JAVA DECOMPILER

By

MANIKANDAN A.V.

Reg. No: 71202702007

Of

KUMARAGURU COLLEGE OF TECHNOLOGY

COIMBATORE.

A PROJECT REPORT

Submitted to the

FACULTY OF SCIENCE AND HUMANITIES

In partial fulfillment of the requirements

for the award of the degree

of

MASTER OF SCIENCE

IN

APPLIED SCIENCE - COMPUTER TECHNOLOGY

JUNE, 2004.

BONAFIDE CERTIFICATE

Certified that this project report titled **JAVA DECOMPILER** is the bonafide work of **Mr. MANIKANDAN A.V.** who carried out the research under my supervision. Certified further, that to the best of my knowledge the work reported herein does not form part of any other project report or dissertation on the basis of which a degree or award was conferred on an earlier occasion on this or any other candidate.

PROJECT GUIDE **HEAD OF THE DEPARTMENT**

The candidate with university Registration no: 71202702007 was examined by us in the project viva-voce examination held on 17/6/04.....

INTERNAL EXAMINER (17-6-2004) **EXTERNAL EXAMINER**

ABSTRACT

The project work titled "**JAVA DE-COMPILER**" is basically system side project. It is developed using java for **Hindustan Software Limited,** Chennai.

The proposed system is a Complete Java Development Environment with Many additional features. It includes an Integrated Development Environment for Java with the Options of a regular editor to create new files, open existing files, save File, cut, copy, paste, undo, find, find next and replace.

It includes options to Compiler and Interpret Java Programs. The file that is open in this system is compiled, or interpreted or executed using applet viewer. It is necessary to note that only files with .Java extension can be compiled; only files with .class extension can be interpreted.

Developing a java De-compiler is to reconstruct the Java file from the class files. The de-compiler must be able to reconstruct the complete source code for the java files from its corresponding class files.

ACKNOWLEDGEMENT

The satisfaction of the successful completion of any task wouldn't be complete without the expression of gratitude to the people who helped to make it a successful one.

I here by take this opportunity to express my sincere gratitude to the management of Kumaraguru College of Technology, Coimbatore for giving me an opportunity to study in this esteemed institution.

I deem it a great privilege to express my profuse thanks to our beloved principal **Dr. K. K. Padmanabhan, PhD** for providing all facilities during the course of study. My sincere thanks to our Head of Department of Computer Science and Engineering **Dr. S. Thangasamy, PhD** who has been a support for his constant encouragement and unending enthusiasm to help me finish this project work.

I would like to express our thanks to Mr. Dinesh Ranganathan, M.S, Project internal guide, for his words of encouragement were the sparkles that has which helped me a lot to proceed my work.

It is my bounder duty to acknowledge my sincere and heart felt thanks to my project guide **Mr. Manikantan, MCA** lecturer Computer Science Department

I sincerely appreciate the guidance, and support given to me by our lecturers, throughout the completion of the project.

Finally I wish to extend my sincere thanks to all my friends who took a lot of effort and support to complete this project.

CONTENTS

LIST OF FIGURES

Chapter 1

INTRODUCTION

1.1 ABOUT THE PROJECT

A De-Compiler produces source code from a binary executable. This is the inverse operation of a compiler, which produces an executable from source code written in a particular source language.

Programs that are written in Java are particularly amenable to de-compilation because Java source code is typically compiled to Java byte code. Java byte code is a platform independent abstraction layer, which can be executed through a Java Virtual Machine.

The core design of Java byte code makes de-compilation considerably easier than most development languages currently in use. Java byte code contains interface and type information, which is not normally present in executables written in other source languages.

Java De-Compiler examines Java byte code to produce a flow graph. From this graph, Java De-Compiler determines a collection of statements (i.e. while, for, if, throw, etc) which when compiled could produce such a graph. Therefore the resulting source code may not be identical to the original, but will indeed have the same semantics.

The outputs of all class files can be readily decompiled with Java De-Compiler. There are a few constructs in arbitrary byte code, which we do not currently handle. While we intend to be able to decompile all verifiable byte code, the usefulness of some of the generated code may be limited.. It renames all illegal names and types in an intelligent manner. The result is highly readable code even when the executable has been obfuscated. Since simple renamingdoes not fundamentally change the execution of the byte code, algorithms are still quite clear when decompiled with Java De-Compiler.

Java De-Compiler was originally designed with the professional developer in mind. A common problem that arises when developing applets and applications in Java is difficult to understand the semantics of third party class libraries. These libraries frequently operate differently on multiple platforms making the portability of the Java programming language largely a myth. Java De-Compiler allows the developer to see the best possible documentation about a third party library: source code equivalent to the original.

Java De-Compiler particularly useful for recovery of lost or accidentally destroyed source code. While most developers are quite careful to make frequent backups of source, disaster still may occur. When it does, Java De-Compiler can recover days, weeks, or even months of development time. All that is needed for recovery is a recent executable.

It includes options to Compiler and Interpret Java Programs. We can also type the programs in java De-Compiler tool and can be saved as java file. The file that is open in this system is compiled, or interpreted or executed

Security professionals find Java De-Compiler useful in determining if applets or applications written in Java have a hostile or malicious design.

Analysis of compilers is another excellent use of Java De-Compiler. To determine the effectiveness of several compilers and optimizers, a user can compile the original source code and then decompile it with Java De-Compiler. The reconstructed source code will clearly reveal which optimizations were applied when generating the executable.

1.2. ABOUT THE ORGANIZATION

Hindustan Software Ltd., established in 1995, has been actively involved in the development of Customized Software for various clients since its inception and also has its presence strongly felt in the areas of Training & Consulting. Market Leadership through customer satisfaction, a commitment to excellence & high growth rate, these have characterized HSL in its rapid climb in the Information Technology Industry.

Our software development strives to develop innovative software's that meets customer needs. A team of developers with exclusive educational background and excellent computing skills exert their maximum efforts and endeavor to satisfy the specialized requirements of out esteemed clients. Out expertise and vast exposure qualifies us in developing software's in Information Systems, Web Based Applications, Internet Solutions, Web Page Designing and Hosting. Our team consists of expertise in JAVA, VB, Oracle and all Internet Application Development Tools.

Hindustan has trained more than 1000 professionals so far in the area of JAVA & VC++, almost all the students have launched in a very good career path both in India & Abroad. The curriculum matches latest trends and make the students to compete in the IT Industry.

The training division gives us an opportunity to meet variety of candidates with various skill sets. Our database consists of details regarding more than 1000

professionals. So we are fit to assist all sort of IT Human resource consultancies and IT companies in India and Abroad.

Excellence through "TEAM WORK" -The Philosophy of Total Quality Management is inculcated in every HSL employee through intensive training program. Our investment on a highly skilled and motivated manpower constitutes towards stabilizing HSL as truly quality conscious company, continually striving to bring the finest solutions to the discerning customers.

Our wide contacts within the IT industry allow us to locate & select Merit Students to be sent to prospective interviews to leading organizations having opening either in India or Abroad. We take this opportunity to inform the student community that most of our previous students have been immensely benefited from this service which comes free of cost.

1.3. GOALS OF THE PROJECT

- To create Integrated Development Environment for Java with the Options of a regular editor to create new files, open existing files, save File, cut, copy, paste, undo, find, find next and replace.

