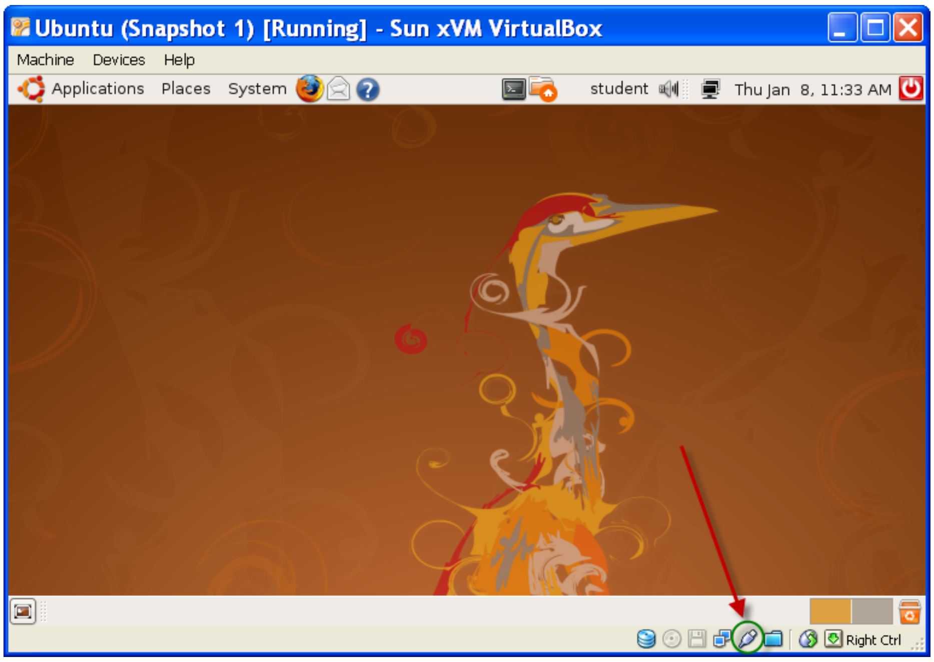
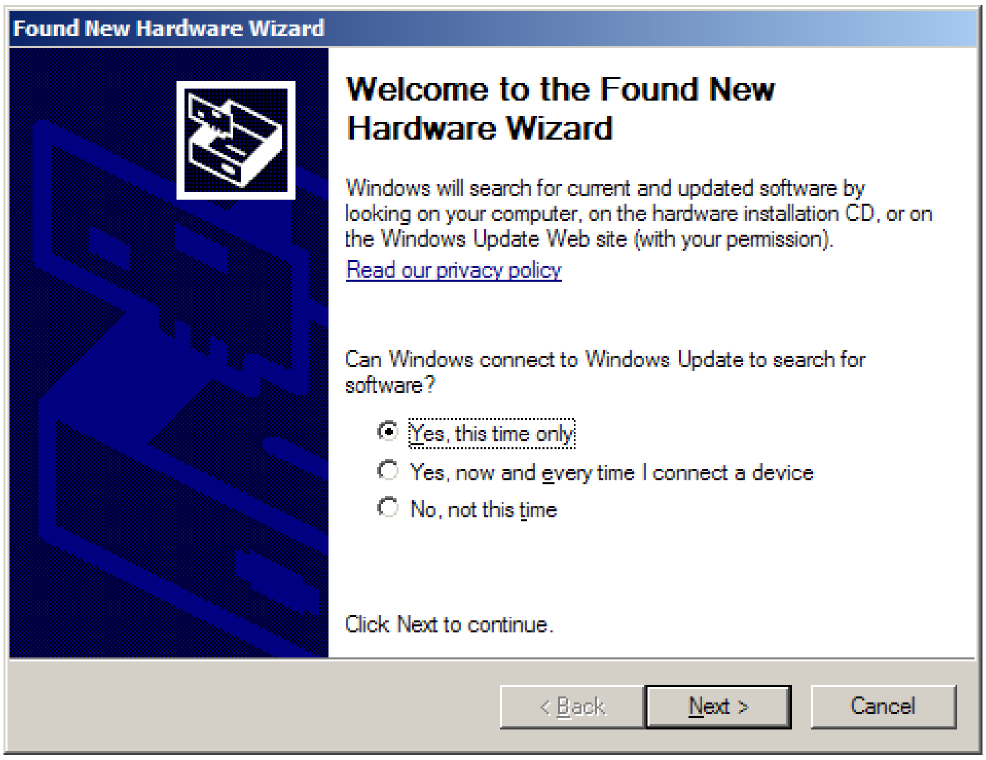


Figure 63

You may be asked to install VirtualBox USB drivers:



Welcome to the Found New Hardware Wizard

Windows will search for current and updated software by looking on your computer, on the hardware installation CD, or on the Windows Update Web site (with your permission).

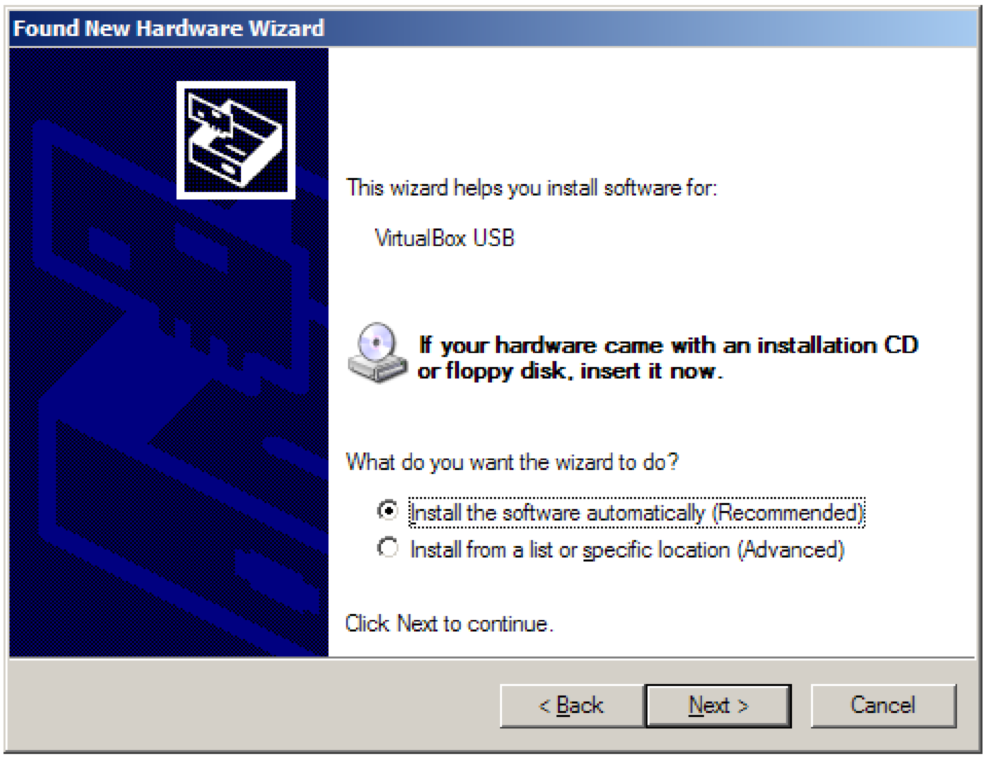
Read our privacy policy

Can Windows connect to Windows Update to search for software?

