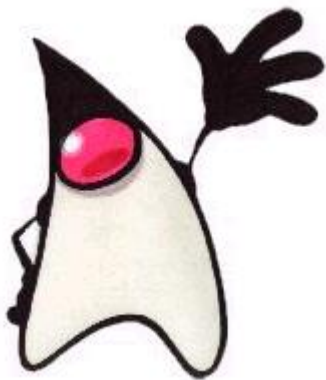
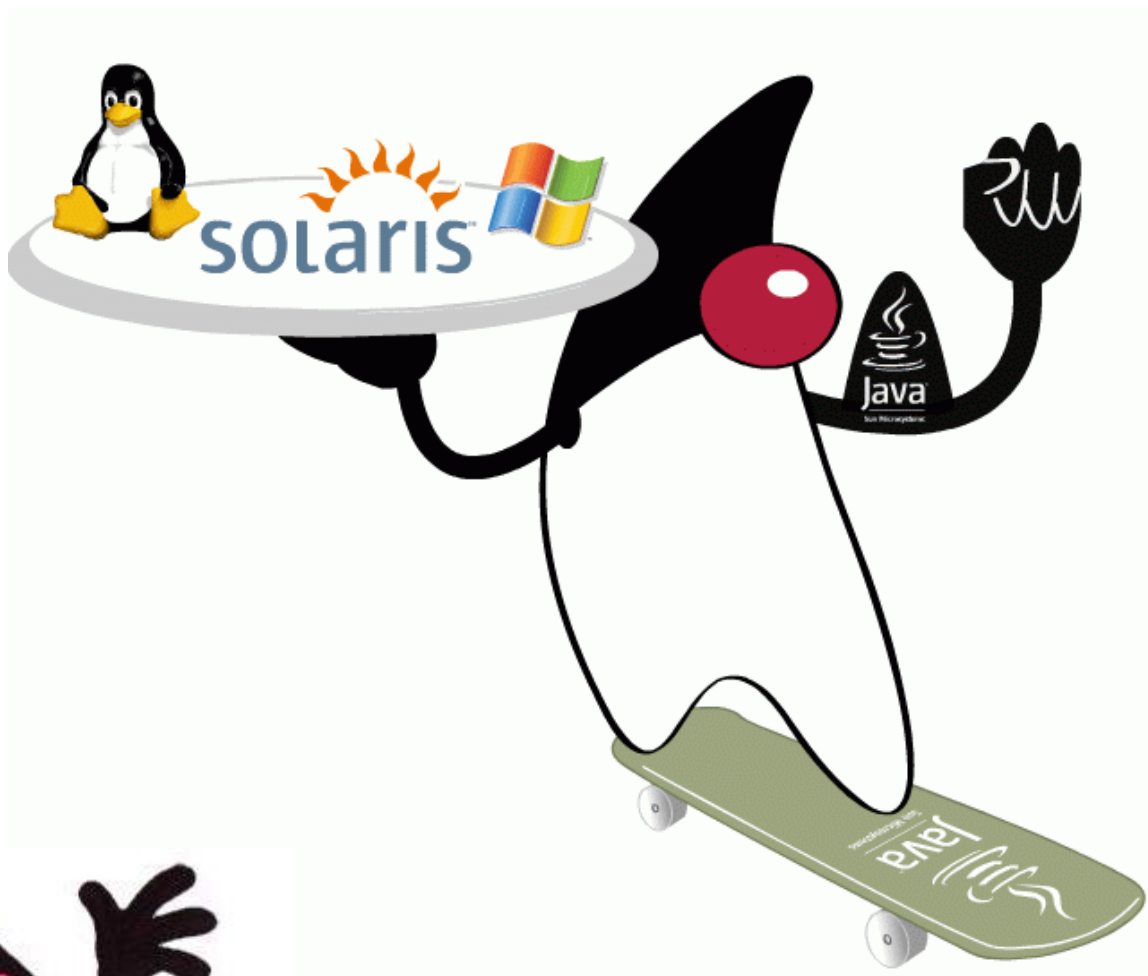
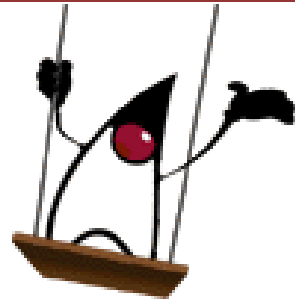


2007

Department of Management Science and Technology
Athens University of Economics and Business

Αδαμόπουλος Παναγιώτης Α.Μ. 8040000
Τόδρη Βίλμα – Γεωργία Α.Μ. 8040140



[Advanced Topics in Software Engineering]

Contribute to an open-source project - jLab



Οικονομικό Πανεπιστήμιο Αθηνών
τμήμα διοικητικής επιστήμης και τεχνολογίας

Index

Summary.....	4
Abstract.....	5
Message from the owner	6
Description.....	6
Instructions	7
Project Role - Membership.....	7
CVS.....	8
Mailing Lists	8
Version Control with Subversion – Breadth of changes	9
Understanding and documentation of legacy system	12
Consistency in Formatting	14
Screenshots of some changes	15
Summary of the changes	18
Code examples	21
Fix code	23
Formatting Java Source Code	24
Batch files.....	25
Integration	26
Testing	27
Scenarios	27
Debug.....	28
Findbugs 1.2.1	29

ckjm-1.7	32
Coordination with the development team – Mails	36
Documentation	37
Blog	40
Working Team	42
Advanced Topics in Software Engineering	43

Summary

Summary A scientific open-source programming environment coded in Java

Categories None

License [GNU General Public License \(GPL\)](#)

Owner [sterg](#)

Approximate source code size:

- Main project 570+ classes
- Toolbox 130+ classes

After our contribution:

- Main project 650+ classes



Abstract

The jLab environment provides a Matlab/Scilab like scripting language that is executed by an interpreter implemented in the Java language.

This language supports all the basic programming constructs and an extensive set of built in mathematical routines that cover all the basic numerical analysis tasks. Moreover, the toolboxes of jLab can be easily implemented in Java and the corresponding classes can be dynamically integrated to the system. The efficiency of the Java compiled code can be directly utilized for any computationally intensive operations. Since jLab is coded in pure Java the build from source process is much cleaner, faster, platform independent and less error prone than similar C/C++/Fortran based open source environments (e.g. Scilab, Octave).

Neuro-Fuzzy algorithms can require enormous computation resources and at the same time an expressive programming environment. We demonstrate the potentiality of jLab by describing the implementation of a Support Vector Machine (SVM) toolkit and by comparing its performance with a C/C++ and a Matlab version and across different computing platforms (i.e. Linux, Sun/Solaris, Windows XP).

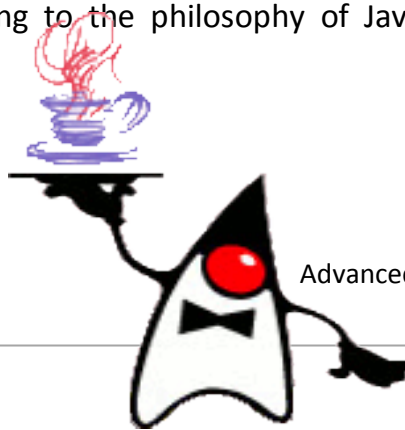


Message from the owner

The jLab project aims to provide a Matlab/Scilab environment with a scripting interpreter implemented in Java, and with the potential of linking dynamically Java numerical computing code. The system will perform very efficiently since the Java class code executes very fast. Moreover the potentiality for distributed execution can be explored.

Description

The jLab environment aims to provide a Matlab/Scilab like scripting language that is executed by an interpreter implemented in the Java language. This language will support all the basic programming constructs and an extensive set of built in mathematical routines that cover all the basic numerical analysis tasks. Moreover, the toolboxes of jLab can be easily implemented in Java and the corresponding classes can be dynamically integrated to the system. The efficiency of the Java compiled code can be directly utilized for any computationally intensive operations. Since jLab will be coded in pure Java the build from source process is much cleaner, faster, platform independent and less error prone than similar C/C++/Fortran based open source environments (e.g. Scilab, Octave). Also the facilities of the Java language for distributed computation will be explored to speed up scientific computations. Adhering to the philosophy of Java language, jLab is totally platform independent.



Instructions

The source code can be compiled directly on the Netbeans platform, and runs by selecting `jExec.jLab.jLab` as the main class, and by specifying as a run time argument the full path to the jar file that the Netbeans create at the "dist" folder.

