The major advantages of Java:Object OrientedWrite once, run anywhere. Java has better portability than any other languages.Support multi-threading, socket communication, automatic garbage collectionSupport web based application (JSP, Servlet, Applet), distributed application (RMI, EJB, Socket) and network protocols (HTTP)Because of it virtual machine it is more secure.

The Syntax of java is similar to C++ ,but java does not support Pointers,multiple inheritance,goto statement and operator overloading.Features of C++ that slow down application development cycle have been omitted in Java,like java has a garbage collector,so unlike C++ in java we dont need to deallocate the memory, or worry about memory fragmentations.

**Advantages of JAVA**  
  
JAVA offers a number of advantages to developers.  
  
Java is simple: Java was designed to be easy to use and is therefore easy to write, compile, debug, and learn than other programming languages. The reason that why Java is much simpler than C++ is because Java uses automatic memory allocation and garbage collection where else C++ requires the programmer to allocate memory and to collect garbage.   
  
Java is object-oriented: Java is object-oriented because programming in Java is centered on creating objects, manipulating objects, and making objects work together. This allows you to create modular programs and reusable code.  
  
Java is platform-independent: One of the most significant advantages of Java is its ability to move easily from one computer system to another.

The ability to run the same program on many different systems is crucial to World Wide Web software, and Java succeeds at this by being platform-independent at both the source and binary levels.  
  
Java is distributed: Distributed computing involves several computers on a network working together. Java is designed to make distributed computing easy with the networking capability that is inherently integrated into it.

Writing network programs in Java is like sending and receiving data to and from a file. For example, the diagram below shows three programs running on three different systems, communicating with each other to perform a joint task.  
  
Java is interpreted: An interpreter is needed in order to run Java programs. The programs are compiled into Java Virtual Machine code called bytecode.

The bytecode is machine independent and is able to run on any machine that has a Java interpreter. With Java, the program need only be compiled once, and the bytecode generated by the Java compiler can run on any platform.  
  
Java is secure: Java is one of the first programming languages to consider security as part of its design. The Java language, compiler, interpreter, and runtime environment were each developed with security in mind.   
  
Java is robust: Robust means reliable and no programming language can really assure reliability. Java puts a lot of emphasis on early checking for possible errors, as Java compilers are able to detect many problems that would first show up during execution time in other languages.   
  
Java is multithreaded: Multithreaded is the capability for a program to perform several tasks simultaneously within a program. In Java, multithreaded programming has been smoothly integrated into it, while in other languages, operating system-specific procedures have to be called in order to enable multithreading. Multithreading is a necessity in visual and network programming.  
  
  
**Disadvantages of JAVA**  
  
Performance: Java can be perceived as significantly slower and more memory-consuming than natively compiled languages such as C or C++.  
  
Look and feel: The default look and feel of GUI applications written in Java using the Swing toolkit is very different from native applications. It is possible to specify a different look and feel through the pluggable look and feel system of Swing.   
  
Single-paradigm language: Java is predominantly a single-paradigm language. However, with the addition of static imports in Java 5.0 the procedural paradigm is better accommodated than in earlier versions of Java.  
  
  
**Where to start?**   
  
**Install Java**  
  
Choose a text editor. You can use notepad, vi, BBEdit or another ASCII text editor. Do not use a word processor such as Microsoft Word.   
  
  
**First Java Program**  
  
In your favorite editor, enter the following code and save it in c:src directory in a file named HelloWorld.java.  
  
/\*\*  
  
\* The HelloWorld class is an application that  
  
\* displays "Hello World!" to the standard output.  
  
\*/  
  
public class HelloWorld {  
  
// Display "Hello World!"  
  
public static void main(String args[]) {  
  
System.out.println("Hello World!");  
  
}  
  
}  
  
To run your java program, you need to first compile it. Open a DOS or command window and change to the directory where you saved HelloWorld.java.   
  
C:>cd c:src  
  
The Java tool set consists of a set of command line programs located in the bin directory of your Java installation.

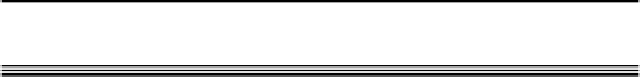
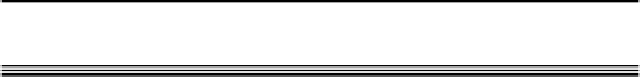
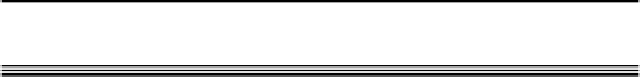
For instance, if your Java installation is in C:j2sdk1.4.2 then the programs you need will be located in C:j2sdk1.4.2bin.