- To Compiler and Interpret Java Programs

- To reconstruct the Java file from the class files

- To Analyze the compilers

CHAPTER 2

SYSTEM STUDY

2.1 HARDWARE REQUIREMENTS

Processor	:	Intel Pentium IV
Memory	:	128 MB RAM
Hard Disk	:	20.1 GB HDD
Floppy Drive	:	3.5 Inches
CD Drive	:	48X Max
Monitor	:	VGA-color-Digital 15 Inches
Keyboard	:	108 Keys
Mouse	:	High sensitive Logitech

2.2 SOFTWARE REQUIREMENTS

FRONT END TOOL : JAVA DEVELOPMENT KIT 1.3

OPERATING SYSTEM : WIN 98, 2000, XP

2.3 SOFTWARE OVERVIEW

The development of Java wasn't as accidental or monumental as the discovery of penicillin, but it shares some of its characteristics. The origins of Java trace back to 1991, when Sun was investigating consumer electronics products. At this time, Mosaic and the World Wide Web were just interesting concepts.

James Gosling, the father of Java, was intent on building a low-cost, hardware-independent software platform using C++. For a number of technical reasons, C++ was dropped, and a new language, called Oak, was developed, based on C++, but eliminating its shortcomings. These shortcomings include problems associated with multiple inheritance, automatic type conversion, the use of pointers, and memory management.

Oak was used to develop a small electronics device called *7. This project resulted in the precursors of many of the components of Java: the development environment, runtime system, and API. The technology was explored in a number of consumer applications but was a little ahead of its time.

By 1994 the Web emerged, Oak was renamed Java, and the proverbial light bulb went on in the minds of the Java developers. Java was used as the basis for a Web browser, called WebRunner. WebRunner was successfully demonstrated, and the Java/HotJava project took off.

HotJava, Java, and the Java documentation and source code were made available over the Web, as an alpha version, in early 1995. Initially Java was hosted on SPARC Solaris, and then on Windows NT. In the summer of 1995, Java was ported to Windows 95 and Linux. In the fall of 1995 the Java Beta 1 version was released through Sun's Web site, and Java support was introduced in the Netscape 2.0 browser. The Java Beta 1 release led scores of vendors to license Java technology, and Java porting efforts were initiated for all major operating systems. In December 1995 the Java Beta 2 version was released, and JavaScript was announced by Sun and Netscape. Java's success became inevitable when, in early December, both Microsoft and IBM announced their intention to license Java technology.

On January 23, 1996, Java 1.0 was officially released and made available for download over the Internet. JavaScript was also released. Netscape 2.0 now provides support for both Java and JavaScript.

Java and HotJava

Java and HotJava are sometimes confused. Java is the language, development and runtime environments, and API. HotJava is a Web browser that is written in Java. HotJava highlights many of the Java features mentioned in the previous section.

HotJava is a Java-enabled browser. This means that HotJava can execute Java applets contained on Web pages. In order to accomplish this, HotJava calls

the Java runtime system. The Netscape 2.0 browser, like HotJava, is also Java enabled. It contains a copy of the Java runtime system embedded within it.

Both HotJava and Netscape request Web pages, written in HTML, from Web servers on the Internet. Since HotJava is written in Java, it uses the Java API and runtime system to display the Web pages to a user. Netscape, on the other hand, is written in C++ and uses C++ functions to display HTML documents.

When either browser encounters an APPLET tag in an HTML file, it requests the Java bytecode file necessary to execute the applet from the Web server. HotJava loads and executes the bytecode file using the Java runtime system. Netscape executes the bytecode file using an embedded version of the Java runtime system.

The Java Language

The Java language is a remarkable example of programming language evolution. Java builds on the familiar and useful features of C++ while removing its complex, dangerous, and superfluous elements. The result is a language that is safer, simpler, and easier to use. The following subsections describe Java in contrast to C++. Appendix B, "Differences Between Java and C++," provides a detailed identification of the differences between the two languages.

Java Is Familiar and Simple

If you have ever programmed in C++, you will find Java's appeal to be instantaneous. Since Java's syntax mirrors that of C++, you will be able to write Java programs within minutes. Your first programs will come quickly and easily, with very little programming overhead.

You will have the feeling that you have eliminated a lot of clutter from your programs-and you will have. All the cryptic header files and preprocessor statements of C and C++ are gone. All the arcane #define statements and typedefs have been taken away. You will no longer have to delve through several levels of header files to correctly reference API calls. And no one will have to suffer to figure out how to use your software.

Java programs simply import the software packages they need. These packages may be in another directory, on another drive, or on a machine on the other side of the Internet. The Java compiler and interpreter figure out what objects are referenced and supply the necessary linkage.

Java Is Safer and More Reliable

Java is safer to use than C++ because it keeps you from doing the things that you do badly, while making it easier to do the things that you do well.

Java won't automatically convert data types. You have to explicitly convert from one class to another. C++, under the most undesirable conditions, will automatically convert one type to another. It has all the flexibility of assembly code. Java doesn't assume that you know what you are doing. It makes sure that you do.

C++ pointers don't exist in Java. You can no longer access objects indirectly or by chance. You don't need to. You declare objects and reference those objects directly. Complex pointer arithmetic is avoided. If you need an indexed set of objects, you can use an array of objects. The concept of "the address of an object" is eliminated from the programming model, and another assembly language dinosaur is laid to rest. As a result, it becomes much easier to do things correctly in Java.

Java's reliability extends beyond the language level to the compiler and the runtime system. Compile-time checks identify many programming errors that go undetected in other programming languages. These checks go beyond syntactic checking to ensure that statements are semantically correct.

Runtime checks are also more extensive and effective. Remember your teacher or mom telling you to "Check your work twice to make sure it's right"? The Java linker understands class types and performs compiler-level type checking, adding redundancy to reliability. It also performs bounds checking and eliminates indirect object access, even under error conditions.

Java Is Secure

If you gave a skilled hacker a program written in C or C++ and told him to find any security flaws, there are half a dozen things that he would immediately look for: gaining access to the operating system, causing an unexpected return of control, overwriting critical memory areas, acquiring the ability to spoof or modify other programs, browsing for security information, and gaining unauthorized access to the file system.

Why is C or C++ more vulnerable than Java? When a programmer develops software, he or she usually focuses on how to get the software to work correctly and efficiently. C and C++ do not constrain the programmer from meeting these goals and provide a number of flexible features that enable the programmer to meet his end. The hacker is also able to take advantage of these features and use them in ways that weren't originally intended, causing the undesirable consequences identified in the previous paragraph. In short, C and C++ provide a great offense, but no defense. Java, on the other hand, is defensive by nature. Every time a Java-enabled browser downloads a compiled Java class, such as an applet, it runs the risk of running Trojan horse code. Because of this ever-present threat, it subjects the code to a series of checks that ensure that it is correct and secure.

The Java runtime system is designed to enforce a security policy that prevents execution of malicious code. It does this by remembering how objects are stored in memory and enforcing correct and secure access to those objects according to its security rules. It performs bytecode verification by passing

compiled classes through a simple theorem prover that either proves that the code is secure or prevents the code from being loaded and executed. The class is Java's basic execution unit and security is implemented at the class level.

The Java runtime system also segregates software according to its origin. Classes from the local system are processed separately from those of other systems. This prevents remote systems from replacing local system software with code that is less trustworthy.

Java-enabled browsers, such as HotJava, allow the user to control the accesses that Java software may make of the local system. When a Java applet needs permission to access local resources, such as files, a security dialog box is presented to the user, requesting explicit user permission. This "Mother may I?" approach ensures that the user always has the final say in the security of his system.