Project Role - Membership

The screenshot shows the 'jlab: Membership' page in Mozilla Firefox. The browser address bar shows the URL `https://jlab.dev.java.net/servlets/ProjectMemberList`. The page header includes the Java logo and 'java.net The Source for Java Technology Collaboration'. The user is logged in as 'padamop' with a 'Logout' link. The page content is organized into a sidebar with 'My pages', 'Projects', 'Communities', and 'java.net' tabs. Under 'Projects', the 'jlab' project is selected, showing 'Membership' details. A 'Request project role' link is visible. Below this is a table of members with columns for 'User', 'Full name', 'Roles', and 'Assigned issues'. The table lists three members: 'gtodri' (Developer), 'padamop' (Developer), and 'sterg' (Project Owner, Click-Through User, Before Click-Through). Each member has a 'View issues' link. A 'Revoke checked roles' button is at the bottom of the table. The browser's taskbar at the bottom shows several open applications, including '2 Messenger', 'jlab', '2000 - Microsoft W...', 'jlab: Membership - ...', and 'Δ.E.T - Music'.

User	Full name	Roles	Assigned issues
gtodri	gtodri	Developer	View issues
padamop	padamop	Developer	View issues
sterg	sterg	Project Owner, Click-Through User, Before Click-Through	View issues

Both of us worked on project as developers.

CVS



Accessing the source code repository

Access the source code repository for this project in one of following ways:

- Browse source code online to view this project's directory structure and files.
- Check out source code with a CVS client using the following command.

Note: replace `username` with your own username.

```
cvs -d :pserver:username@cvs.dev.java.net:/cvs login
```

followed by

```
cvs -d :pserver:username@cvs.dev.java.net:/cvs checkout jlab
```

Mailing Lists

- ❖ dev@jlab.dev.java.net
- ❖ cvs@jlab.dev.java.net
- ❖ commits@jlab.dev.java.net
- ❖ anounce@jlab.dev.java.net
- ❖ issues@jlab.dev.java.net
- ❖ users@jlab.dev.java.net

Advance



Version Control with Subversion – Breadth of changes

At our work directory home at Ionian (**FreeBSD Unix**) we execute the following command before we import our projects to the repository, we have to create the repository

```
svnadmin create svnroot //create the repository
```

Now, if we list the directory contents we will see that the directory svnroot was created and contains some other files and directories (README.txt, conf, dav, db, format, hooks, locks).

The repository URL will then be:

```
svn+ssh://<username>@ionian.dmst.aueb.gr:/disk2/shome/Students2004/<username>/svnroot 1
```

Then, if the project isn't in our directory we have to download and unarchive it

```
tar zxvf jLabSrcArxiko.tar.gz // extracts files from an archive file in tar format
```

After that we import our project called jLabSrcArxiko (initial version) to a repo path

```
svn import jLabSrcArxiko  
svn+ssh://<username>@ionian.dmst.aueb.gr:/disk2/shome/Students2004/<username>/svnroot/jLabSrcArxiko 2
```

//Commit an unversioned tree into the repository

¹ Username can be either padam or vtodri

² All commands must be executed in the project's directory

Afterwards, we check out a working copy from our repository in order to store locally a specific project version

```
svn co  
svn+ssh://<username>@ionian.dmst.aueb.gr/disk2/shome/Students2004/<user  
name> /svnroot/jLabSrcArxiko jLabSrcArxiko1
```

For now and on every change we make in the directory jLabSrcArxiko1 we have to commit it or add it to the repository in order to send local changes to the repository.

```
Svn status //Print the status of working copy files and directories
```

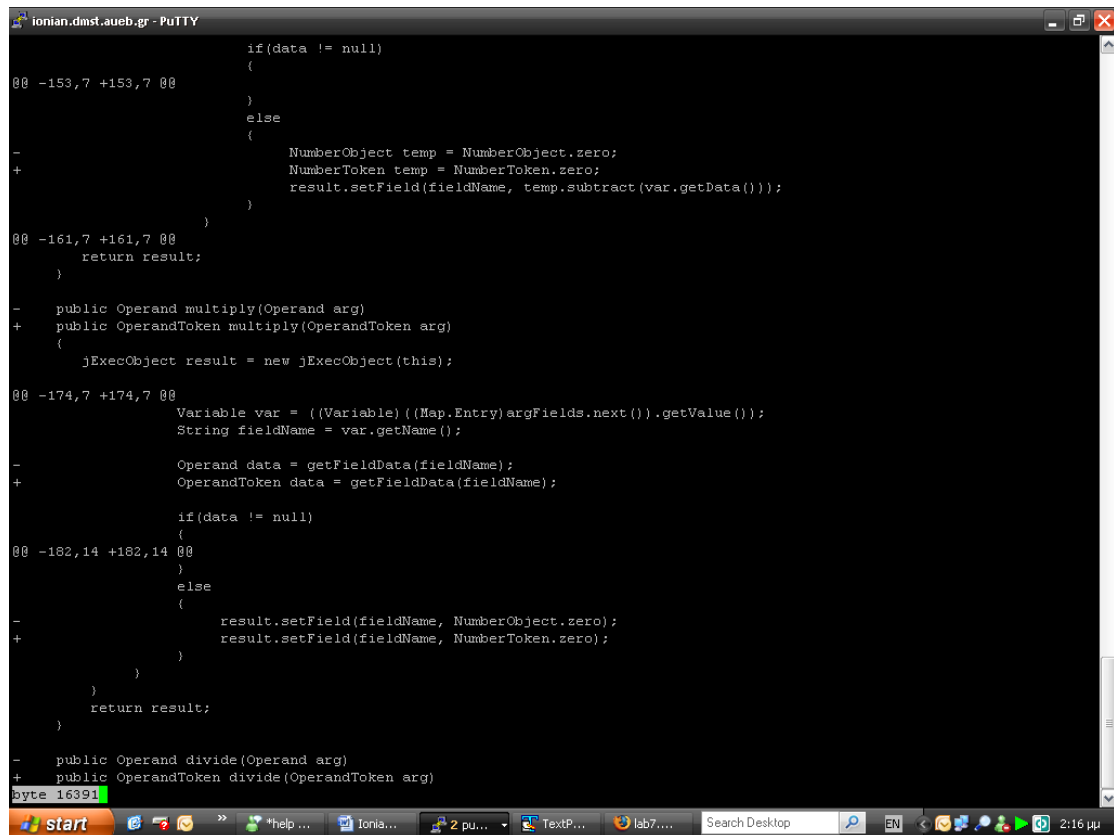
In the results of the above command first column says if an item was added, deleted, or otherwise changed.

Finally, when we decide so we can send changes from your working copy to the repository through the following command:

```
Svn commit //for modified files
```

```
Svn add <file> //for new files
```

Using this useful tool we can keep track of our changes and different versions of the project. We can see the differences between two versions and examine the summary of changes through time.



Picture 1: Watching the differences of two versions

In order to display differences between versions we execute the command:

```
svn diff -r1:14 | more
```

But if we want to have an idea about the breadth of changes we can execute this command (through pipe) followed by another that shows how many lines were added.

```
svn diff -r1:14 | grep \+ | wc -l    //show the differences between version 1
and 14 and grep the number of lines that were added
```

The answer was 672 lines.

Understanding and documentation of legacy system

In gemini (Red Hat linux 9 Server) we can list contents of directories in a tree-like format. In this way we can see how our project is organized.

We execute the command `tree -dl jLabSrc`

```
.
|-- jLabSrc
    |-- Graph
    |-- jExec
    |   |-- Constants
    |   |-- Functions
    |   |   |-- Bench
    |   |   |-- Chaotic
    |   |   |-- Crypto
    |   |   |-- DataFormats
    |   |   |-- FunFun
    |   |   |-- General
    |   |   |-- Graphics
    |   |   |   |-- Graph2d
    |   |   |   |-- Graph3d
    |   |   |   |-- OGraph2d
    |   |   |   `-- OGraph3d
    |   |   |-- Internal
    |   |   |-- Matrix
    |   |   |   |-- _private
    |   |   |   |-- Jama
    |   |   |   `-- util
    |   |-- Net
    |   |-- Neural
    |   |   |-- SVM
    |   |-- Signal
    |   |-- Statistics
    |   |-- String
```

```
| | |-- System
| | |-- Time
| | |-- Trigonometric
| | |-- Wavelets
| | |-- io
| | |-- numanal
| | `-- poly
| |-- Graphics
| |-- Interfaces
| |-- Interpreter
| |-- OGraphics
| |-- Tokens
| |-- Tools
| | `-- TreeAnalyser
| |-- gui
| `-- jLab
|-- jLab
| `-- help
|-- jLabEdit
| `-- resources
|-- neuralLib
|-- numal
| |-- TestFourier
| |-- Test_ch1
| |-- Test_ch2
| |-- Test_ch3
| |-- Test_ch4
| |-- Test_ch5
| |-- Test_ch6
| `-- Test_ch7
|-- utils
|-- wavelets
`-- weka
```

60 directories in JLabSrc

Consistency in Formatting

Opening brace on separate line.

```
// create matrix
double[][] values = new double[a_dy][a_dx];
for (int xi=0; xi<a_dx ; xi++)
{
    for (int yi=0; yi<a_dy ; yi++)
    {
        if ( ((a[yi][xi] != 0.0) && (b[yi][xi] != 0.0)
            || ((a[yi][xi] == 0.0) && (b[yi][xi] == 0.0)
            {
                values[yi][xi] = 0.0;
            }
            else
            {
                values[yi][xi] = 1.0;
            }
        }
    }
}
return new NumberObject(values);
} // end eval
}
```