[✓ Yes, this time only]
Yes, now and every time I connect a device
No, not this time

Click Next to continue.

*Select **Yes**.*



Found New Hardware Wizard

This wizard helps you install software for:

VirtualBox USB

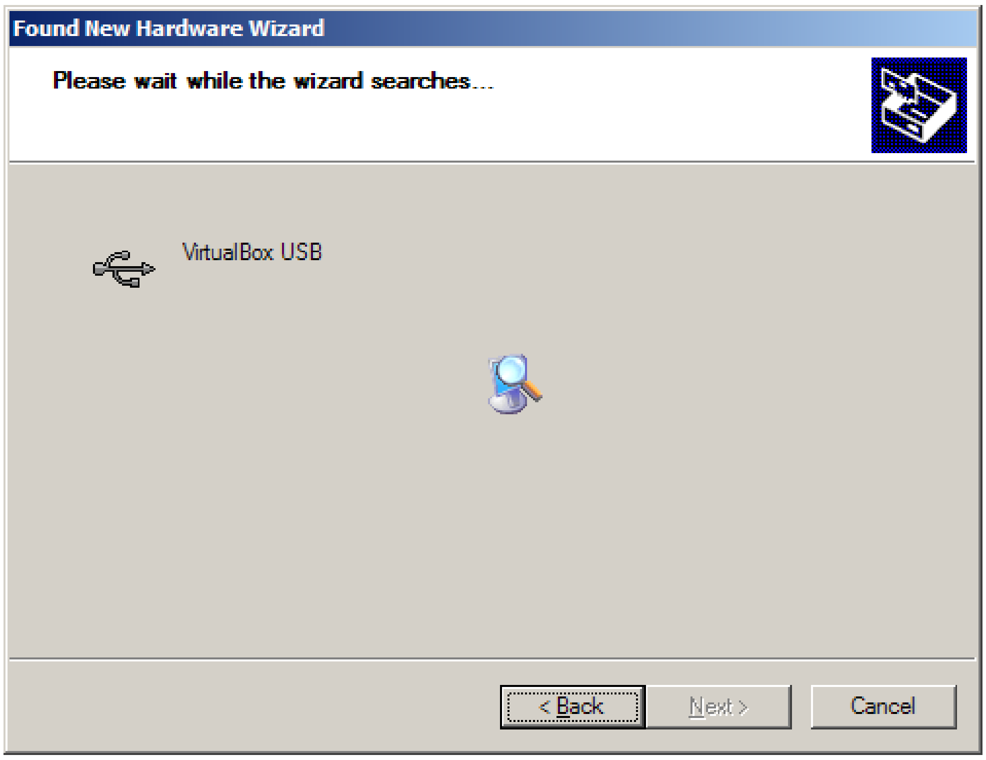
If your hardware came with an installation CD or floppy disk, insert it now.

What do you want the wizard to do?

[✓ Install the software automatically (Recommended)]
Install from a list or specific location (Advanced)

Click Next to continue.

Select ***Install the software automatically (Recommended)***.

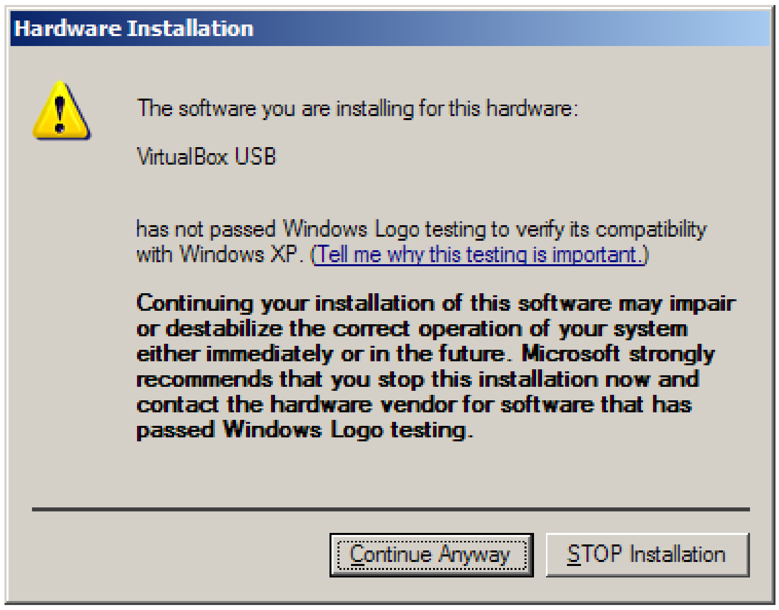


Found New Hardware Wizard

Please wait while the wizard searches...

VirtualBox USB

The VirtualBox USB software will install.



Hardware Installation

The software you are installing for this hardware:

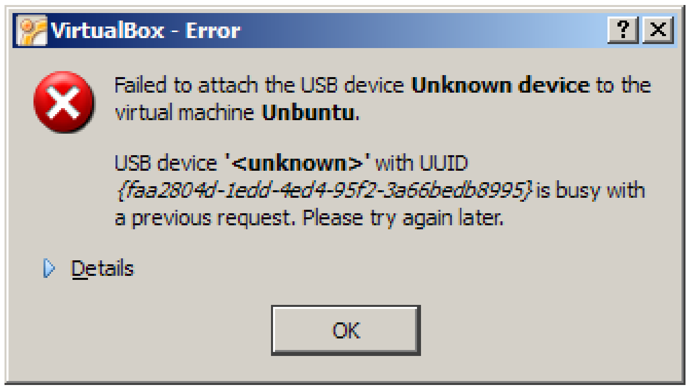
VirtualBox USB

has not passed Windows Logo testing to verify its compatibility with Windows XP. (Tell me why this testing is important.)

Continuing your installation of this software may impair or destabilize the correct operation of your system either immediately or in the future. Microsoft strongly recommends that you stop this installation now and contact the hardware vendor for software that has passed Windows Logo testing.

Select to Continue Anyway.

NOTE: When using the NXT Brick, you may get an error such as the following:



VirtualBox - Error

Failed to attach the USB device **Unknown device** to the virtual machine **Unbuntu**.

USB device '**<unknown>**' with UUID *{faa2804d-1edd-4ed4-95f2-3a66bedb8995}* is busy with a previous request. Please try again later.