Make sure you have modified your PATH environment variable to include this bin directory. You can modify your PATH through the control panel or temporarily on the DOS command line (path=c:j2sdk1.4.2bin;c:j2sdk1.4.2lib). You should test the Java command before compiling your code.

The following command should output version information. If it throws an error, set your PATH before continuing.   
  
C:src>java -version  
  
The "javac" command is used to compile .java files into intermediate bytecode files known as class files. Class files have a .class extension. The "java" command instantiates a JVM (Java Virtual Machine) instance and loads a Java class file that has a main method. The main method is the first code that is executed in your program.   
  
Now compile HelloWorld.java into Java bytecode with the javac command.   
  
C:src>javac HelloWorld.java  
  
  
If you do a dir command and see "HelloWorld.class" in the directory, then your compilation was successful.   
  
Now to run you first java program run the following command:  
  
  
C:src>java HelloWorld  
  
  
This should give the following output:  
  
Hello World!  
  
  
The above example merits a bit of explanation:  
  
All executable statements in Java are written inside a class, including stand-alone programs.   
  
Source files are by convention named the same as the class they contain, appending the mandatory suffix .java. A class which is declared public is required to follow this convention. (In this case, the class is Hello, therefore the source must be stored in a file called Hello.java).   
  
The compiler will generate a class file for each class defined in the source file. The name of the class file is the name of the class, with .class appended. For class file generation, anonymous classes are treated as if their name was the concatenation of the name of their enclosing class, a $, and an integer.   
  
The keyword void indicates that the main method does not return any value to the caller.   
  
The main method must accept an array of String objects. By convention, it is referenced as args although any other legal identifier name can be used. Since Java 5, the main method can also use variable arguments, in the form of public static void main(String... args), allowing the main method to be invoked with an arbitrary number of String arguments. The effect of this alternate declaration is semantically identical (the args parameter is still an array of String objects), but allows an alternate syntax for creating and passing the array.   
  
The keyword static indicates that the method is a static method, associated with the class rather than object instances.   
  
The keyword public denotes that a method can be called from code in other classes, or that a class may be used by classes outside the class hierarchy.   
  
The Java launcher launches Java by loading a given class (specified on the command line) and starting its public static void main(String[]) method. Stand-alone programs must declare this method explicitly. The String[] args parameter is an array of String objects containing any arguments passed to the class. The parameter to main are often passed by means of a command line.   
  
The method name "main" is not a keyword in the Java language. It is simply the name of the method the Java launcher calls to pass control to the program. Java classes that run in managed environments such as applets and Enterprise Java Beans do not use or need a main() method.   
  
The printing facility is part of the Java standard library: The System class defines a public static field called out. The out object is an instance of the PrintStream class and provides the method println(String) for displaying data to the screen while creating a new line (standard out).   
  
Standalone programs are run by giving the Java runtime the name of the class whose main method is to be invoked. For example, at a Unix command line java -cp . Hello will start the above program (compiled into Hello.class) from the current directory. The name of the class whose main method is to be invoked can also be specified in the MANIFEST of a Java archive (Jar) file (see Classpath).