Java Is Multithreaded

Java, like Ada, and unlike other languages, provides built-in language support for multithreading. Multithreading allows more than one thread of execution to take place within a single program. This allows your program to do many things at once: make the Duke dance, play his favorite tune, and interact with the user, seemingly all at the same time. Multithreading is an important asset because it allows the programmer to write programs as independent threads, rather than as a convoluted gaggle of intertwined activities.

Multithreading also allows Java to use idle CPU time to perform necessary garbage collection and general system maintenance, enabling these functions to be performed with less impact on program performance.

Writing multithreaded programs is like dating several people concurrently. Everything works fine until the threads start to interact with each other in unexpected ways. Java provides the support necessary to make multithreading work safely and correctly. Java supports multithreading by providing synchronization capabilities that ensure that threads share information and execution time in a way that is thread safe.

Java Is Interpreted and Portable

While it is true that compiled code will almost always run more quickly than interpreted code, it is also true that interpreted code can usually be developed and fielded more inexpensively, more quickly, and in a more flexible manner. It is also usually much more portable.

Java, in order to be a truly platform-independent programming language, must be interpreted. It does not run as fast as compiled native code, but it doesn't run much slower, either. The Java Source Code," provides some Java performance benchmarks. For the cases where execution in native machine code is absolutely essential, work is underway to translate Java bytecode into machine code as it is loaded.

The advantages of being interpreted outweigh any performance impacts. Because Java is interpreted, it is much more portable. If an operating system can run the Java interpreter and support the Java API, then it can faithfully run all Java programs.

Interpreted programs are much more easily kept up-to-date. You don't have to recompile them for every change. In Java, recompilation is automatic. The interpreter detects the fact that a program's bytecode file is out-of-date with respect to its source code file and recompiles it as it is loaded.

Because of Java's interpreted nature, linking is also more powerful and flexible. Java's runtime system supports dynamic linking between local class files and those that are downloaded from across the Internet. This feature provides the basis for Web programming.

The Java API

The Java API is what makes Java attractive and useful. It consists of a set of packages that are distributed with the JDK as class libraries. These packages provide a common interface for developing Java programs on all Java platforms. The Java API furnishes all the capabilities needed to develop console programs, window programs, client and server networking software, applets, and other applications. It is the difference that takes Java from being a really good programming language to making it a very powerful and efficient programming environment.

The Java API consists of eight major development packages and a supporting debug package. *Packages* are collections of related objects. For example, there are separate packages for developing window programs, applets, and networking software.

Interface

Interfaces provide many advantages to the Java programmer. One is that they allow standard sets of methods to be used across the class hierarchy. For example, you can define the Editable interface to support cut, copy, and paste operations. The Editable interface can then be implemented by relevant classes and establish a uniform approach to implementing these common operations.

Interface types allow objects to be referenced by the methods they support without considering their location in the class hierarchy. They make maximal use of dynamic binding, allowing objects to be accessed independently of their implementation details. For example, parameters can be defined as interface types and used by methods. These methods can invoke the interface methods of their arguments without having to determine the classes to which the arguments belong.

Interfaces also support *selective* multiple inheritance. They allow various subsets of the features supported by different classes to be shared without mandating that all features of these classes be uniformly imposed as the result of inheritance.

Finally, because interfaces are declared independently of classes, they are unaffected by changes to specific classes or to the class hierarchy as a whole

Chapter 3

SYSTEM DESIGN

3.1 OVERVIEW OF PROPOSED SYSTEM

Using java De-Compiler tool we can type the java programs in the editor and can be saved in the required place. The programs can also be compiled and interpreted and can be executed in the child window. There is no need to use the MS-DOS for compiling and interpretation. it is easy to use .it has basic edit commands like cut ,copy ,paste so that we can make required editing.

Features of java De-Compiler

- ✓ Java De-Compiler is useful for recovery of lost or accidentally destroyed source code.
- ✓ Decompiling Java is an excellent way of learning both Java and how the Java VM works. Java De-Compiler makes it easy to peek into Java classes and learn from the source. Its easy to use and intuitive graphic user interface eases the learning curve for new starters in Java.
- ✓ Fixing and debugging *.class* files. Use Java De-Compiler when developers are slow to respond to questions that need immediate answers.
- ✓ Java De-Compiler is useful for exploring the sources of Java runtime libraries.
- ✓ Like the possibility to inline byte code.
- ✓ To check the results of your obfuscation

3.2 INPUT AND OUTPUT DESIGN

In the Input Design, the user-oriented inputs are converted in to computer recognizable format. The collections of input data in the most expensive part of the system in terms of equipment used, time and number of clients involved. In the Input Design data is accepted and it can be readily used for data processing or can be stored in the database for further use. Input Design is that part of design phase which requires the most attention. Data should be accurate because inaccurate data is the most common cause of errors in data processing. The input screens are very user friendly.

Open Dialog Screen

A Facility to open the Java File which we want to compile.

- **Input**:

The extent ion with .java files Process: The Software checks the given file is java file or not. If not an error will be displayed.

- **Output**:

The editor displays the java file.

Compile Screen

A facility to compile the java program which we opened.

- **Process:**

The Software Compiles the Java Program, and creates the class file. If there is an error it displays the corresponding error to the user.

- **Output**:

Displays the Byte Code.

Run Screen

A facility to run the Java Program by Clicking the Interpreter Menu.

- **Process:**

The Software takes the class file which we compiled and then Execute the Program.

- **Output**:

Displays the output in the Child Window

Open Dialog screen for Class File

- **Process:**

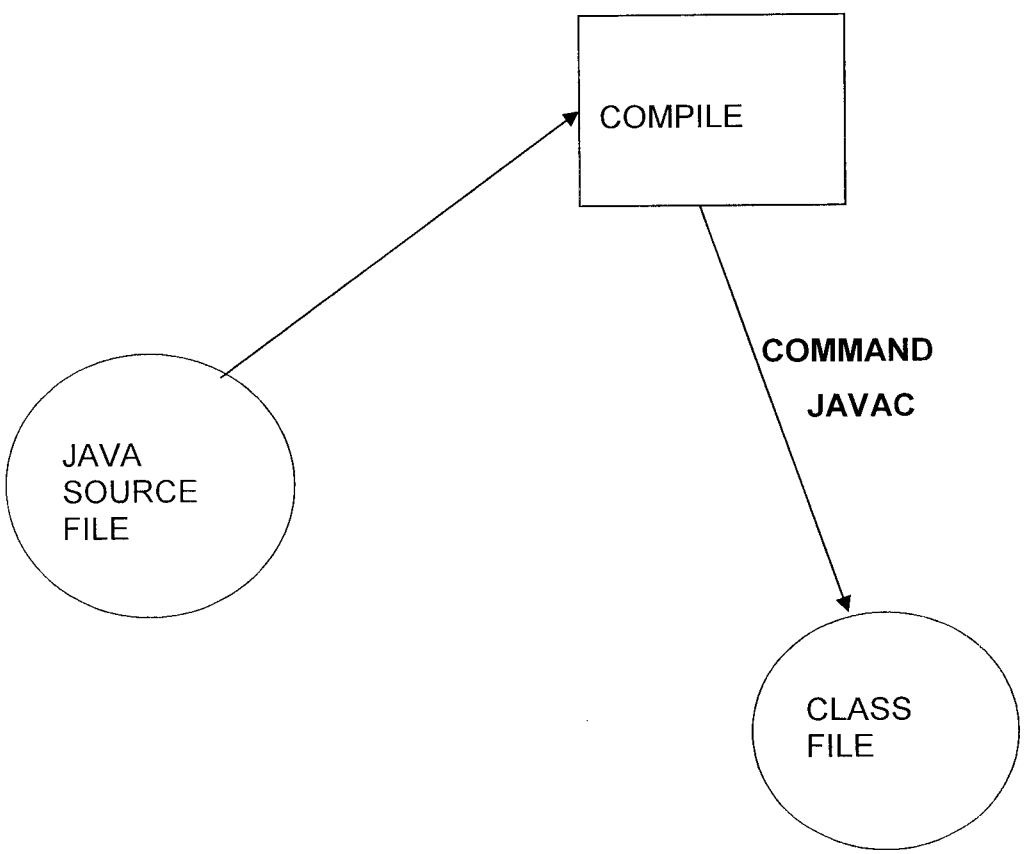
Converts the Byte Code into Respective Source code (JavaFile) in the Child window.