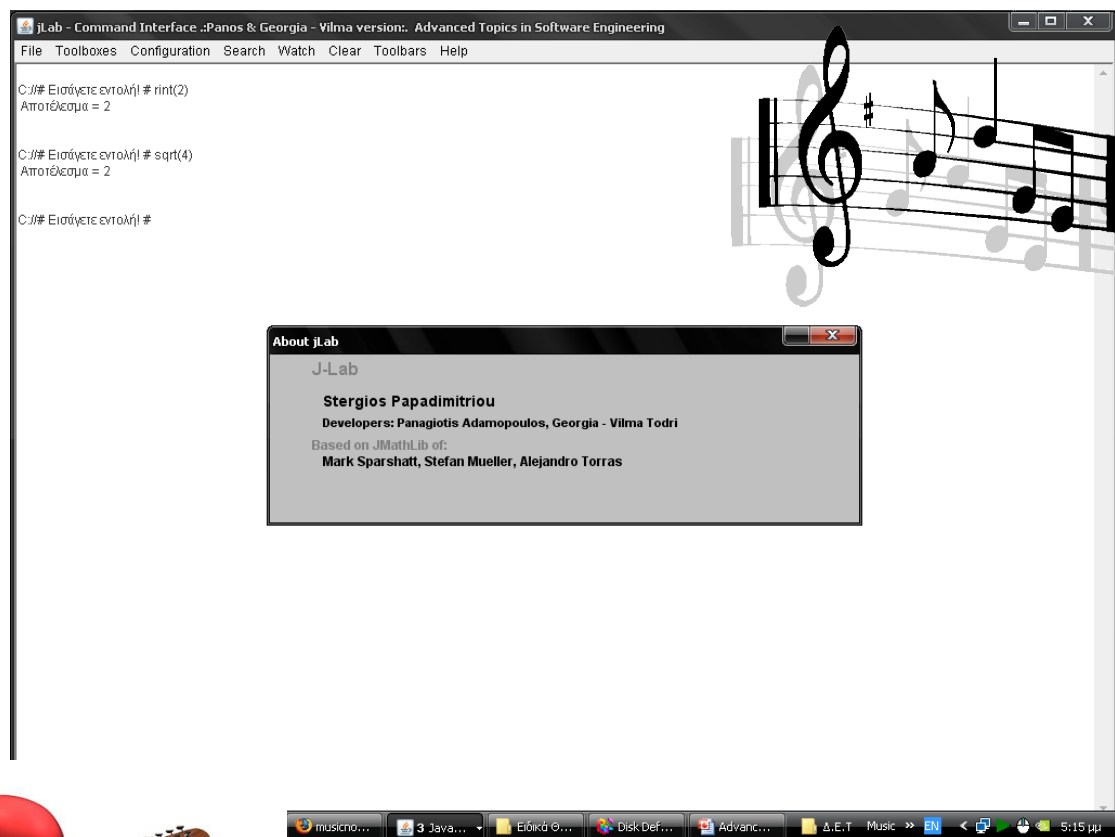
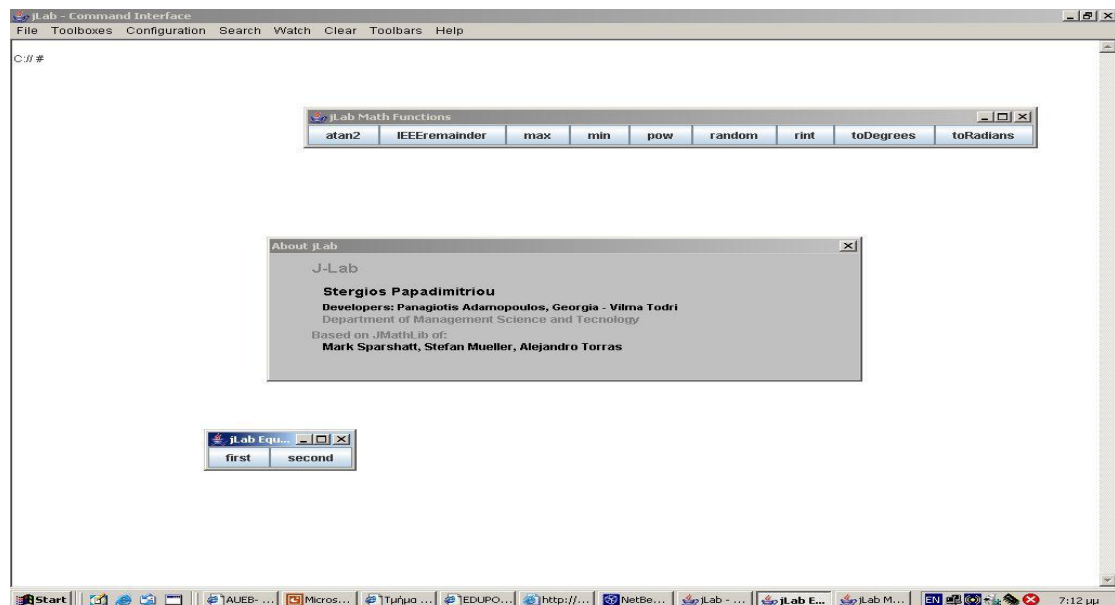
Standard values often appear as editor commands.

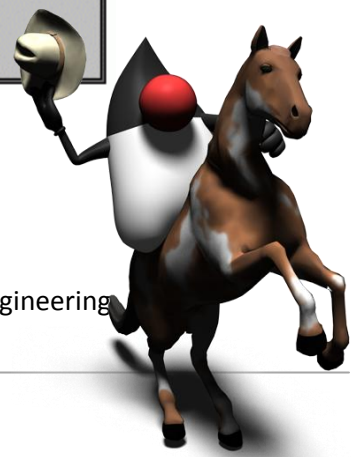
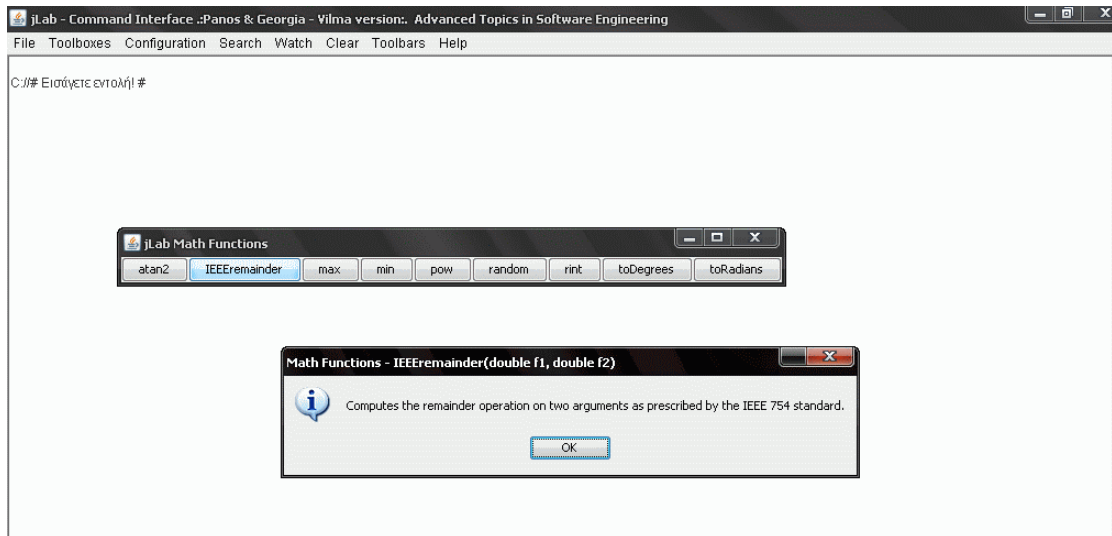
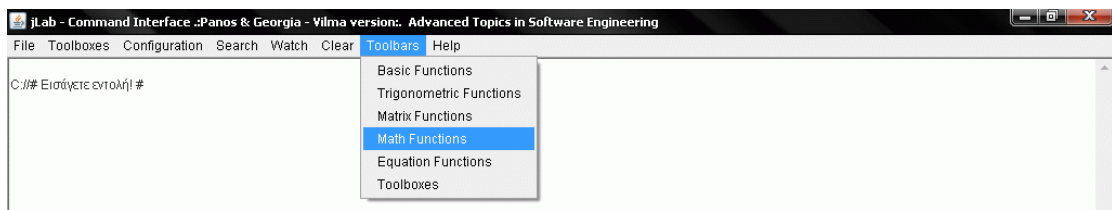
```
/*
@GROUP
matrix
@SYNTAX
answer = XOR(matrix1, matrix2)
@DOC
Returns the boolean xor of the elements of two matrices.
@NOTES
@EXAMPLES
XOR([1,0;0,1], [1,1;0,0]) = [0,1;0,1]
@SEE
<A HREF="and.html">AND</A>
<A HREF="or.html">OR</A>
*/
```