Unplug the USB cable on the computer and plug the cable into a different slot. On the list of USB devices in Figure 59 you will see duplicate listings for Unknown Device (select the top one, if that doesn't work, try the bottom one). If neither works, shut down the virtual machine and start it up again. If there is still a problem restart your computer and try again. You may wish to try steps v through viii for this new "unknown device".

xvi. If you get an error message as shown above, *shut down* the virtual machine and *reboot* the host computer. Repeat the previous step (xiii). This time the brick will be captured by the virtual machine.

xvii. You now may proceed to load the firmware into the Lego NXT Brick. (See next section).

11.2 How to install the firmware

To update the firmware, press the reset button (at the back of the NXT, upper left corner in a hole beneath the USB connector) for more than 5 seconds while the Lego NXT is turned on. Use a bent paper clip to access the button (**Error! Reference source not found.**).

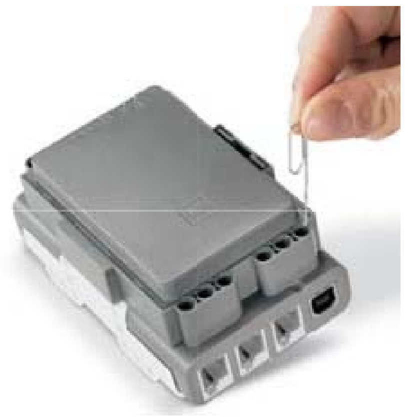
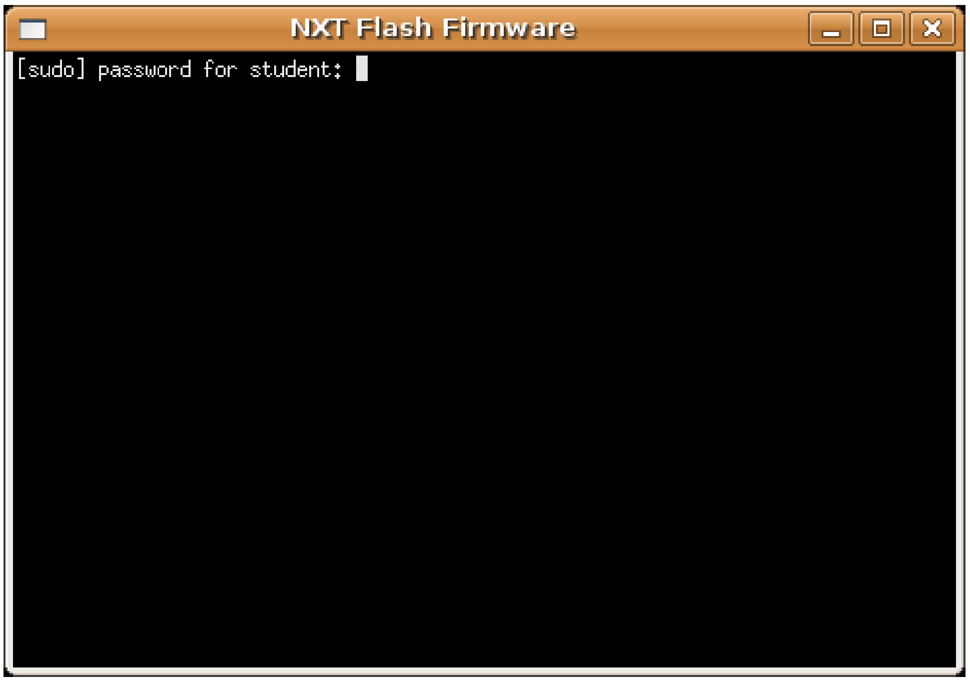


Figure 64 Entering firmware update mode

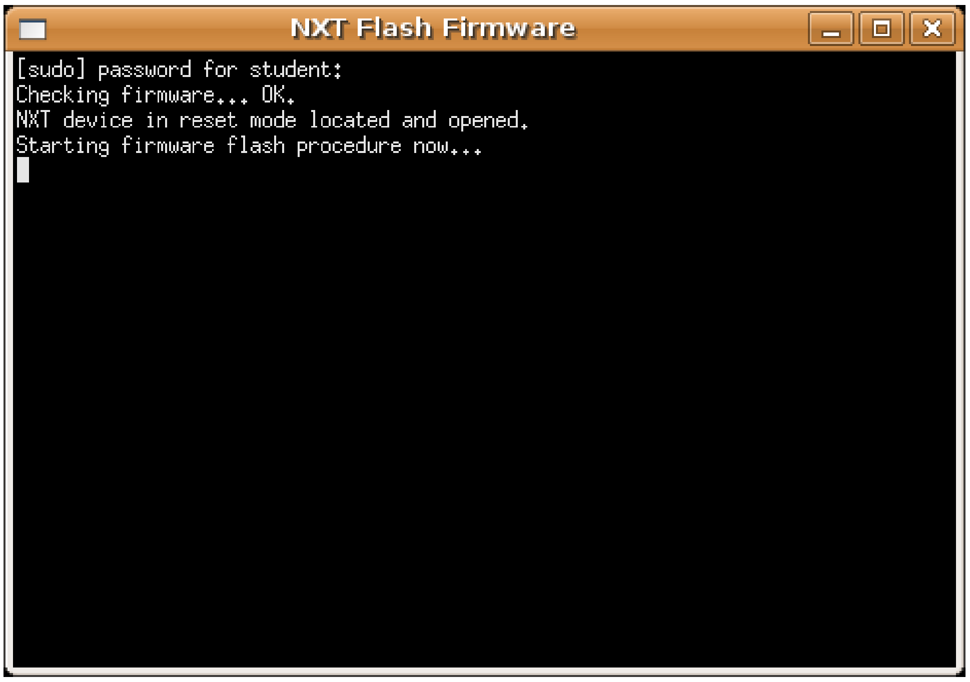
The NXT will make a low tick sound when it is in firmware update mode. In the USB Taskbar, Windows will show “NXT Running in Update mode” as the title of the USB Device instead of the Robot name. In Virtual Box, this will have a different ID next to “Unknown Hardware” in the list of USB devices. **Select the top one on the list** to enable USB Update mode within Virtual Box. Start the Code::Blocks application and choose *Tools/NXT Flash Firmware* from the Code::Blocks menu and a window will come up for flashing the firmware (Figure 65).