- **Output**:

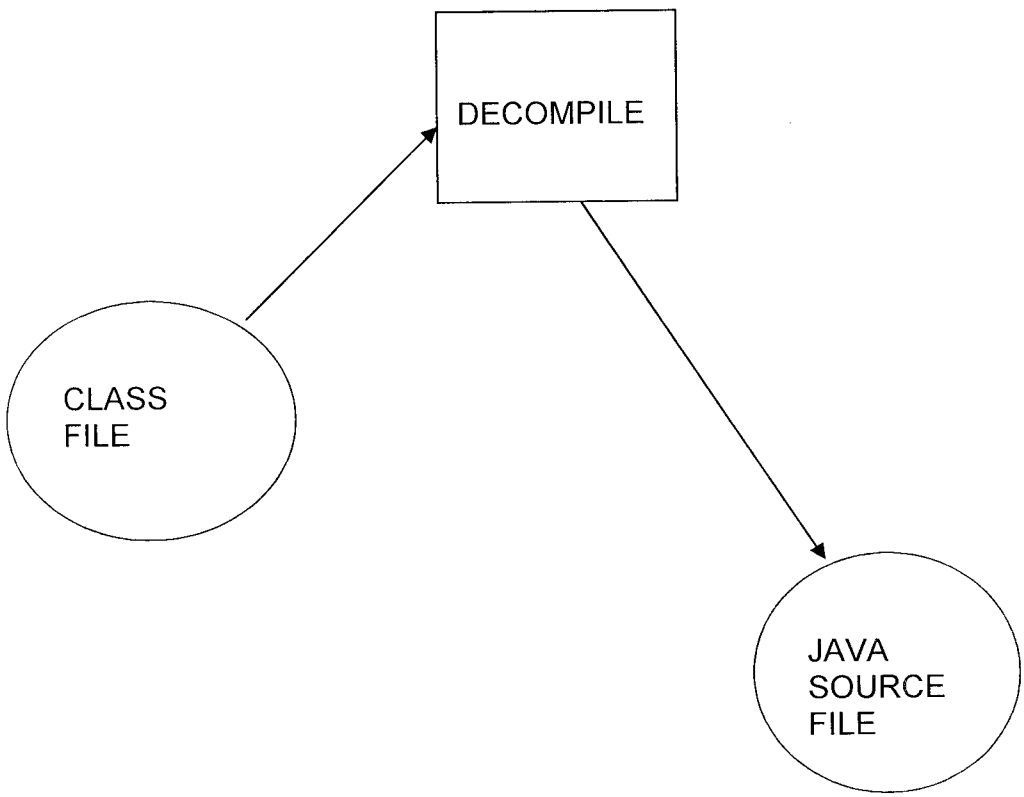
Displays the output in the Child Window

3.3 SYSTEM FLOW DIAGRAM

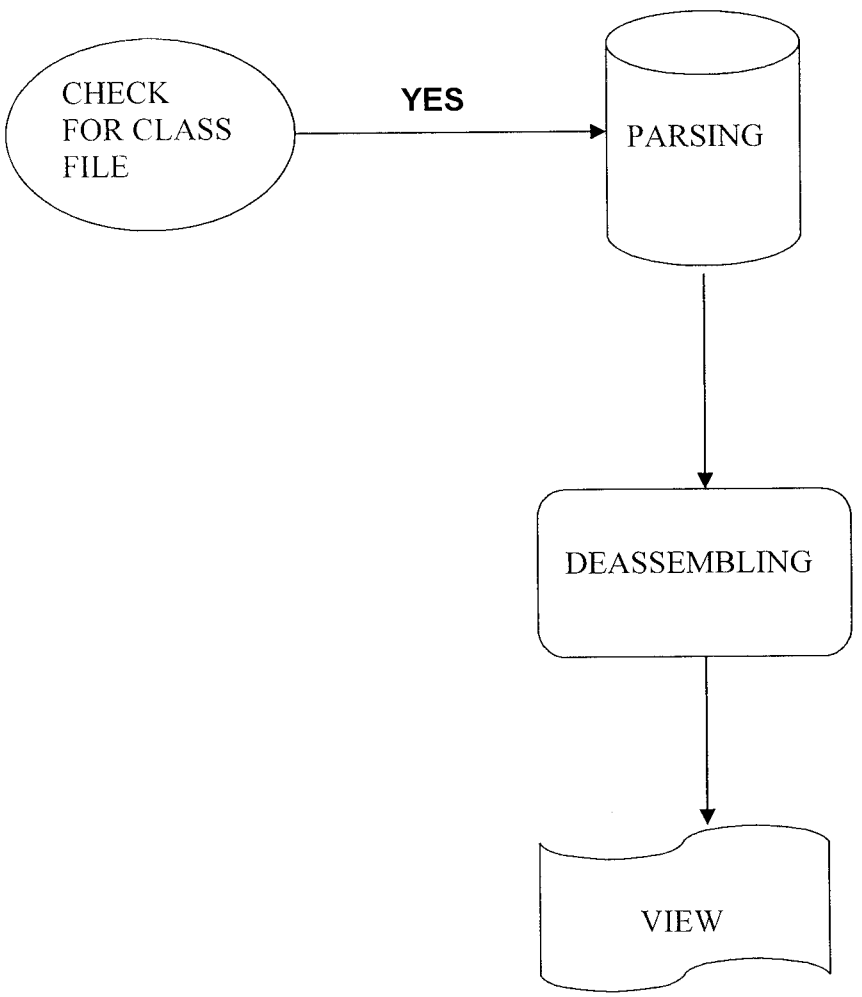
3.3.1COMPILATION PROCESS



3.3.2 DECOMPILATION PROCESS



3.3.3 PARSING TECHNIQUE



CHAPTER 4

TESTING AND IMPLEMENTATION

System testing forms another major part of any system development process. Care should be given during the whole process of testing. Inadequate testing or non-testing leads to errors that may not appear until months later. Effective testing early in the process translates directly into long-term cost savings from a reduced number of errors.

Another reason for system testing is its utility as a user-oriented vehicle before implementation. The performance of the system is measured in this phase.

TESTING METHODS

System testing begins by testing program modules separately, followed by testing "bundled" modules as a unit. A program module may function perfectly in isolation but fail when interfaced with other modules. The approach is to test each entity with successively large ones, upto the system test level.