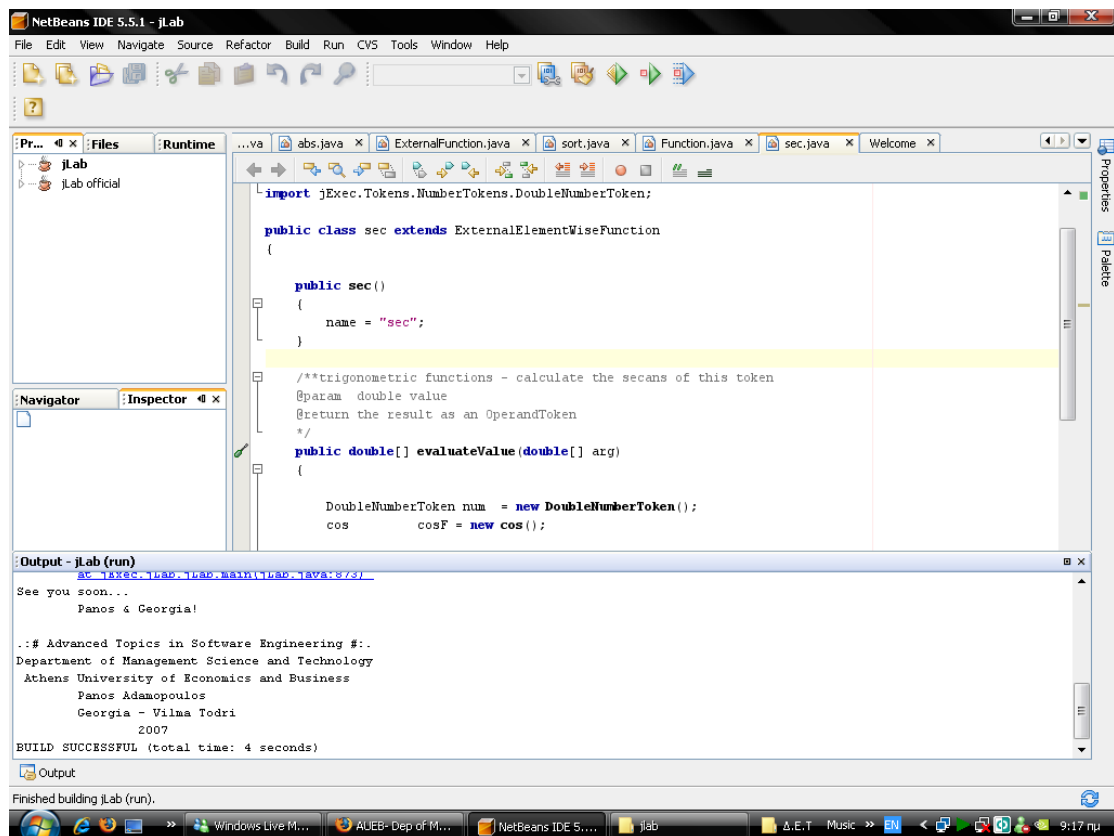
Consistent coding style

Readable structure

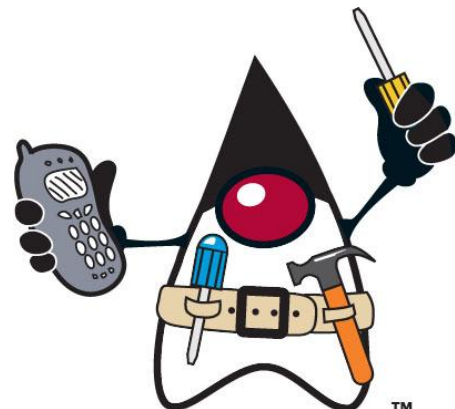
Screenshots of some changes







To produce our changes we used various programs. For example, Netbeans IDE, Eclipse IDE and Textpad.



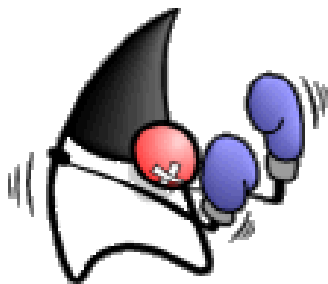
Summary of the changes

- AboutGUIDialog
- jExecObject
- OperandToken
- MathFunction
- NumberToken
- FunctionManager
- Equations
- AddSubOperatorToken
- jExec\Tokens\Expression
- jExec\gui\Console
- jExec.Det.*
- svm_predict
- Errors
- svm_train
- load
- FunctionToken : Class used to represent any functions used in the expression
- .properties



Matrix

- ❖ `exp` : Calculates the exponent of a complex number and takes as arguments the value as an array of double. The result is also an array of double
- ❖ `floor` : Rounds the value of the first operand down to the nearest integer. Takes as argument a double array and return the result as an operand token
- ❖ `ln` : Returns the natural logarithm of value. Takes as argument an array of double and return the result as an array of double too.
- ❖ `log` : Returns the logarithm of value to the base. Takes as argument an array of double and return the result as an array of double too.
- ❖ `round` : Rounds a value to the nearest integer. Takes as argument an array of double and returns the result as an OperandToken
- ❖ `sqrt` : Calculates the sqrt of a complex number. Takes as argument an array of double and return the result as an array of double too.
- ❖ `sum` : Returns the sum of all the elements of a matrix per column. Takes as argument the matrix to sum as an operand.



Math

- ❖ `atan2`: Converts rectangular coordinates (x, y) to polar (r, θ)
- ❖ `IEEEremainder`: Computes the remainder operation on two arguments as prescribed by the IEEE 754 standard.
- ❖ `Max`: Returns the greater of two double values.
- ❖ `Min`: Returns the smaller of two double values.
- ❖ `Pow`: Returns the value of the first argument raised to the power of the second argument.
- ❖ `Random`: Returns a double value with a positive sign, greater than or equal to 0.0 and less than 1.0.
- ❖ `Rint`: Returns the double value that is closest in value to the argument and is equal to a mathematical integer.
- ❖ `toDegrees`: Converts an angle measured in radians to an approximately equivalent angle measured in degrees.
- ❖ `toRadians`: Converts an angle measured in degrees to an approximately equivalent angle measured in radians.



Code examples

```
/**Executes the equation - the code run is based on the index number
    @param operands - the array of parameters
    @return the result of the function as an OperandToken*/
public OperandToken evaluate(Token[] operands)
{
    OperandToken result = null;

    String input = operands.toString();

    OperandToken result1 = new NumberToken(0);

    //execute the equation depending on the index
    switch(index)
    {
    case FIRST:

        double a = ((NumberToken)operands[0]).getValue();

        double b = ((NumberToken)operands[3]).getValue();

        double g = ((NumberToken)operands[6]).getValue();

        g = b-g;

        if (a!=0) {

            double temp_result = - b / a;

            result = new NumberToken(temp_result);

        } else { .....
```



```
/**Calculates the arctangent of a complex number  
  
* @param arg = the value as an array of double  
  
* @return the result as an array of double*/  
  
public OperandToken rint() {  
  
    double[][][] results = new double[sizeY][sizeX][2];  
  
    for (int yy=0; yy<sizeY; yy++) {  
  
        for (int xx=0; xx<sizeX; xx++) {  
  
            results[yy][xx][REAL] = java.lang.Math rint(values[yy][xx][REAL]);  
  
            results[yy][xx][IMAGINARY] =  
java.lang.Math.rint(values[yy][xx][IMAGINARY]);  
  
        }  
  
    }  
  
    return new NumberToken(results);  
  
}
```

Fix code

Before

```
Container box = Box.createHorizontalBox();

box.add(bones);box.add(bzeros);box.add(beye);box.add(binvin);box.add(butriag
);box.add(breshape); box.add(bany);box.add(bfind);box.add(bisEmpty);
```

After

```
// A (AWT) container object that contains boxes (other AWT
components).
```

```
Container box = Box.createHorizontalBox();
```

```
// Appends the specified component to the end of this container.
```

```
box.add(bones);
```

```
box.add(bzeros);
```

```
box.add(beye);
```

```
box.add(binvin);
```

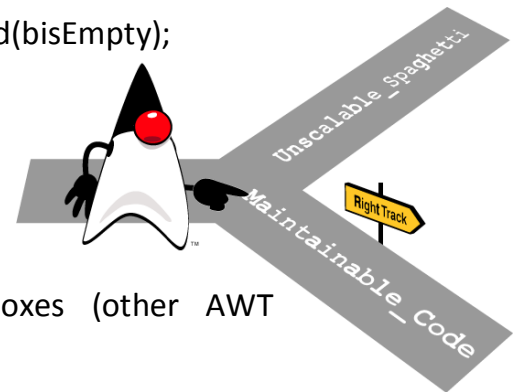
```
box.add(butriag);
```

```
box.add(breshape);
```

```
box.add(bany);
```

