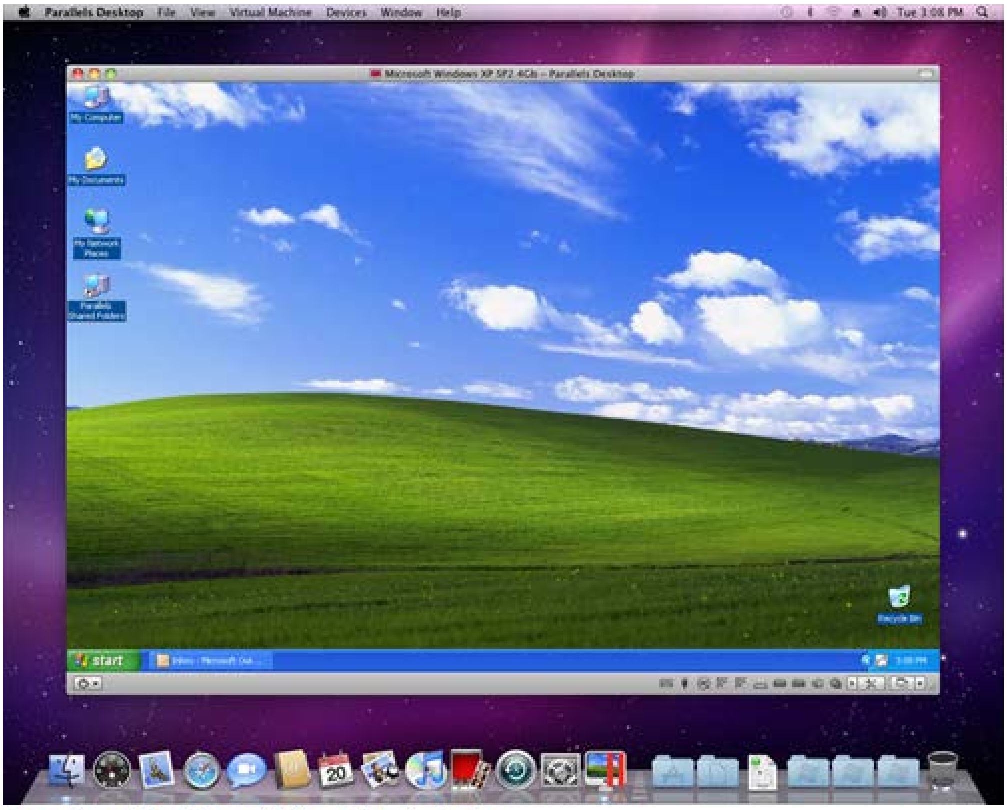
3.4.2 Virtual machine

Introduction:

A virtual machine is a software computer that, like a physical computer, runs an operating system and various applications.

Virtual machines allow you to run an operating system in a window on your desktop. A virtual machine could be used to run software made for other operating systems. For example, running a program built for Ubuntu on your Windows 8 computer. The virtual machine runs in a window on your main operating system.



Running Windows XP on Macintosh

3.4.2 Virtual machine

What is a Virtual Machine?

A virtual machine program is a computer program that creates a virtual computer system, complete with virtual hardware devices. This virtual computer “machine” runs as a process in a window on your current operating system. You can boot an operating system installer disc inside the virtual machine, and the operating system will be “tricked” into thinking it’s running on a real computer allowing you to install different operating systems on your virtual machine.

Your virtual machine’s operating system is stored on a virtual hard drive – a big, multi-gigabyte file stored on your Hard Drive. The file is presented to the operating system as a real hard drive. (This means you won’t have to mess around with your partitioning.)

Virtual machines add some overhead, so they won’t be as fast as if you had installed the operating system on real hardware. Demanding games that require serious graphics and CPU power are one particular problem, so virtual machines aren’t the ideal way to **Windows games on Linux or Mac OS X** – not unless those games are much older or aren’t graphically demanding. Also, when multiple virtual machines are simultaneously running on a host computer, each virtual machine may hang; this also depends on the workload on the system by other running virtual machines.

Why You’d Want to Create a Virtual Machine

Virtual machines allow you to experiment with another operating system without leaving your current OS. A virtual machine is also a great way to test out a new version of Windows. When the latest version of Windows comes out, it will have some bugs present in it. Hence, you can avoid system instability by installing it on a virtual machine.

When you’re done with an OS, you can just delete the virtual machine.

Virtual machines are also “sandboxed” from the rest of your system, which means that software inside a virtual machine can’t escape the virtual machine and tamper with the rest of your system. A virtual machine can be a good place to test out programs you don’t trust and see what they do. For example, if you ever come across the “Hi, we’re from Windows” scam, you can run their software in a virtual machine to see what it actually does. The virtual machine prevents scammers from accessing your computer’s real operating system and files.

