A Web-Based System to Support Java Programming Learning

António Mendes, Valentino Ivanov, Maria Marcelino

Abstract:. *The difficulties felt by students and teachers in programming learning in higher education led to the development of ELJT, a tool that has as main objectives to support the work of both the student and the teacher of programming classes giving the former an easy possibility for extra training. In ELJT students can submit solutions to problems proposed by teachers in the form of Java computer programs. They can also test the solutions and get feedback from the system in several ways. Teachers besides providing extra problems for students can monitor their evolution.*

Key words: *on-line training, educational technology, programming teaching and learning.*

INTRODUCTION

Many students experience lots of difficulties in initial programming learning. To become a good programmer it is necessary to practice a lot, solving different types of exercises, in order to develop important competencies like abstraction, generalization, and transfer. It is important that students work individually and in groups, creating solutions to problems, correcting and refining them, until a correct solution is achieved. This work is important not only to develop programming capacities, but also to increase students' confidence in their own abilities. To many students the usual classroom time is not enough and they need to practice on their own outside classroom. Doing this without teacher supervision is difficult to many of them. However, detailed teacher supervision is not feasible when classes have many students, and teachers have difficulties to know each student characteristics and main difficulties. Considering this situation, we believe it is important to develop systems that can help students to work autonomously and teachers to have a better knowledge of each student particular difficulties.

Some attempts have been made during the years to cope with this problem. Among them we can mention the use of software tools specially developed to help programming learning. We believe that animation based simulation tools that can allow students to visualize their own algorithms and programs execution are particularly useful [1-4]. But we could refer many others, like programmable micro-worlds [5] or automatic assignment of student tasks [6] and automatic assessment of programs developed by students [7-9].

The tool presented in this paper tries to contribute also to the overall objective of supporting programming learning by novice students. The idea is to complement the utilization of other systems and increase student autonomy and teacher information.

E-LEARNING JAVA TRAINER

ELJT (E-Learning Java Trainer) is a web-based environment that can present programming problems to students and allows them to submit their solutions to these problems as precompiled Java programs. ELJT analyses the student solution and gives some feedback about its correctness. The tool also registers student actions, so that it can give useful information to both students and teachers.

We believe that a tool such as ELJT can facilitate both teacher and student tasks. The student can have access to a higher number of problems to solve, challenging her/his competences and creating conditions to a deeper involvement in programming activities. It is important that students have conscience of their own limitations and the environment feedback can help in that objective. The environment can also be used to create competition between students or groups of students, improving their motivation to the course.

ELJT has two modes:

- a student mode and
- a teacher mode.

To use the environment, both students and teachers must be registered and identify themselves through a username and password.

In Figure 1 we can see ELJT main page in student mode:

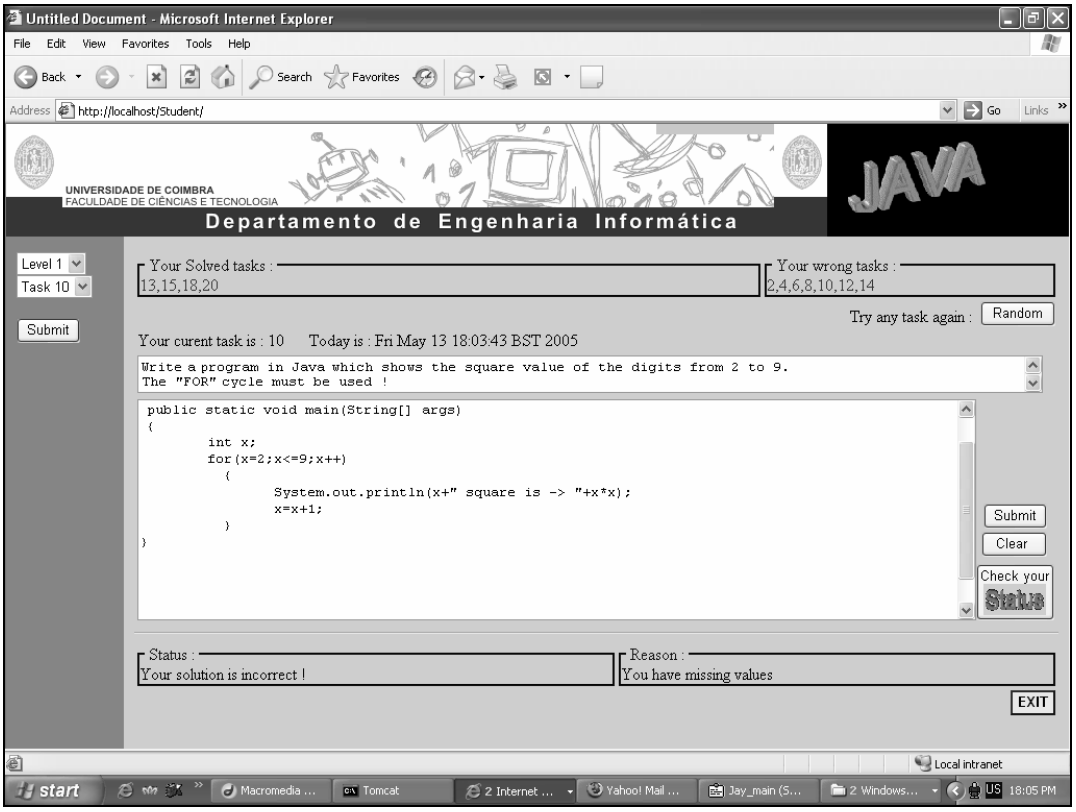

Figure 1 – ELJT main page in student mode.