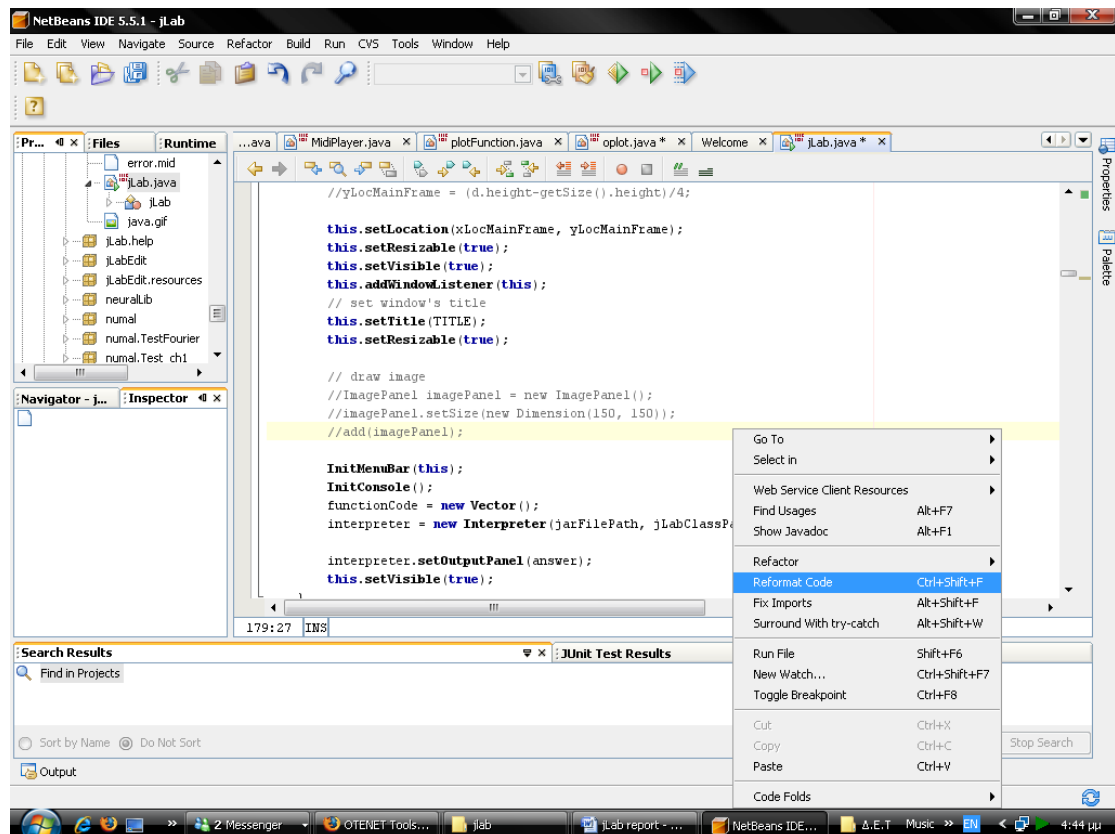
```
box.add(bfind);
```

```
box.add(bisEmpty);
```



Formatting Java Source Code

The IDE automatically formats your code.



But in some cases the output wasn't satisfactory. So, we had to do it manually until we produced an efficient output.

Batch files

REM This script makes everything from scratch.

SET CLASSDIR=..\build\classes

SET SOURCEDIR=jLabSrc

SET JAVAC_OPTS=-classpath %classpath%;dist\jLab.jar;. -d %classdir%

javac %javac_opts% jLab*.java

javac %javac_opts% jLab\Graph*.java

javac %javac_opts% jLab\wavelets*.java

javac %javac_opts% jLab\weka*.java

javac %javac_opts% jLab\neuralLib*.java

javac %javac_opts% jLab\numal*.java

javac %javac_opts% jLab\numal\Test_ch1*.java

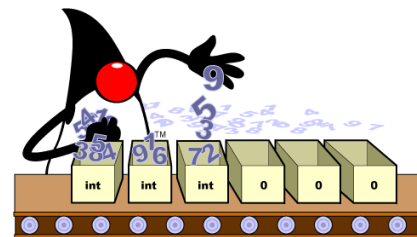
javac %javac_opts% jLab\numal\Test_ch2*.java

javac %javac_opts% jLab\numal\Test_ch3*.java

javac %javac_opts% jLab\numal\Test_ch4*.java

cd classes

java -classpath dist\jLab.jar;



Integration

In order to make our changes and add functionality to jLab project we had to integrate our source code into the rest of the project.

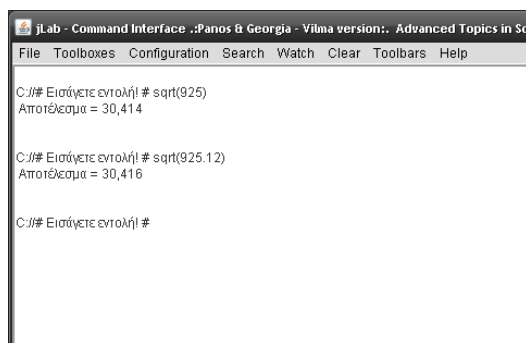
Specifically, as far as matrix functions are concerned, for example, we created and extended the class `ExternalElementWiseFunction.java` which extends `ExternalFunction.java` that is a class that already existed and is considered base class for all external function classes.

Moreover, in many cases we added source code in already existing classes without disturbing the legacy project!

Testing

Scenarios

As far as testing is concerned, we conducted test cases and implement examples of actual use. For example in the package `jExec.Functions.Matrix` we have added the class `sqrt`, it calculates the square root of a complex number. Takes as parameter argument the value as an array of double and return the result as an array of double. It is adapted from "Numerical Recipies in C" (ISBN 0-521-43108-5), William H. Press et al.



```
JavaLab - Command Interface - .Panos & Georgia - Vilma version: Advanced Topics in Software Engineering
File  Toolboxes  Configuration  Search  Watch  Clear  Toolbars  Help

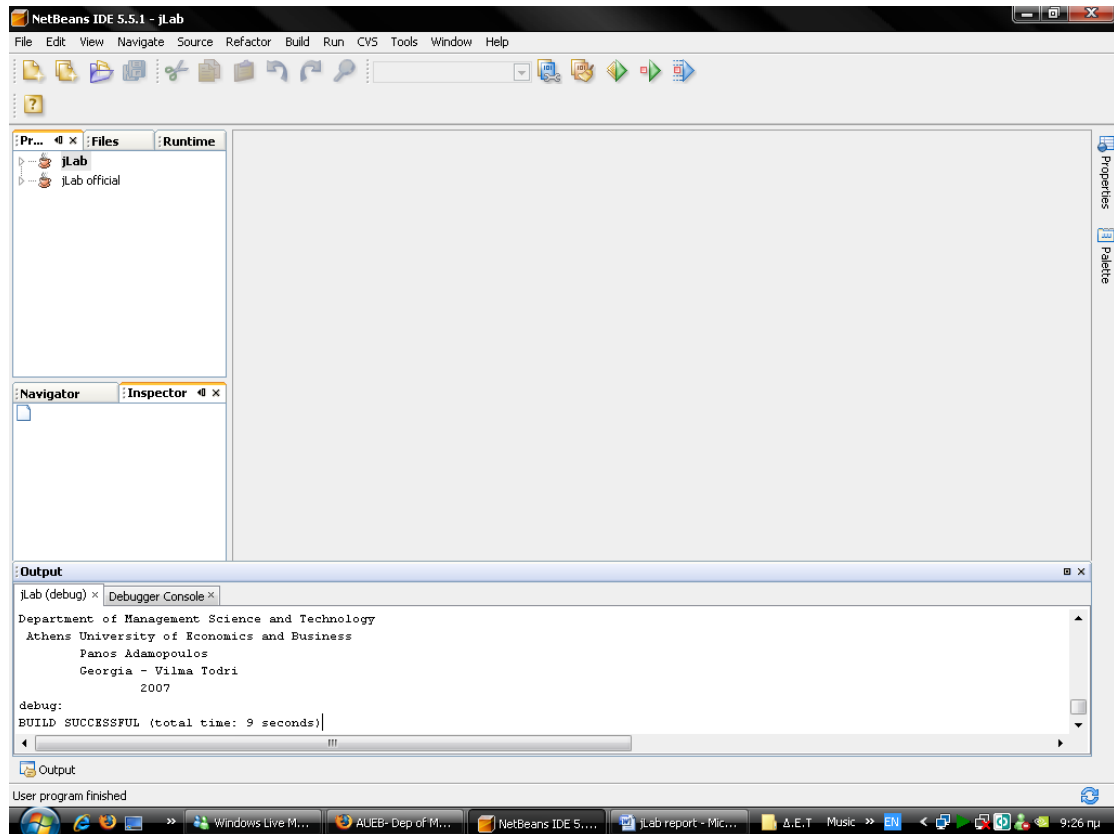
C:\j# Εισάγετε εντολή! # sqrt(925)
Αποτέλεσμα = 30,414

C:\j# Εισάγετε εντολή! # sqrt(925.12)
Αποτέλεσμα = 30,416

C:\j# Εισάγετε εντολή! #
```

As we can see in the following screenshot in the command line of the program we type the command by giving a double such as 925 or 925.12 as argument and the result is what we expect. In this way we have tested the changes and confirm that they work in the way they are supposed to do so.