NXT Flash Firmware

```
[sudo] password for student:
```

Figure 65 - Loading the John Hansen Enhanced Firmware on to the RCX Brick

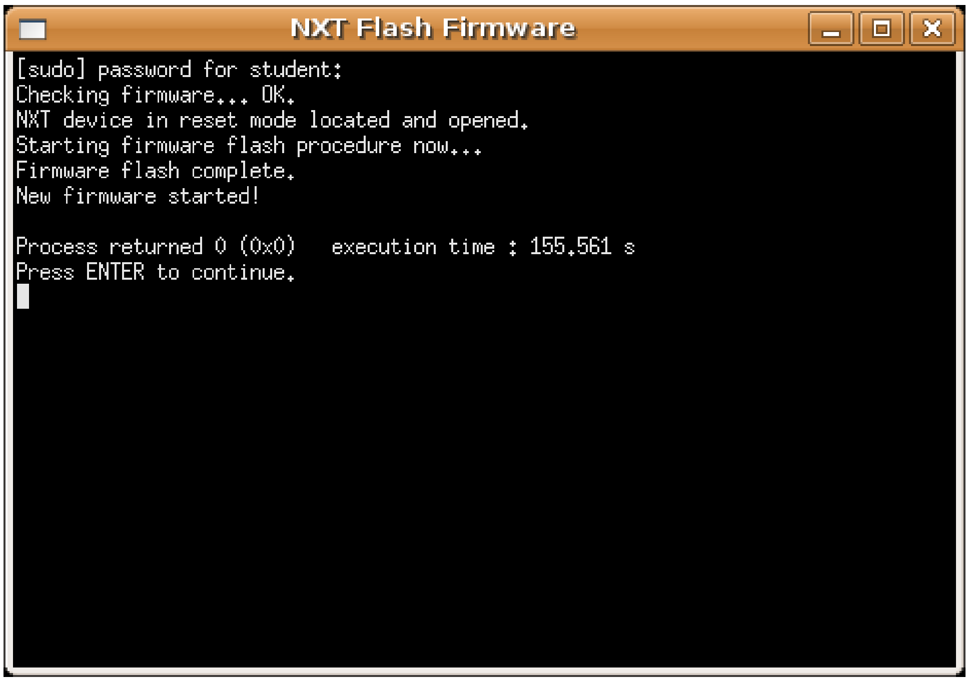
Enter the password (‘*student*’) and wait for the process to complete.


NXT Flash Firmware

```
[sudo] password for student:
Checking firmware... OK.
NXT device in reset mode located and opened.
Starting firmware flash procedure now...
```

Figure 66

It will take about 3 minutes until the firmware flash procedure is complete.



```
[sudo] password for student:
Checking firmware... OK.
NXT device in reset mode located and opened.
Starting firmware flash procedure now...
Firmware flash complete.
New firmware started!

Process returned 0 (0x0)   execution time : 155.561 s
Press ENTER to continue.
```

Figure 67

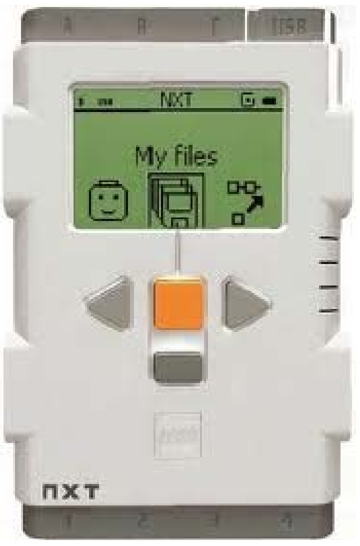
Press **Enter** when it is finished.

It is recommended that you shutdown the Ubuntu System and close VirtualBox before proceeding to use the NXT.

11.3 How to transfer the compiled program to the LEGO Brick

Attach the USB cable. Turn the brick on. Start the Code::Blocks application and choose *Tools/NXT Transfer Brick* from the Code::Blocks menu. A message window opens up, showing the progress of the transfer. When the transfer completes, press '*Enter*'.

11.4 Running your program



- The orange square is the select button on the NXT. The arrows on either side of it are used for making choices.
- Turn the NXT Brick on, and press orange button to select "*My Files*". Then select "*Software Files*".
- Select the program to run; such as "main".
- Select "*Run*" from the menu.
- The brick's battery status is shown on the program screen.
- To run the program, press the right arrow button. If you selected the wrong program, pressing the bottom button will shut off the brick.

To end the program, press the bottom button. Remember that once the NXT is shut off (or unplugged from the USB cable), you will need to recapture it in VirtualBox.

Section 12: Advanced Topics

12.1 Setting up the printer

There are six steps for setting up a printer as follows:

i. First make sure that your USB printer is installed correctly for your host operating system.
ii. Make sure your printer is powered ON.
iii. Add the USB filter for your printer as described in Section 11.1
iv. Start your Ubuntu Virtual machine
v. Choose *System/Administration/Printing* from the Ubuntu menu.
vi. Choose the *New Printer* icon and follow the wizard to install your printer.

Note: Items selected in the VirtualBox USB settings section (Figure 62) will become unavailable to the host as long as the client is active. If you print anything from the host, it will be held in the printer queue until you shut down the virtual machine and only then will it print.

12.2 How to Setup Shared Folders

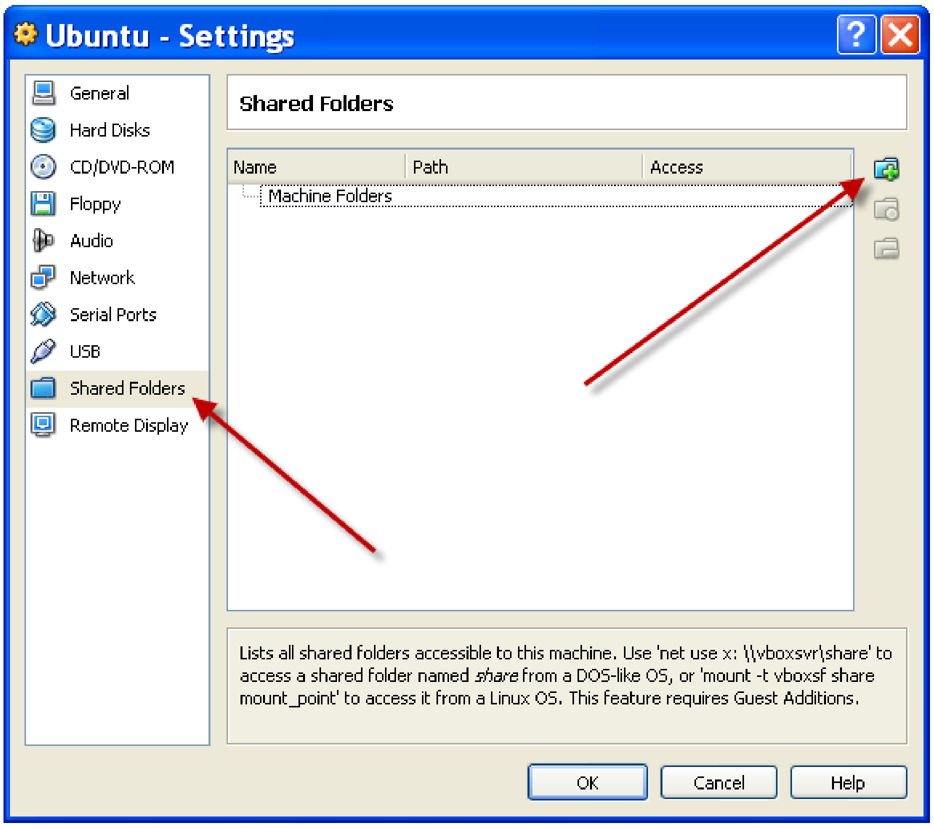
It is often useful to be able to transfer information between the host computer and the virtual Ubuntu machine. For simple transfer of text, *cut and paste*, will function just as it does on the host. Simply copy the text you wish to copy in the source window and paste it at the desired location in the destination window.