System testing consists of the following steps.

- Code testing and Debugging
- Integration testing
- Data validation testing

CODE TESTING AND DEBUGGING

Testing is a process of executing a program with the interest of finding an error. A good test is one that has a high probability of finding the yet undiscovered error. The testing should systematically uncover different class of errors in a minimum amount of time with minimum amount of effort. Two classes of inputs are provided to the test process. They are

- A software configuration that includes a software requirements specification, Design specification and a source code.
- A test configuration that includes a test plan and procedure, any testing tools that is to be used and test cases and their expected result.

Testing is divided into 3 distinct operation viz Modular testing, Integration testing and data validation testing. In the series of testing the following tests are implemented.

INTEGRATION TESTING

Though each program works individually, they should work after linking them together. This is also referred to as Interfacing. Data may be lost across interface and one module can have adverse effect on another. Subroutines after linking may not do the desired function expected by the main routine. Integration testing is a systematic technique for constructing program structure while at

same time, conducting test to uncover errors associated with the interface. In this testing, the programs are constructed and tested in all segments.

DATA VALIDATION TESTING

Data validation is done to see whether the corresponding entries made in the tables are correct. Proper validation checks are done in case of insertion and updating of tables. Duplication of data has to be avoided to the maximum extend.

If any such case arises, then proper error messages or a warning has to be displayed. A double confirmation is made before deleting any specific entries. White box testing is a test case design method that uses the control structure of other procedurals designs to divide the test cases. The different test cases are:

- Guarantee that all independent parts within a module have been exercised at least once.
- Exercise all logical decision on their true/false side.
- Execute all loops at their boundaries and within their operational bounds.
- Exercise internal data structure to ensure their validity.

Each module was tested and the tested modules were linked and integrated.

4.2 IMPLEMENTATION

Implementation is the stage of the project where theoretical design is turned into a working system. At this stage, the main work load, the greatest upheaval and the major impact on the existing system shifts to the user department. If the implementation is not carefully planned and controlled, it can cause chaos and confusion. Thus it can be considered to be the most crucial stage in achieving a successful new system and giving the user confidence that the new system will work efficiently and effectively. It involves careful planning, investigation of current settings and its constraints on implementation, design of methods to achieve the change over. The more complex the system being implemented, the more involved will be the system analysis and the design effort required just for implementation.

Proper implementation is essential to provide a reliable system to meet the organization requirements. Successful implementation may not guarantee improvement in the organization using new system, but proper installation will prevent it.

The implementation stage involves the following task.

- Careful planning
- Investigation of system and constraints
- Design of methods to achieve the change over
- Training of staff in the change over phase
- Evaluation of the change over method.

The method of implementation and timescale to be adopted are found initially. Next the system is tested properly and at the same time users are trained in the new procedures.

4.3. MAINTENANCE

The maintenance is the enigma of system development .it holds the software industry captive. Maintenance can be classified as corrective, adaptive, or perfective.

Corrective maintenance means repairing processing or performance failures or making changes because of previously un corrected problems failures or making changes because of previously uncorrected problems or false assumptions.

Adaptive maintenance means changing the program function.

Perfective maintenance means enhancing the performance or modifying the programs to respond to the user's additional or changing needs.

Maintenance covers a wide range of activities, including correcting coding and design errors, updating documentation and test data, and upgrading user support. Maintenance means restoring something to its original condition.

4.4 FUTURE ENHANCEMENTS

The system is designed and developed in such away that further expansion or modification can be made to permit evaluation. the focus of the system is to inherit the requirements and update the software as per the needs.

This java de-compiler is so perfectly designed that it satisfies the programmers. so that if any further modifications in future they can add it. Suppose they want to add an applet viewer they can add.

CHAPTER 5

CONCLUSIONS

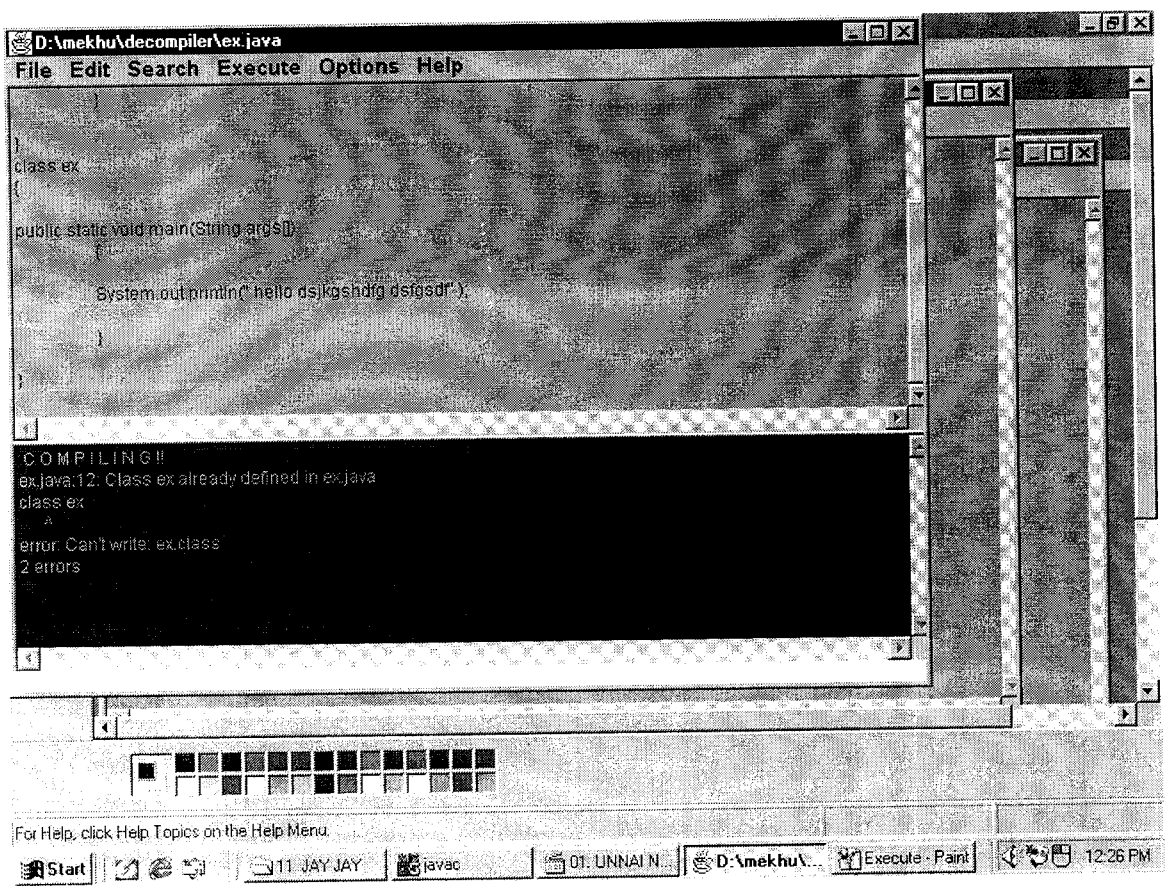
The objectives of the proposed system, having achieved, this system it to be functionally implemented in Hindustan Softwares Ltd. the software has got the management authorization, to be installed. The system is waiting for the feedback from the user so that the requirements can be rectified during the maintenance phase or can be updated in its later versions.