In the evolution of Java, B gave birth to C. C evolved into C++, and C++ transmuted intoJava. Java is the language of the Internet. It was conceived by James Gosling, Patrick  Naughton, and Mike Sheridan at Sun Microsystems, Inc. in 1990 and took five years todevelop.: : Java can be used to create two types of programs: applications and applets. The outputof a Java compiler is not executable code. Rather it is bytecode. Java run-time system isan interpreter for bytecode. It is simply a highly efficient means of encoding a programfor interpretation. It is much easier to allow Java programs to run in a wide variety of environments. Once the run-time package exists for a given system, the bytecode versionof any Java program can run on it. Therefore, using bytecode to represent programs is theeasiest way to create truly portable programs.: : There are two surface similarities between Java and C++. First, Java uses a syntaxsimilar to C++, such as the general forms of the for, while, and do loops. Second, Javasupports object-oriented programming, same way as C++.: : There are also significant differences from C++, which fundamentally makes Javadistinct from C++. Perhaps the single biggest difference between Java and C++ is thatJava does not support pointers. Pointers are inherently insecure and troublesome. Since pointers do not exist in Java, neither does the -> operator. Some other C++ features arenot found in Java.: : „h Java does not include structures or unions because the class encompasses theseother forms. It is redundant to include them.: : „h Java does not support operator overloading.: : „h Java does not include a preprocessor or support the preprocessor directives.: : „h Java does not perform any automatic type conversions that result in a loss of  precision.: : „h All the code in a Java program is encapsulated within one or more classes.Therefore, Java does not have global variables or global functions.: : „h Java does not support multiple inheritance.: : „h Java does not support destructors, but rather, add the finalize() function.: : „h Java does not have the delete operator.: : „h The << and >> are not overloaded for I/O operations.: : „h Java does not support templates.: : Java shares many similarities with C++ as it relates to classes, but there are alsoseveral differences. By default, members of a class are accessible by other members of their class, derived classes, and by other members of their package. Therefore, classmembers are ¡§more public¡¨ than they are in C++, however, the private access specifier applies only to the variable or method that it immediately precedes. All class objects areinstantiated in Java using the new operator. Therefore, all class objects are dynamicallyallocated. When there are no references to an object, then the object is consideredinactive.

: : Java includes two class-management features that help make using and organizingclasses easier. The first is called a package, which defines a scope. Therefore, namesdeclared inside a package are private to that package. Java uses file directories to store packages. Therefore, each package must be stored in a directory that has the same nameas the package¡Xincluding capitalization.: : Java, like C++, supports hierarchies of classes. However, the way that inheritance isimplemented in Java differs substantially from the way that it is implemented in C++.Since multiple inheritance is not allowed in Java, then Java class hierarchies are linear. InJava, inheritance is referred to as subclassing. A base class in C++ is referred to assuperclass in JavaThe Syntax of java is similar to C++ ,but java does not support Pointers,multipleinheritance,goto statement and operator overloading.Features of C++ that slow downapplication development cycle have been omitted in Java,like java has a garbagecollector,so unlike C++ in java we dont need to deallocate the memory, or worry aboutmemory fragmentations.Java s biggest advantage is portability of code and its power lies in its APIs.It offerssupport to various types of applications and is a pure Object oriented language,while C++is not.It is not Platform independent,but performance wise it is better than Java.Datatypes have been defined fairly specifically, and should not vary between vendor implementations.Pointers are (generally) hidden or taken care of by the language.Very, very solid set of objects and methods for you to use. No multiple interitance, but there are interfaces for you to use thataddress a lot of the reasons why multiple inheritance exist.Error handling is very solid and done well.Javadocs.Good (but not great) implementation of OO.Bad things about Java:Why the hell are there primitives when there are also objects thatmirror the primitives as well?Sun's web site sucks for finding info about Java.Java on the client browser is dead.Think it's hard getting metrics about CF? Try getting non-BS performance stats for Java



Hmm, I haven't delved into C++ too much but it seems to me that C++ is justobject oriented C with a few compromises for C programmers just getting intoOO. Automatic memory management/garbage collection is nice too. Java justseems a little more evolved if you ask me. Not to mention the zillion free packages Sun, IBM, and the rest of the Java bandwagon have put out. I think this one has been debated a million times though...I think a more interesting question is, what advantages does C# have over Java? It's been a little while since I played around in Java (gonna have todust those books off again...), but I don't recall there being an equivalentto C#'s built in get/set properties. Very handy beasts if you ask me. I justfind C# a more comfortable language to work with overall, maybe it's just methough. MS definitely has taken advantage of the lessons learned from Java. jon> With the next version of ColdFusion being a Java base instead of a C /> C++ base, I'm asking the question...>> What are the benefits of Java over C++? Discuss amongst yourselves!>>Java is designed to be simple, object oriented and similar to C++ while removing theunnecessary complexities of C++. It is also said to be a robust, architecturally neutral, portable, interpreted, threaded, dynamic and high performance language. Java enables thedevelopment of robust applications on multiple platforms in heterogeneous, distributednetworks. C++ is not truly portable nor is it suited to heterogeneous, distributednetworks. While C++ excels in high performance, its powerful features and complexitiesare often the source of many errors.

http://htmlimg3.scribdassets.com/2bhy8bnao0bbjek/images/3-8923611e43.jpghttp://htmlimg3.scribdassets.com/2bhy8bnao0bbjek/images/3-8923611e43.jpg