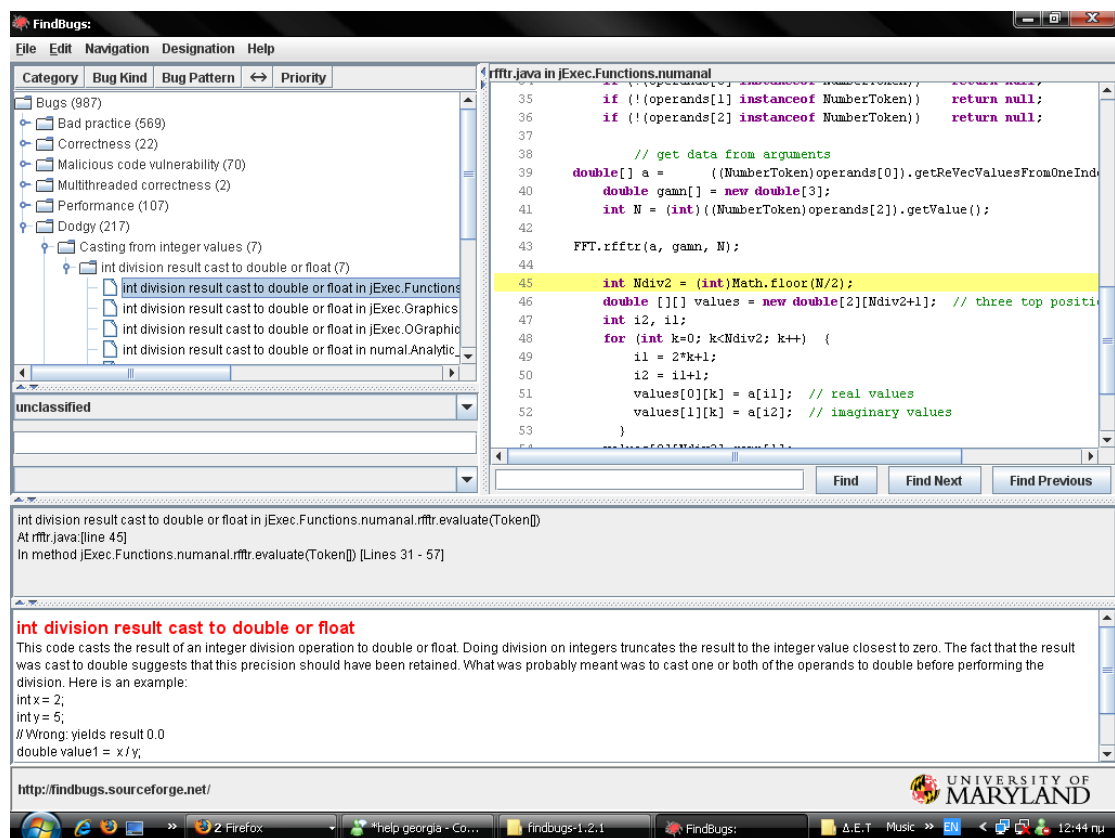
Debug



We debug our project using Netbeans IDE.

Findbugs 1.2.1

It looks for instances of "bug patterns" --- code instances that are likely to be errors.



<?xml version="1.0" encoding="UTF-8"?>

<BugCollection version="1.2.1" sequence="0" timestamp="1182430746651"
analysisTimestamp="1182431092094" release="">

<Project filename="<<unnamed project>>" projectName="jLab">



```
<Jar>W:\jlab\build\classes</Jar>

<SrcDir>W:\jlab\jLabSrc</SrcDir>

<SuppressionFilter>

  <LastVersion value="-1" relOp="NEQ"/>

</SuppressionFilter>

</Project>

<BugInstance type="DLS_DEAD_LOCAL_STORE" priority="2" abbrev="DLS"
category="STYLE">

  <UserAnnotation/>

  <Class classname="Graph.ScatterPlot">

    <SourceLine classname="Graph.ScatterPlot" sourcefile="ScatterPlot.java"
sourcepath="Graph/ScatterPlot.java"/>

  </Class>

  <Method      classname="Graph.ScatterPlot"      name="mouseReleased"
signature="(Ljava/awt/event/MouseEvent;)V" isStatic="false">

    <SourceLine  classname="Graph.ScatterPlot"  start="479"  end="483"
startBytecode="0"      endBytecode="24"      sourcefile="ScatterPlot.java"
sourcepath="Graph/ScatterPlot.java"/>

  </Method>

  <LocalVariable      name="x"      register="2"      pc="5"
role="LOCAL_VARIABLE_NAMED"/>
```

```
<SourceLine    classname="Graph.ScatterPlot"    start="479"    end="479"  
startBytecode="4"        endBytecode="4"        sourcefile="ScatterPlot.java"  
sourcepath="Graph/ScatterPlot.java"/>  
  
</BugInstance> .....
```

ckjm-1.7

```

C:\WINDOWS\system32\cmd.exe
C:\>java -jar ckjm-1.7.jar C:\build\classes\jExec\Functions\Matrix\*.class
jExec.Functions.Matrix.xor 2 1 0 4 5 1 0 2
jExec.Functions.Matrix.max 2 1 0 5 9 1 0 2
jExec.Functions.Matrix.inf 2 1 0 4 7 1 0 2
jExec.Functions.Matrix.magic 3 1 0 4 8 3 0 3
jExec.Functions.Matrix.min 2 1 0 5 9 1 0 2
jExec.Functions.Matrix.cumsum 2 1 0 4 10 1 0 2
jExec.Functions.Matrix.fliplr 2 1 0 4 6 1 0 2
jExec.Functions.Matrix.ltriag 4 1 0 5 10 6 0 2
jExec.Functions.Matrix.zeros 2 1 0 4 7 1 0 2
jExec.Functions.Matrix.prod 2 1 0 4 5 1 0 2
jExec.Functions.Matrix.all 2 1 0 5 9 1 0 2
jExec.Functions.Matrix.adj 4 1 0 8 17 6 0 2
jExec.Functions.Matrix.elemAt 2 1 0 6 12 1 0 2
jExec.Functions.Matrix.lu 2 1 0 6 12 1 0 2
jExec.Functions.Matrix.sort 2 1 0 5 16 1 0 2
jExec.Functions.Matrix.SubMatrix 3 1 0 10 25 0 0 3
jExec.Functions.Matrix.log 2 1 0 1 7 1 0 2
jExec.Functions.Matrix.reshape 2 1 0 4 5 1 0 2
jExec.Functions.Matrix.or 2 1 0 4 5 1 0 2
jExec.Functions.Matrix.isfinite 2 1 0 4 9 1 0 2
jExec.Functions.Matrix.diag 2 1 0 4 15 1 0 2
jExec.Functions.Matrix.nnz 2 1 0 5 11 1 0 2
jExec.Functions.Matrix.inv 2 1 0 8 15 1 1 2
jExec.Functions.Matrix.isreal 2 1 0 4 8 1 0 2
jExec.Functions.Matrix.ln 2 1 0 1 7 1 0 2
jExec.Functions.Matrix.nan 2 1 0 4 7 1 0 2
jExec.Functions.Matrix.det 4 1 0 5 12 6 0 2
jExec.Functions.Matrix.pow2 2 1 0 4 6 1 0 2
jExec.Functions.Matrix.isEmpty 2 1 0 5 6 1 0 2
jExec.Functions.Matrix.eig 2 1 0 6 15 1 0 2
jExec.Functions.Matrix.svd 2 1 0 6 14 1 0 2
jExec.Functions.Matrix.not 2 1 0 4 5 1 0 2
jExec.Functions.Matrix.utriag 4 1 0 5 10 6 0 2
jExec.Functions.Matrix.isimaginary 2 1 0 4 7 1 0 2
jExec.Functions.Matrix.exp 2 1 0 1 6 1 0 2
jExec.Functions.Matrix.fliplr 2 1 0 4 6 1 0 2
jExec.Functions.Matrix.floor 2 1 0 1 4 1 0 2
jExec.Functions.Matrix chol 2 1 0 5 9 1 0 2
jExec.Functions.Matrix.and 2 1 0 4 7 1 0 2
jExec.Functions.Matrix.SinEq 2 1 0 7 8 1 0 2
jExec.Functions.Matrix.ones 2 1 0 4 5 1 0 2
jExec.Functions.Matrix.abs 2 1 0 1 5 1 0 2
jExec.Functions.Matrix.sqrt 2 1 0 1 5 1 0 2
jExec.Functions.Matrix.InverseMatrix 2 1 0 8 15 1 0 2
jExec.Functions.Matrix.sum 2 1 0 7 18 1 0 2
jExec.Functions.Matrix.round 2 1 0 1 4 1 0 2
jExec.Functions.Matrix.renAt 2 1 0 4 5 1 0 2
jExec.Functions.Matrix.mean 2 1 0 5 11 1 0 2
jExec.Functions.Matrix.numel 2 1 0 5 8 1 0 2
jExec.Functions.Matrix.find 2 1 0 4 7 1 0 2
jExec.Functions.Matrix.qr 2 1 0 6 11 1 0 2
jExec.Functions.Matrix.any 2 1 0 4 7 1 0 2
jExec.Functions.Matrix.SubAssign 3 1 0 10 29 0 0 3
jExec.Functions.Matrix.isnan 2 1 0 4 9 1 0 2
jExec.Functions.Matrix.ceil 2 1 0 1 4 1 0 2
jExec.Functions.Matrix.eye 2 1 0 4 7 1 0 2
jExec.Functions.Matrix.cumprod 2 1 0 4 10 1 0 2
  