This system reduces the timework and has also resulted in quick programming for the programmers. This system works with higher degree of accuracy and user friendliness, which are very vital for the progress of the organization.

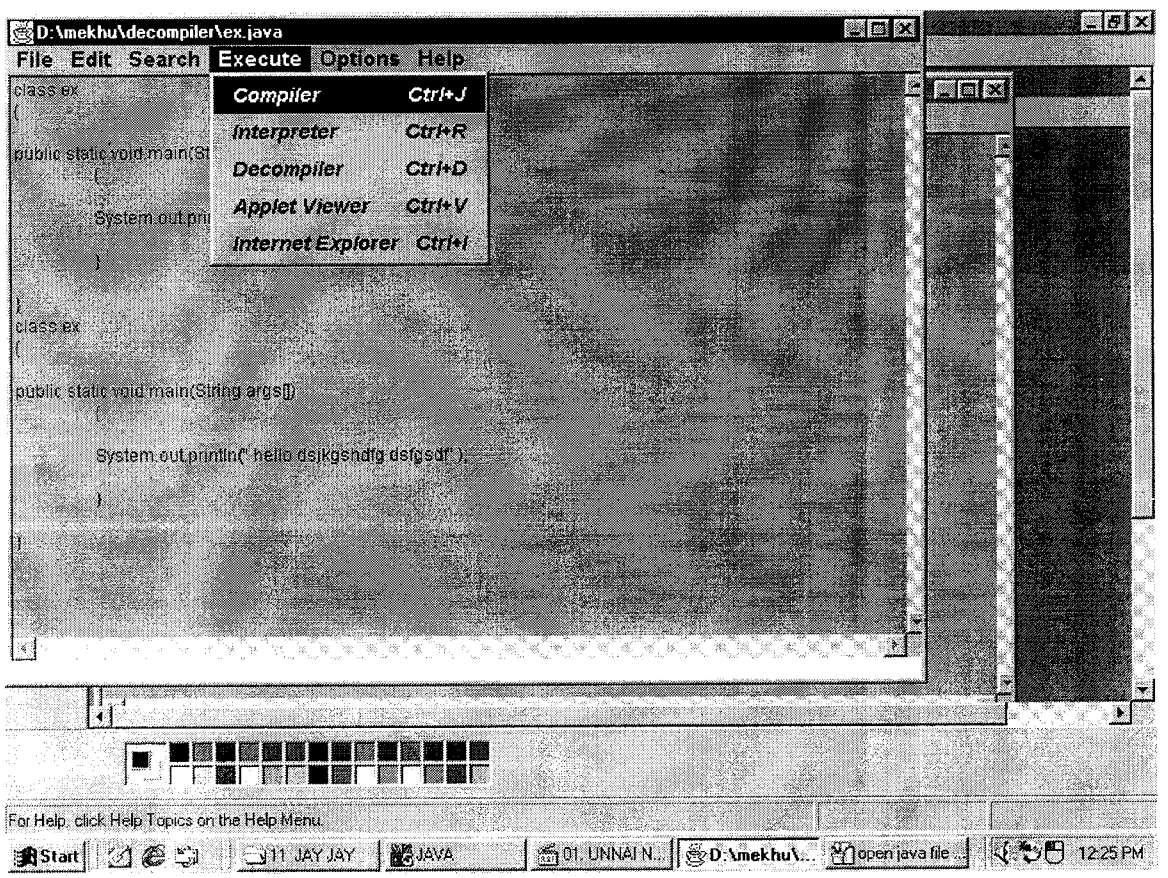
Analysis of compilers is another excellent use of Java de-compiler. To determine the effectiveness of several compilers and optimizers, a user can compile the original source code and then decompile it with Java de-compiler. The reconstructed source code will clearly reveal which optimizations were applied when generating the executable.