However, it is also useful to be able to exchange files between the host and the client. This is done by what is known as "*Shared Folders*", that is a folder in the host and a folder in the client that mirror each other's contents.

The procedure is similar for both a Windows and Mac hosts. In the following steps we will use a Windows XP screenshots to illustrate the process of setting up a shared folder.

a. Create a folder on the host. For this example we will create a folder called *My Documents\VirtualBoxShared\Ubuntu*.

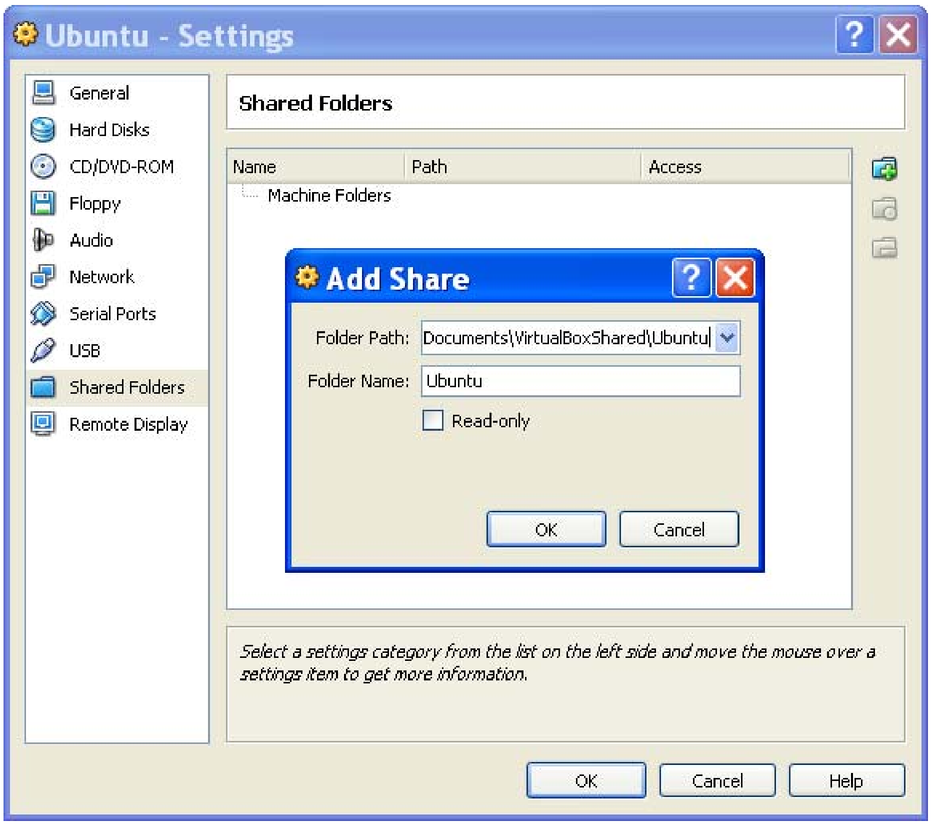
b. From the VirtualBox's menu select *Machine/Settings* and from the list of icons on the left select *Shared Folders* (Figure 68). Press the button with the + symbol to add a new shared folder in the list. You will have to specify a folder name for each folder you add. Make sure you memorize that name because you will need it very soon.



Lists all shared folders accessible to this machine. Use 'net use x: \\vboxsvr\share' to access a shared folder named *share* from a DOS-like OS, or 'mount -t vboxsf share mount_point' to access it from a Linux OS. This feature requires Guest Additions.

Figure 68 - Adding Shared Folders

c. Using the pull down menu, select the location of the folder you created in step a.

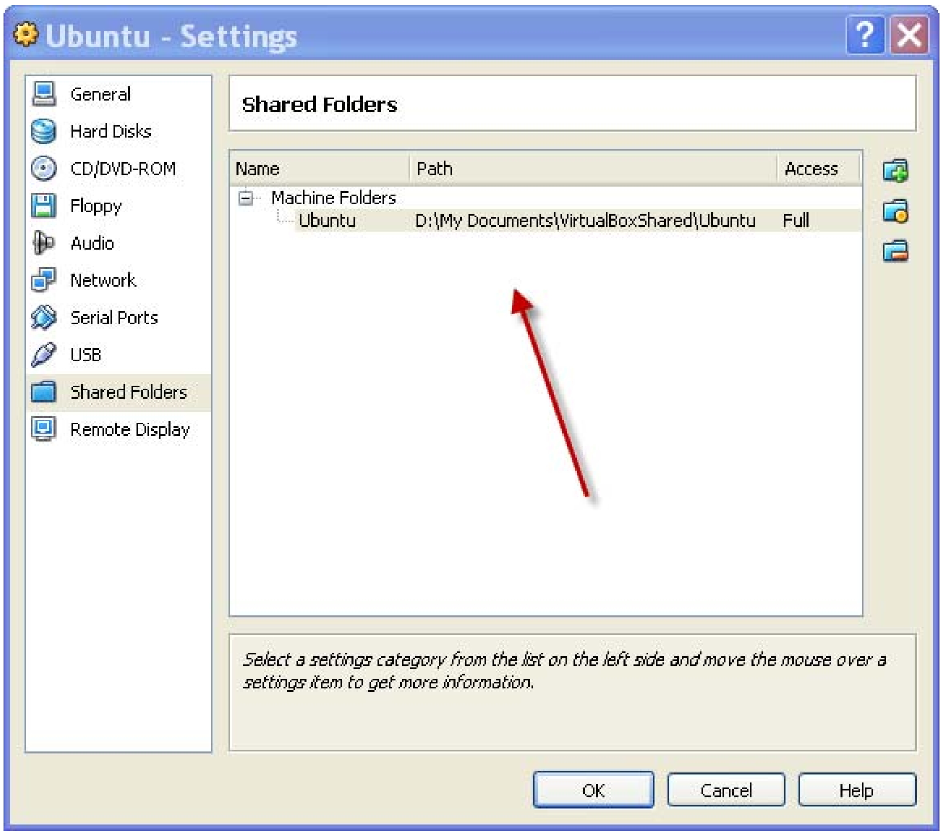


Folder Path: Documents\VirtualBoxShared\Ubuntu
Folder Name: Ubuntu

Select a settings category from the list on the left side and move the mouse over a settings item to get more information.

Figure 69 - Specifying the Shared Folder

d. After selecting OK, the window should appear as in Figure 70



Shared Folders

Name	Path	Access
Machine Folders		
Ubuntu	D:\My Documents\VirtualBoxShared\Ubuntu	Full

Select a settings category from the list on the left side and move the mouse over a settings item to get more information.