```

The program *ckjm* calculates Chidamber and Kemerer object-oriented metrics by processing the bytecode of compiled Java files. The program calculates for each class the following six metrics, and displays them on its standard output, following the class's name:

- WMC: Weighted methods per class
- DIT: Depth of Inheritance Tree
- NOC: Number of Children
- CBO: Coupling between object classes
- RFC: Response for a Class
- LCOM: Lack of cohesion in methods
- Ca: Afferent coupling (not a C&K metric)
- NPM: Number of Public Methods for a class (not a C&K metric)

C:\>java -jar ckjm-1.7.jar C:\build\classes\jExec\Functions\Matrix*.class

Class	WM C	DI T	NO C	CB O	RF C	LCO M	C e	NP M
jExec.Functions.Matrix.xor	2	1	0	4	5	1	0	2
jExec.Functions.Matrix.max	2	1	0	5	9	1	0	2
jExec.Functions.Matrix.inf	2	1	0	4	7	1	0	2
jExec.Functions.Matrix.magic	3	1	0	4	8	3	0	3
jExec.Functions.Matrix.min	2	1	0	5	9	1	0	2
jExec.Functions.Matrix.cumsum	2	1	0	4	10	1	0	2
jExec.Functions.Matrix.fliplr	2	1	0	4	6	1	0	2
jExec.Functions.Matrix.ltriag	4	1	0	5	10	6	0	2
jExec.Functions.Matrix.zeros	2	1	0	4	7	1	0	2
jExec.Functions.Matrix.prod	2	1	0	4	5	1	0	2
jExec.Functions.Matrix.all	2	1	0	5	9	1	0	2
jExec.Functions.Matrix.adj	4	1	0	8	17	6	0	2
jExec.Functions.Matrix.elemAt	2	1	0	6	12	1	0	2
jExec.Functions.Matrix.lu	2	1	0	6	12	1	0	2
jExec.Functions.Matrix.sort	2	1	0	5	16	1	0	2
jExec.Functions.Matrix.SubMatrix	3	1	0	10	25	0	0	3
jExec.Functions.Matrix.log	2	1	0	1	7	1	0	2
jExec.Functions.Matrix.reshape	2	1	0	4	5	1	0	2
jExec.Functions.Matrix.or	2	1	0	4	5	1	0	2
jExec.Functions.Matrix.isfinite	2	1	0	4	9	1	0	2

jExec.Functions.Matrix.diag	2	1	0	4	15	1	0	2
jExec.Functions.Matrix.nnz	2	1	0	5	11	1	0	2
jExec.Functions.Matrix.inv	2	1	0	8	15	1	1	2
jExec.Functions.Matrix.isreal	2	1	0	4	8	1	0	2
jExec.Functions.Matrix.ln	2	1	0	1	7	1	0	2
jExec.Functions.Matrix.nan	2	1	0	4	7	1	0	2
jExec.Functions.Matrix.det	4	1	0	5	12	6	0	2
jExec.Functions.Matrix.pow2	2	1	0	4	6	1	0	2
jExec.Functions.Matrix.isEmpty	2	1	0	5	6	1	0	2
jExec.Functions.Matrix.eig	2	1	0	6	15	1	0	2
jExec.Functions.Matrix.svd	2	1	0	6	14	1	0	2
jExec.Functions.Matrix.not	2	1	0	4	5	1	0	2
jExec.Functions.Matrix.utriag	4	1	0	5	10	6	0	2
jExec.Functions.Matrix.isimaginary	2	1	0	4	7	1	0	2
jExec.Functions.Matrix.exp	2	1	0	1	6	1	0	2
jExec.Functions.Matrix.flipud	2	1	0	4	6	1	0	2
jExec.Functions.Matrix.floor	2	1	0	1	4	1	0	2
jExec.Functions.Matrix.chol	2	1	0	5	9	1	0	2
jExec.Functions.Matrix.and	2	1	0	4	7	1	0	2
jExec.Functions.Matrix.SimEq	2	1	0	7	8	1	0	2
jExec.Functions.Matrix.ones	2	1	0	4	5	1	0	2
jExec.Functions.Matrix.abs	2	1	0	1	5	1	0	2
jExec.Functions.Matrix.sqrt	2	1	0	1	5	1	0	2
jExec.Functions.Matrix.Inverse Matrix	2	1	0	8	15	1	0	2
jExec.Functions.Matrix.sum	2	1	0	7	18	1	0	2

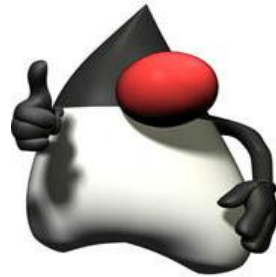
jExec.Functions.Matrix.round	2	1	0	1	4	1	0	2
jExec.Functions.Matrix.remat	2	1	0	4	5	1	0	2
jExec.Functions.Matrix.mean	2	1	0	5	11	1	0	2
jExec.Functions.Matrix.numel	2	1	0	5	8	1	0	2
jExec.Functions.Matrix.find	2	1	0	4	7	1	0	2
jExec.Functions.Matrix.qr	2	1	0	6	11	1	0	2
jExec.Functions.Matrix.any	2	1	0	4	7	1	0	2
jExec.Functions.Matrix.SubAssi gn	3	1	0	10	29	0	0	3
jExec.Functions.Matrix.isnan	2	1	0	4	9	1	0	2
jExec.Functions.Matrix.ceil	2	1	0	1	4	1	0	2
jExec.Functions.Matrix.eye	2	1	0	4	7	1	0	2
jExec.Functions.Matrix.cumpro d	2	1	0	4	10	1	0	2
jExec.Functions.Matrix.isinf	2	1	0	4	8	1	0	2

Coordination with the development team – Mails

Από: "root" <sterg@philippos.teikav.edu.gr>

Μπραβο για tin grigori prosarmogi sas ston kodika!!

> Fisiko einai na xathite ston kodika giati einai poliplokos
> kai xriazetai prosektiko diavasma.
>
> Stergios



Documentation

In general terms, **documentation** is any communicable material (such as text, video, audio, etc., or combinations thereof) used to explain some attributes of an object, system or procedure.

Documentation of Source Code

One of the most important forms of documentation for computer software is one that ordinary users rarely, if ever, see. It is the *comments* that are included in the *source code* of programs. Source code is the version of software (usually an application program or an operating system) as it is originally *written* (i.e., typed into a computer) by a human in plain text (i.e., human readable alphanumeric characters) in a programming language.

Comments are separated from the source code by special markers and do not affect its operation. They are statements by programmers explaining their code to other programmers who may work on the same programs and to remind themselves of what they did or what remains to be done. The comments ideally include the reasons that each section of code is written a particular way and what it is intended to do.

[<http://www.linfo.org/documentation.html>]

We used Java `/**` comments that are read by javadoc.