APPENDIXIES

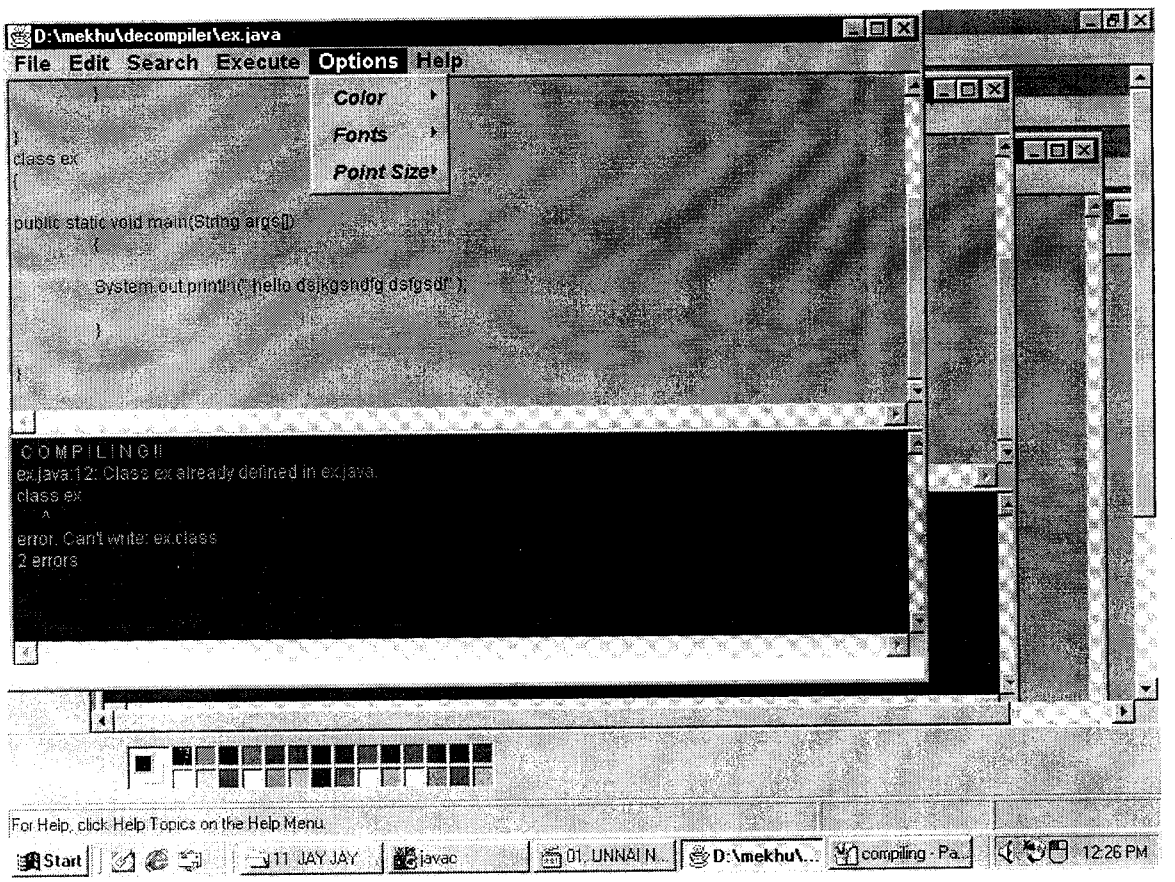
COMPILE

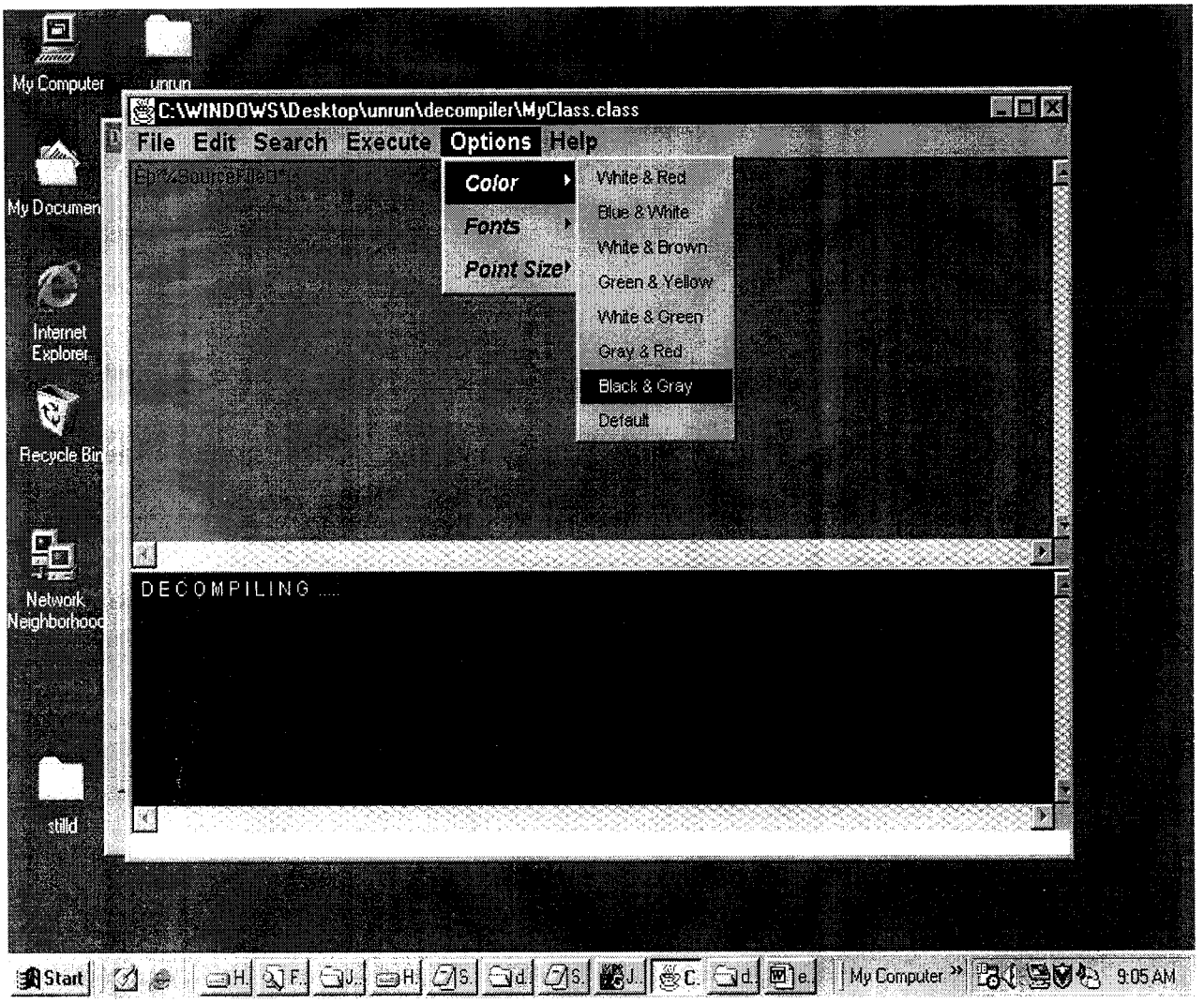


EXECUTE

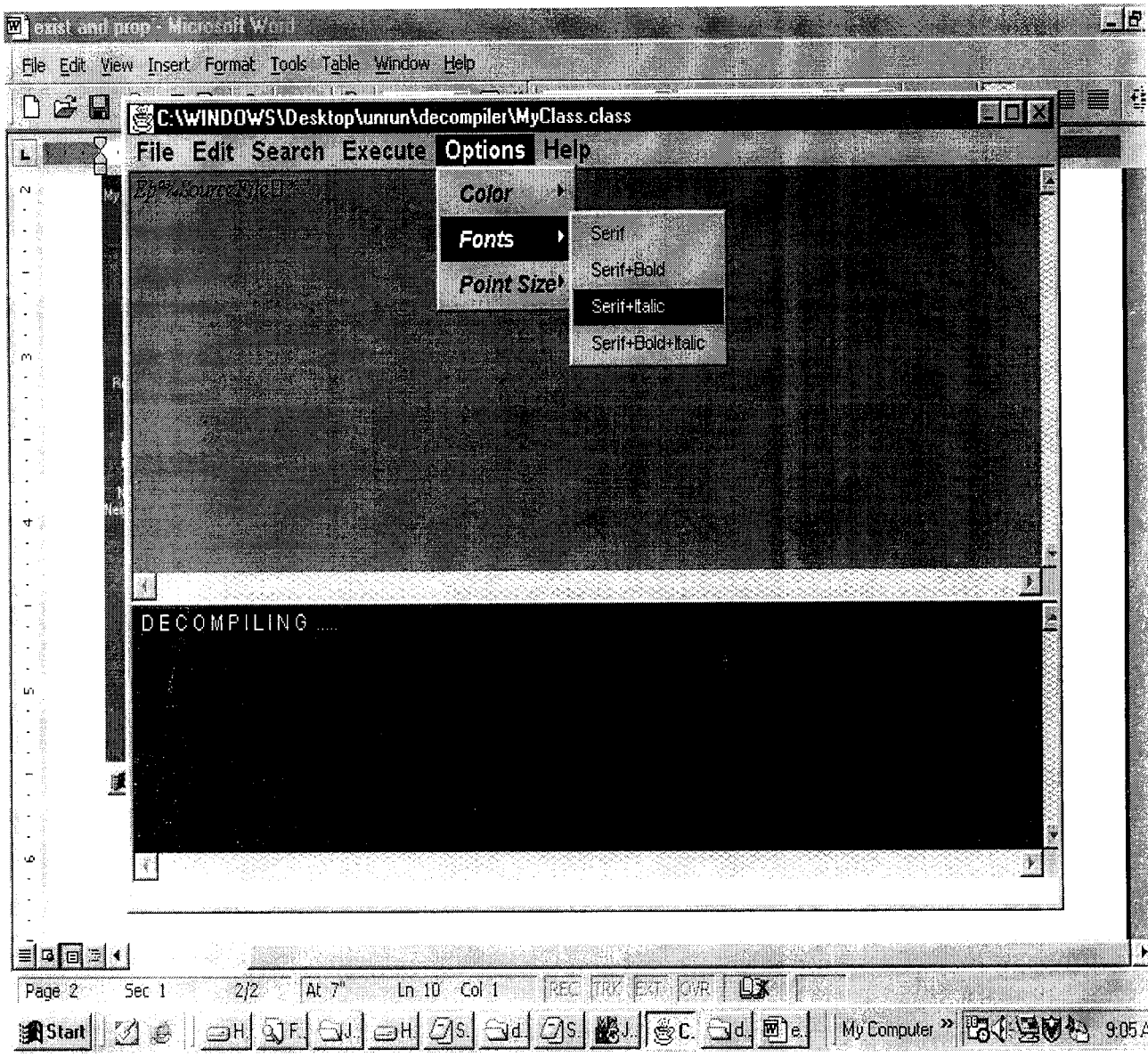

D:\mekhu\decompiler\ex.java
File Edit Search Execute Options Help
Compiler Ctrl+J
Interpreter Ctrl+R
Decompiler Ctrl+D
Applet Viewer Ctrl+V
Internet Explorer Ctrl+I
class ex
{
public static void main(St
{
System.out.pri
}
}
class ex
{
public static void main(String args[])
{
System.out.println(" hello dsjkgshdfg dsfgsdf ");
}
}
For Help, click Help Topics on the Help Menu.
Start
11 JAY JAY
JAVA
01. UNNAI N...
D:\mekhu\...
open java file ...
12:25 PM