Figure 70 - Shared Folder Selected

e. Choose OK and then start the Ubuntu virtual machine.

f. Choose Places/Home Folder from the Ubuntu menu and create the folder in your Home directory which you wish to share. (Note: *the File Browser* in Ubuntu is similar to *Windows Explorer*.) In our example I have called the folder *WindowsShare* as shown in Figure 71

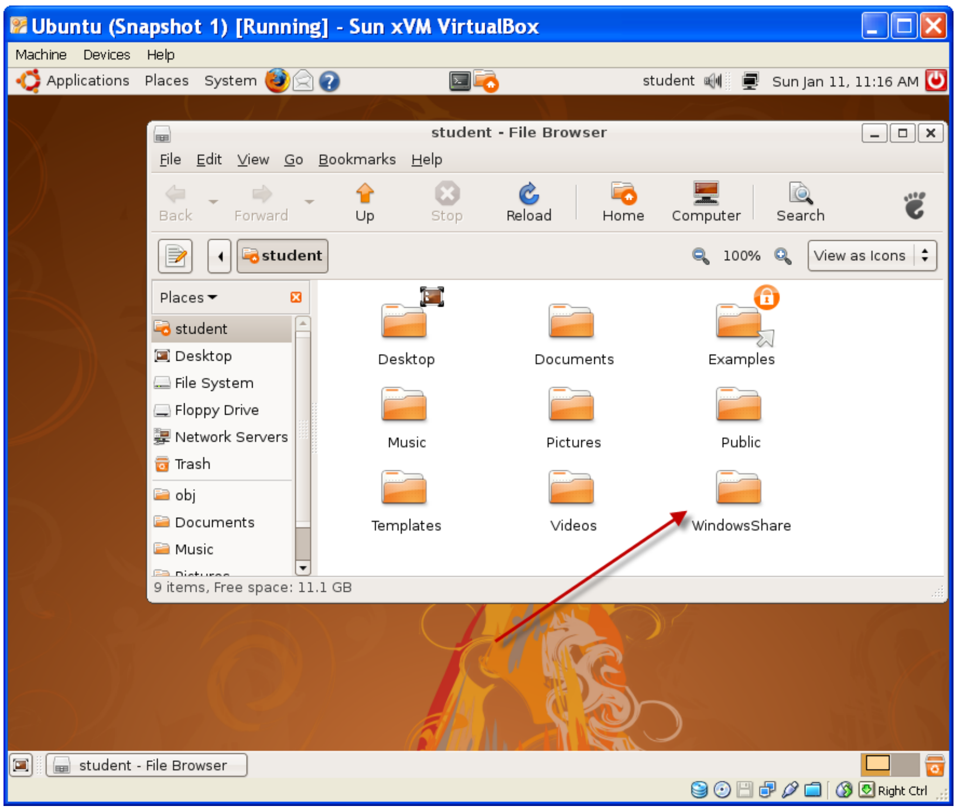
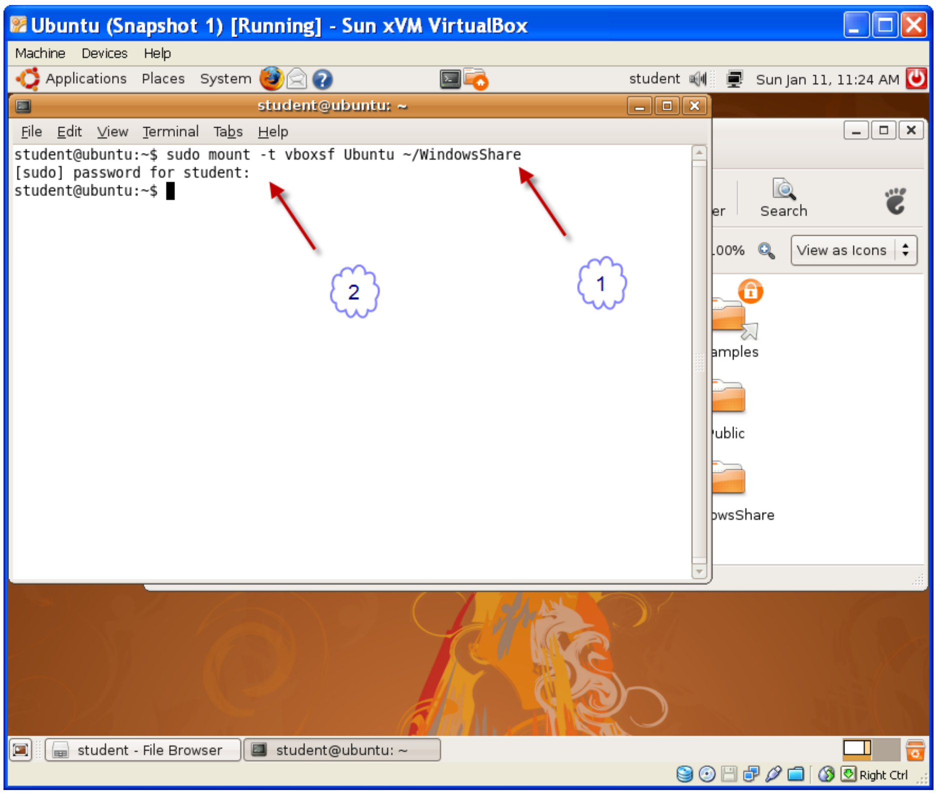


Figure 71 - Ubuntu File Browser

g. Choose *Applications/Accessories/Terminal* from the Ubuntu menu and (assuming you names the folders as I did above) type the following line at the prompt: (Note: Linux is case sensitive, so you must use consistent capitalization.)

```
sudo mount -t vboxsf Ubuntu ~/WindowsShare
```

h. Press *Enter* and enter your password at the prompt (*student*). Your screen should look like Figure 72



```
student@ubuntu:~$ sudo mount -t vboxsf Ubuntu ~/WindowsShare
[sudo] password for student:
student@ubuntu:~$ 
```

Figure 72 - Mounting the Shared Folder in Ubuntu

i. You may now close the Ubuntu *Terminal* window. The *WindowsShare* folder in Ubuntu and the *C:\My Documents\VirtualBoxShared\Ubuntu* folder in Windows will now mirror each other.

Note: You will have to remount the shared folder each time you reboot the Ubuntu virtual machine. Users familiar with Linux may use the /etc/init.d/rc.local script to execute these commands on startup to have the shared folders automatically mounted every time you start your Ubuntu VirtualBox.