For example:

```
/**Calculates the logarithm of a complex number
```

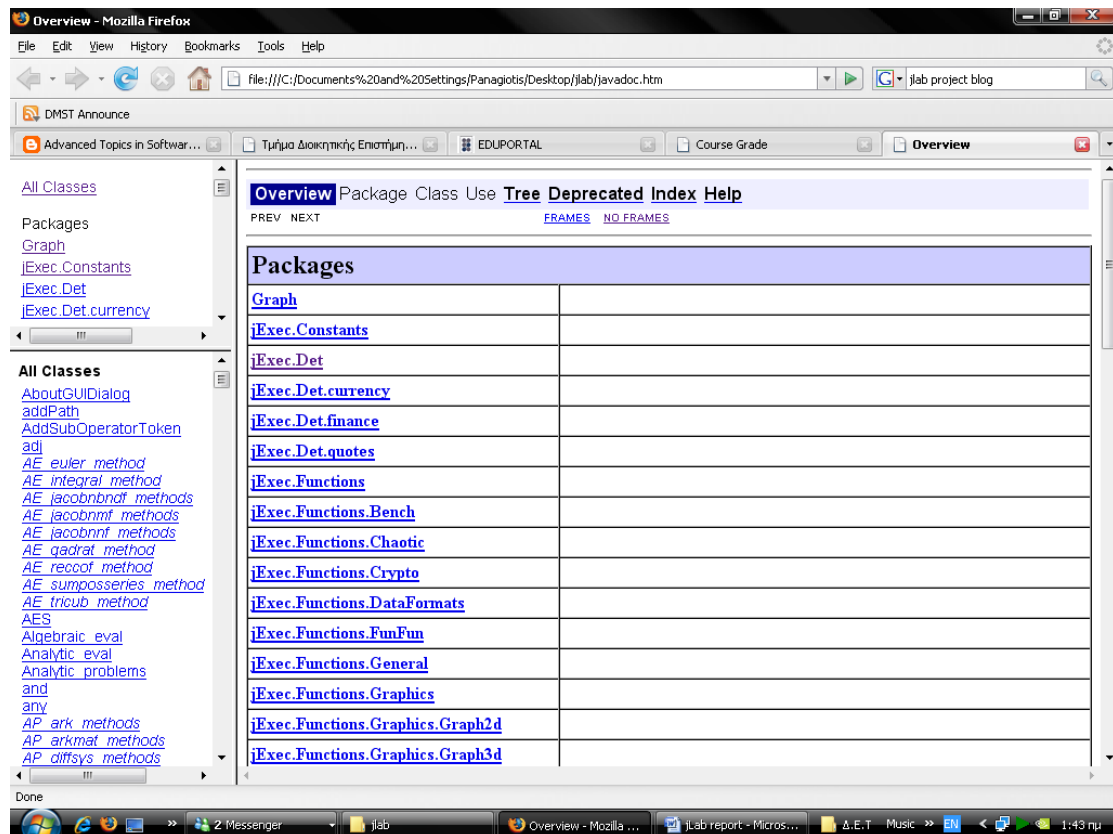
```
    @param arg = the value as an array of double
```

```
    @return the result as an array of double*/
```

```
public double[] evaluateValue(double[] arg)
```

```
{ .....
```

So, we used a doclet to generate the documentation. The standard doclet generates HTML and is built into the Javadoc tool.



Hopefully, we didn't reached any XXX (means something is probably wrong here) TODO (marks areas of further work) or FIXME (marks areas of further enhancement) comments.



Blog

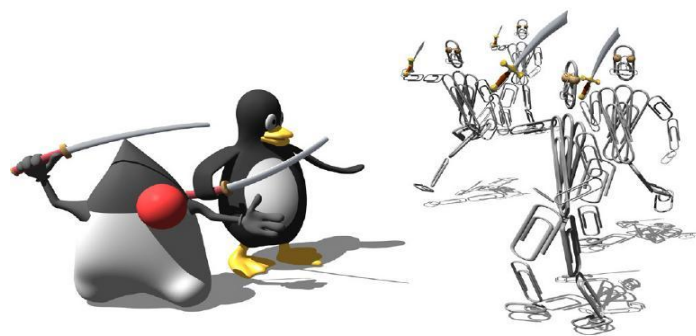
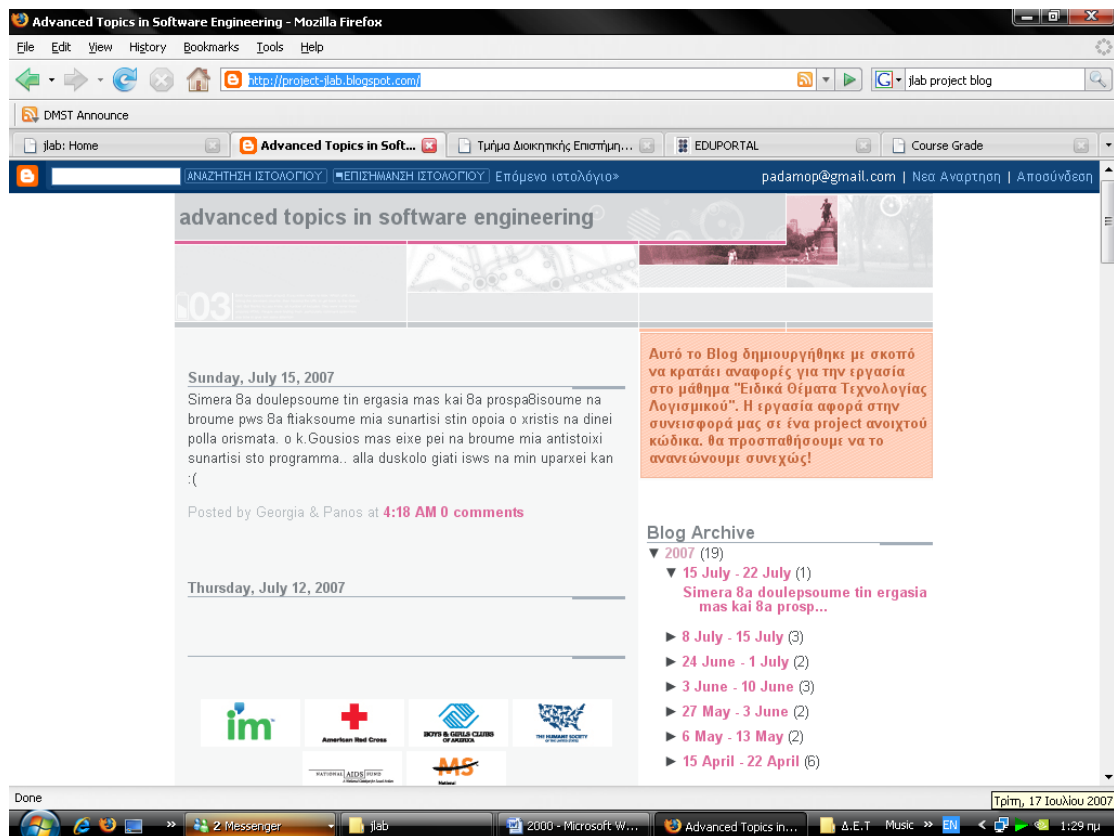
We tried to keep our blog up-to-date. So, we did many posts which explain our contribution in jLab project.

Our blog is written mostly in Greek.

The post that is included below is our first post in our blog.

“Αυτό το Blog δημιουργήθηκε με σκοπό να κρατάει αναφορές για την εργασία στο μάθημα «Ειδικά Θέματα Τεχνολογίας Λογισμικού». Η εργασία αφορά στην συνεισφορά μας σε ένα project ανοιχτού κώδικα. Θα προσπαθήσουμε να το ανανεώνουμε συνεχώς!”

URL: <http://project-jlab.blogspot.com/>



Working Team

Panagiotis I. Adamopoulos

Student in the department of management science and technology (DMST) at
the Athens University of Economics (AUEB)

www.dmst.aueb.gr

Number of Registration: 8040000

padam@dmst.aueb.gr



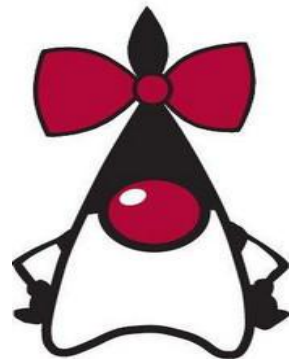
Vilma – Georgia N. Todri

Student in the department of management science and technology (DMST) at
the Athens University of Economics (AUEB)

www.dmst.aueb.gr

Number of Registration: 8040140

vtodri@dmst.aueb.gr



Department of Management Science and Technology

Athens University of Economics and Business

Advanced Topics in Software Engineering

[Department of Management Science and Technology](#)

[Athens University of Economics and Business](#)

Profesor: Diomidis Spinellis: dds@aueb.gr

E-mail: dds @aueb.gr

Office phone: +30 2108203682

Office address: Ydras 28, 5th floor

Postal address: Patision 76, GR-104 34 Athens, Greece

Web site: <http://www.dmst.aueb.gr/dds>

Lab assistant: Giorgos Gousios: gousiosg@aueb.gr

E-mail: gousiosg @aueb.gr

Office phone: +30 2108203370

Office address: Athens University of Economics and Business Main Building,

Derigny ct, 3rd fl

Postal address: 76 Patission str, 104 34, Athens, Greece

Web site: <http://istlab.dmst.aueb.gr/~george>