OPTION

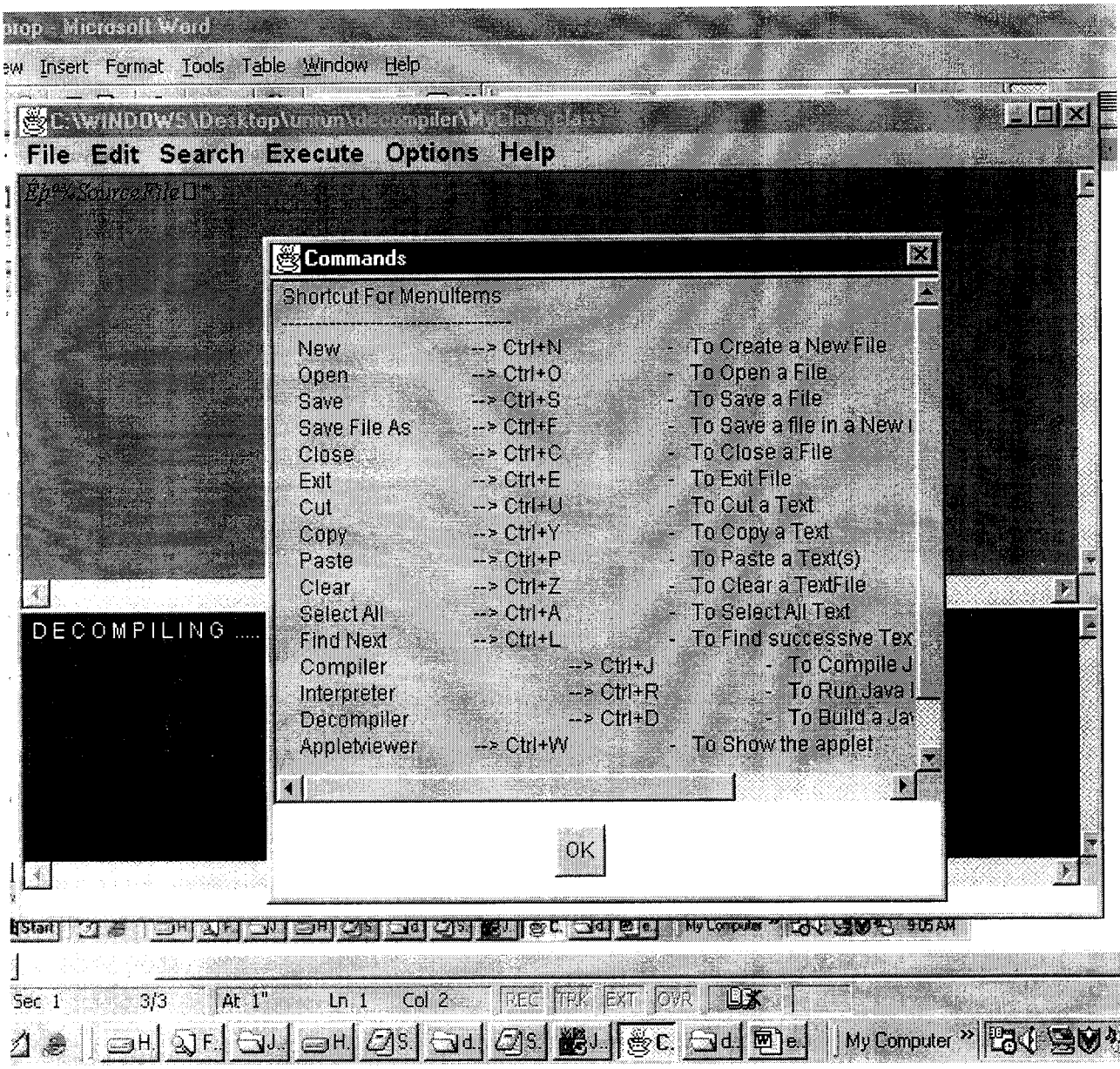

D:\mekhu\decompiler\ex.java
File Edit Search Execute Options Help
Color
Fonts
Point Size
class ex
{
public static void main(String args[])
{
System.out.println(" hello dsjkgsndfg dsfgsdf ");
}
}
COMPILING!!
ex.java:12: Class ex already defined in ex.java.
class ex
^
error, Can't write: ex.class
2 errors
For Help, click Help Topics on the Help Menu.
Start
11 JAY JAY
javac
01. UNNAI N...
D:\mekhu\...
compiling - Pa...
12:26 PM


My Computer
unrun
My Documen
Internet Explorer
Recycle Bin
Network Neighborhood
stilld
C:\WINDOWS\Desktop\unrun\decompiler\MyClass.class
File Edit Search Execute Options Help
Color
Fonts
Point Size
White & Red
Blue & White
White & Brown
Green & Yellow
White & Green
Gray & Red
Black & Gray
Default
DECOMPILING ....
Start
My Computer
9:05 AM

DE-COMPILE


File Edit View Insert Format Tools Table Window Help
C:\WINDOWS\Desktop\unrun\decompiler\MyClass.class
File Edit Search Execute Options Help
Color
Fonts
Point Size
Serif
Serif+Bold
Serif+Italic
Serif+Bold+Italic
DECOMPILING ....
Page 2 Sec 1 2/2 At 7" Ln 10 Col 1
Start
My Computer

SHORTCUTS


Insert Format Tools Table Window Help
File Edit Search Execute Options Help
Commands
Shortcut For MenuItems
New --> Ctrl+N - To Create a New File
Open --> Ctrl+O - To Open a File
Save --> Ctrl+S - To Save a File
Save File As --> Ctrl+F - To Save a file in a New
Close --> Ctrl+C - To Close a File
Exit --> Ctrl+E - To Exit File
Cut --> Ctrl+U - To Cut a Text
Copy --> Ctrl+Y - To Copy a Text
Paste --> Ctrl+P - To Paste a Text(s)
Clear --> Ctrl+Z - To Clear a TextFile
Select All --> Ctrl+A - To Select All Text
Find Next --> Ctrl+L - To Find successive Tex
Compiler --> Ctrl+J - To Compile J
Interpreter --> Ctrl+R - To Run Java
Decompiler --> Ctrl+D - To Build a Ja
Appletviewer --> Ctrl+W - To Show the applet
OK
DECOMPILING .....
Sec 1 3/3 At 1" Ln 1 Col 2
My Computer

REFERENCES

1. Herbert Schildt "Java 2: the Complete Reference, Fifth Edition" Tata McGraw-Hill Edition 2002.
2. Java Course Manual2: Micro hard Institute Of Technology
3. Unleaded Java2.0: Fifth Edition" Tata McGraw-Hill Edition 2001.
4. System Analysis And Design: Award E.M, Galgotia Pub, 1991

WEBSITE

www.hotjava.com