Appendix I – References

a. Programming nxtOSEK
    i. http://lejos-osek.sourceforge.net/
    ii. http://lejos-osek.sourceforge.net/api.htm
b. More on Ubuntu Linux
    i. http://www.techotopia.com/index.php/Ubuntu_Linux_Essentials
    ii. http://www.techotopia.com/index.php/Ubuntu_Desktop_Essentials
    iii. https://help.ubuntu.com/ubuntu/desktopguide/C/index.html
    iv. http://ubuntuforums.org/index.php
    v. http://www.ubuntu.com/
    vi. http://ubuntutip.googlepages.com/
c. More on VirtualBox
    i. http://www.virtualbox.org/
d. More on CodeBlocks
    i. http://www.sci.brooklyn.cuny.edu/~goetz/codeblocks/
    ii. http://www.sci.brooklyn.cuny.edu/~goetz/codeblocks/codeblocks-instructions.pdf
    iii. http://www.codeblocks.org/

Appendix II   Installing VirtualBox Guest Additions

The “*VirtualBox Guest Additions*” are a set of drivers and utilities that are shipped as a subset of the VirtualBox for the purpose of being installed *inside* a *guest* to improve its performance and cooperation with the host.[8] It provides such enhancements as a special video driver that allows for better performance and features such as dynamically adjusting the guest resolution when the virtual machine window is resized.[9] Other features include support for mouse integration, communication, and shared folders.

When we originally installed the system, the guest additions for Ubuntu were automatically installed. If you update the Ubuntu client, it may be necessary to reinstall these guest additions. The following is the procedure: (Note we use screenshots of a Mac OS X host to illustrate the procedure. However, since the installation of the Guest Additions takes place completely within the client, the procedure will be the same on Windows based hosts.)

[8] http://www.virtualbox.org/wiki/VirtualBox_PUEL
[9] http://en.wikipedia.org/wiki/VirtualBox

1. Go to the *Devices* Menu from VirtualBox and select *Install Guest Additions* (Figure 73).

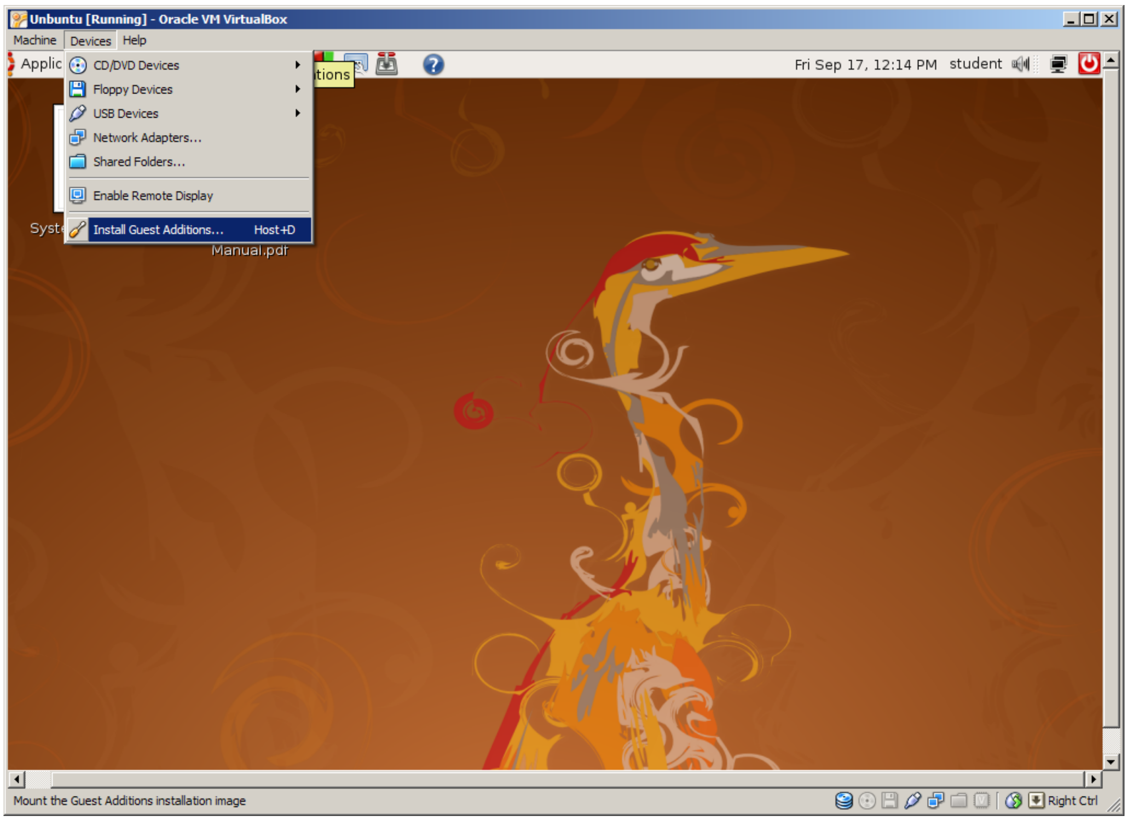
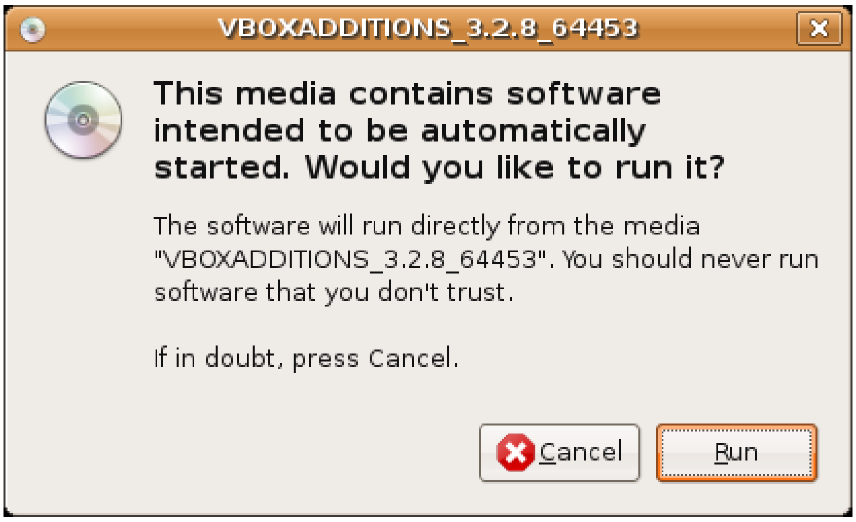


Figure 73 - Installation of Guest Additions

2. Select **Run**.


VBOXADDITIONS_3.2.8_64453

This media contains software intended to be automatically started. Would you like to run it?