All in all, a virtual machine has its pros and cons and is mainly used by program developers.

One example that we will discuss in detail is the **“Java Virtual Machine”**

3.4.2 Virtual machine

Java Virtual Machine (JVM)

Java is a high-level language used to write computer programs. Other examples of high level languages are C, C++, Pearl, Visual Basic, and Pascal.

When you write a computer program in high level language, the microprocessor needs to convert your code into machine language in order for the program to run. This translation process is different for every microprocessor.

Around the world, people use many kinds of microprocessors and in turn many kinds of operating systems. This creates a major issue; all microprocessors have a different machine code. (They have different instruction sets) This means that a program written on Intel microprocessor for example, will not run on an AMD microprocessor. **A program should work on all type of computers.**

A computer with x86 Based Intel microprocessor and windows operating system

A computer with x64 Based Intel microprocessor and windows operating system

A computer with x86_64 Based AMD microprocessor and Linux operating system

A computer with SPARC Based IBM microprocessor and Solaris operating system

Java tackles this problem beautifully, you write a program once and run it anywhere on any type of computer. In other words, Java is a platform independent language, which means that any program written using Java will run on any type of computer which Java supports (majority of computer do support java).

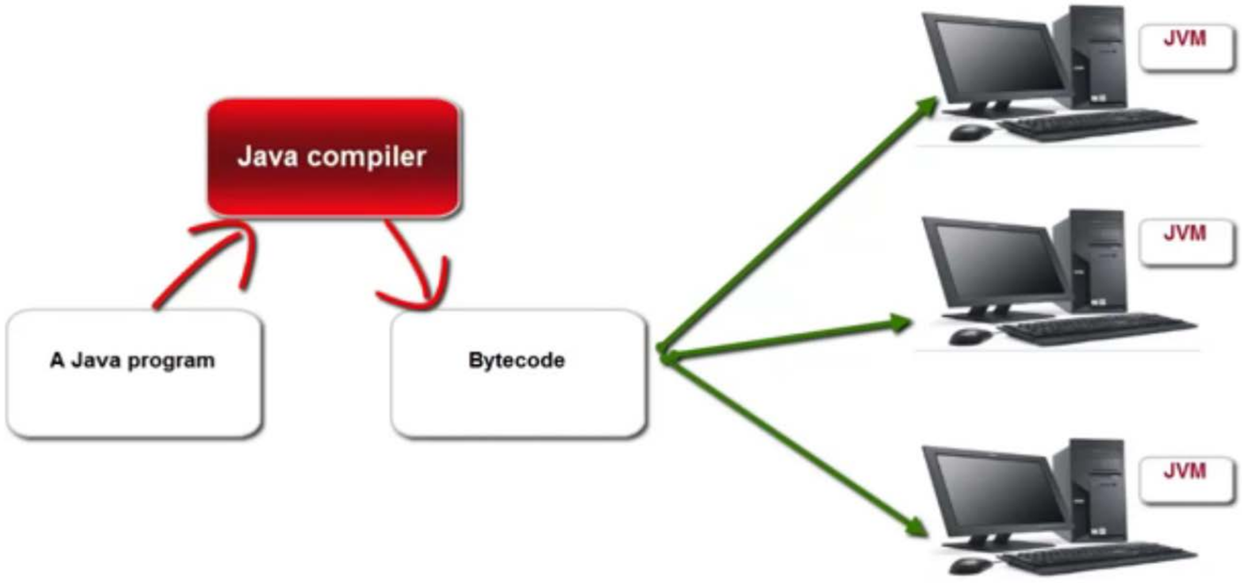
3.4.2 Virtual machine

So how does Java do it?

When you write a Java program and hit run, you pass your program to another program known as a **Complier**. The Complier will convert your high level code into a lower level code known as **Bytecode.** All computers have a **Java Virtual Machine** built in. Java provides a JVM for almost all kinds of computers, such as:

- JVM for Windows operating system X86 microprocessor
- JVM for Macintosh operating system X64 microprocessor
- Etc…

The JVM understands Bytecode language, its microprocessor's machine language, and knows how to establish communication between operating system and the bytecode program.



The JVM converts bytecode into the machine language. And, once the program is into its own machine language form, the computer can now run the program. This way, your bytecode is executed by a JVM on all different type of computers. So this way, a program written using Java can run on any computer or in other words, is a **Platform independent language** and the reason that Java code can run on any type of computer is because of the JVM program.

This is also a reason why Java runs fairly slower compared to other high level programming languages.