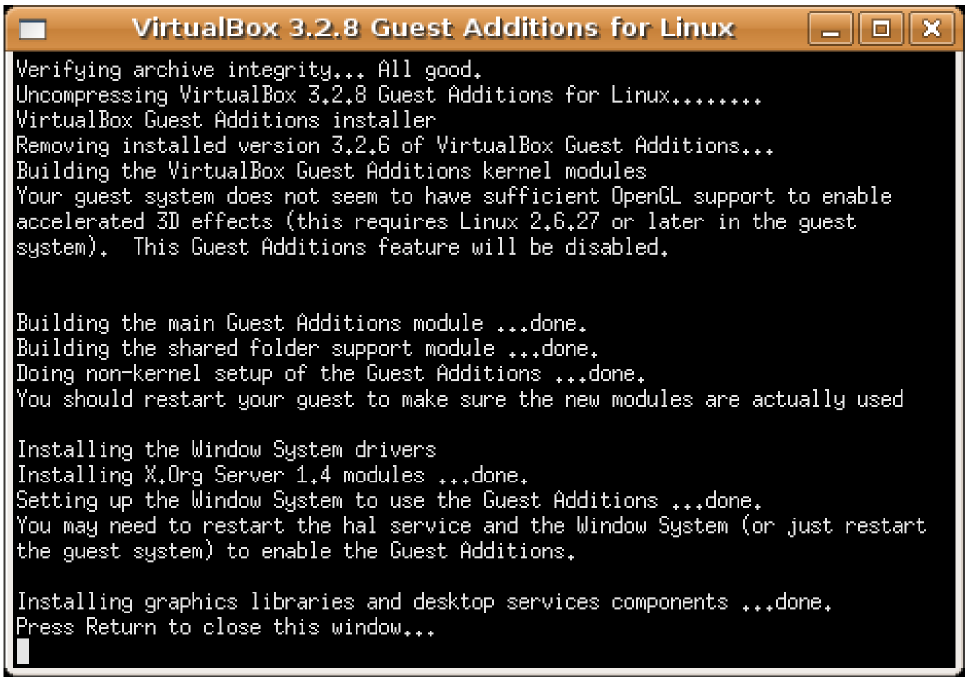
The software will run directly from the media "VBOXADDITIONS_3.2.8_64453". You should never run software that you don't trust.

If in doubt, press Cancel.

Figure 74

3. The system may ask you for your password. If so, enter in your password (***student***), and press OK.

4. When the Additions have finished installing press **Return**.



```
Verifying archive integrity... All good.
Uncompressing VirtualBox 3.2.8 Guest Additions for Linux........
VirtualBox Guest Additions installer
Removing installed version 3.2.6 of VirtualBox Guest Additions...
Building the VirtualBox Guest Additions kernel modules
Your guest system does not seem to have sufficient OpenGL support to enable
accelerated 3D effects (this requires Linux 2.6.27 or later in the guest
system).  This Guest Additions feature will be disabled.


Building the main Guest Additions module ...done.
Building the shared folder support module ...done.
Doing non-kernel setup of the Guest Additions ...done.
You should restart your guest to make sure the new modules are actually used

Installing the Window System drivers
Installing X.Org Server 1.4 modules ...done.
Setting up the Window System to use the Guest Additions ...done.
You may need to restart the hal service and the Window System (or just restart
the guest system) to enable the Guest Additions.

Installing graphics libraries and desktop services components ...done.
Press Return to close this window...
```

Figure 75

5. Go to the *Devices* Menu from VirtualBox and select *CD/DVD Devices*, and uncheck *VirtualGuestAdditions*.

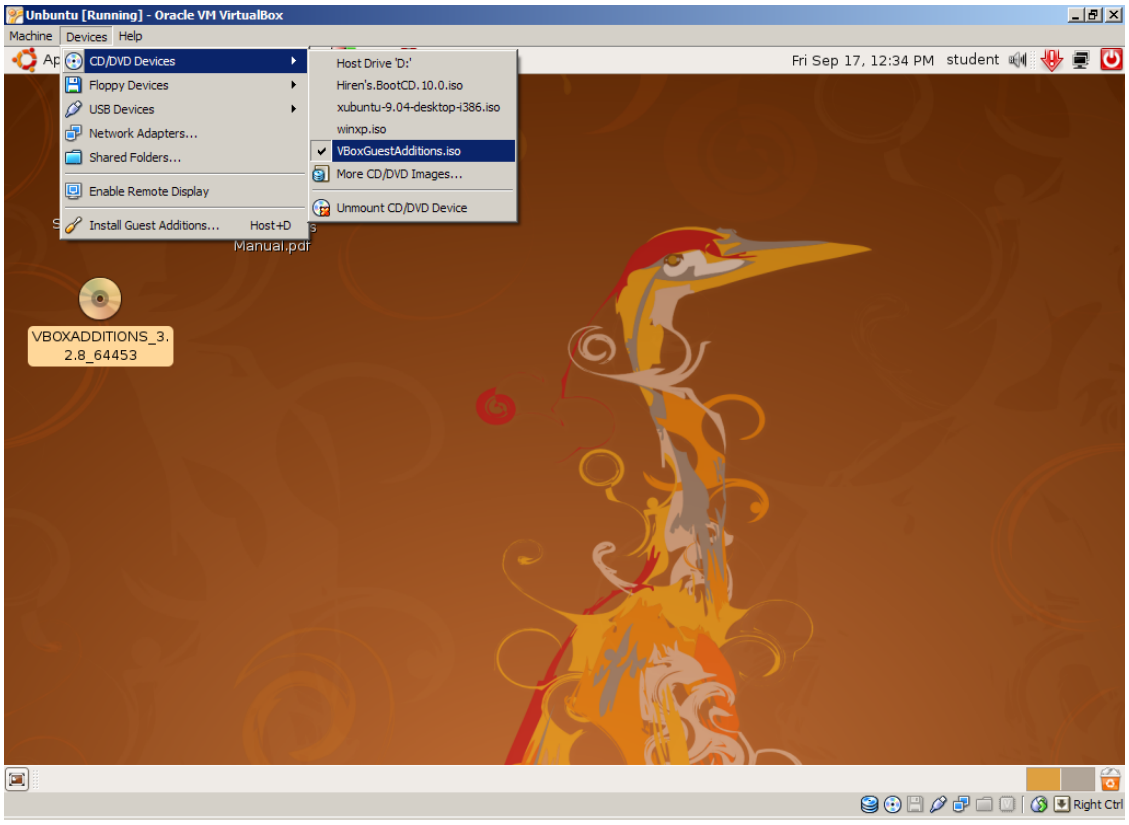


Figure 76

6. Shutdown the Ubuntu guest system and then restart Ubuntu for the guest additions to load.

Acknowledgements:

The images of the NXT Brick and Robot were obtained from the following sites:

- http://engk12.ece.missouri.edu/LegoCamp/pictures/NXT%20Robots/NXT%20Education%20Robot.jpg
- http://www.uaerobotchallenge.com/Pictures/General/images/NXT%20Brick_jpg.jpg
- http://lejos-osek.sourceforge.net/installation_linux_files/reset_nxt.JPG

John Hansen's Enhanced Firmware was obtained from:
http://bricxcc.sourceforge.net/lms_arm_jch.zip

nxtOSEK was obtained from:
http://lejos-osek.sourceforge.net/

Code::Blocks was obtained from:
http://www.codeblocks.org/

libNXT (firmware flashing tool):
http://code.google.com/p/libnxt/

nxtTransfer (nxt uploading tool):
http://nxttransfer.xdir.fr/