The page is organized in three areas. The upper part just gives some information about the student past utilization, namely in terms of current difficulty level problems successfully solved and those tried but not solved.

In the left frame students have the possibility to choose the next task, or problem, they want to solve. These tasks are organized by level of difficultly. There are currently three different levels of difficulty in the system (easy, regular and difficult). Every level can contain several tasks. Students can choose at any time which level they want to work in and the task that they want to solve. The student can also ask ELJT to propose him/her a random problem of a certain level or just ask for a problem, letting the environment choose the level based on the student previous activity. It is also possible to ask to solve a problem the student tried before unsuccessfully.

When a problem is selected the environment presents its description and waits for the student to submit a solution to it. So, the student main task is to create and submit a JAVA written solution to the proposed problem. Solution submission can be made simply by pasting the code into the appropriate text area or indicating the file that contains the solution code.

When a student completes and submits a solution to the current task the environment tests it in terms of syntactic correctness using a JAVA compiler. If it is syntactically correct the system analyses if the solution really solves the task. To do this it

runs the solution using test data and comparing the results obtained with expected results. Both test data and expected results are given by teachers when they specify tasks in ELJT teacher mode. If the proposed solution fails to solve the problem considering all the data sets available the environment informs the student and makes a code analysis trying to find failure reasons. This is done based on information given by the teacher when creating the task, for example the teacher may say that the correct solution has two repetitions, and also information about common novice errors, such as infinite loops, variables not properly initialized and so on. The conclusions, if any, are given to the student, so that she/he can correct the proposed solution.

The other ELJT functioning mode, the teacher mode, is shown in Figure 2. In this mode a teacher can insert, delete and update problems and monitor her/his students or the tasks evolution.

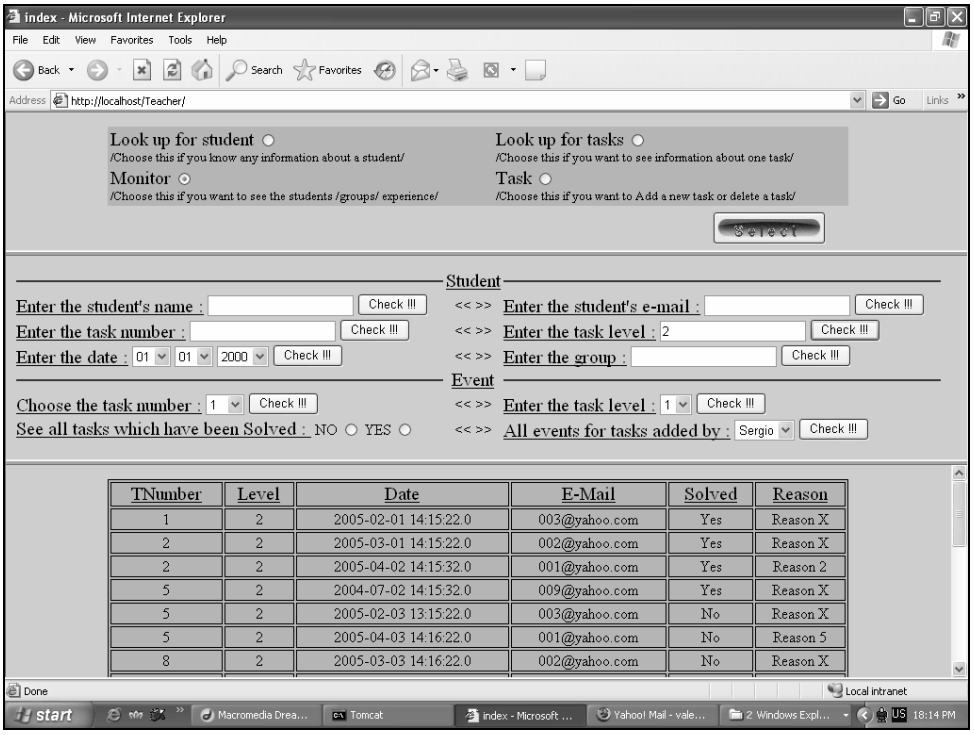

Figure 2 – Main page of ELJT in teacher mode.

In the top part the teacher has four available choices:

- "**Look up for student**" allow the access to information about a particular student.
- "**Look up for tasks**" permits the access to information about a specific task.
- "**Task**" gives access to the management of tasks, namely:
  - add a new task,
  - delete a task or
  - update an existing task.
- The "**Monitor**" option allows the teacher to make more complex queries to the system. Those are specified in the middle part of the page. The teacher can center her/his attention on information about student performance or about task solutions. Inside these two broad groups, several queries can be made and the information that will appear in the bottom frame depends on what the teacher is interested in. For example, it is possible to concentrate on the performance of a particular student, the tasks she/he solved, the errors most commonly made and the tasks the student wasn't able solve. It is also possible to know information about a task, such as how many students tried it and how many solved

it, the most common errors on that task and so on. Figure 2 shows a possible situation where the teacher wants to know all the times that level 2 problems were unsuccessfully tried. The results of a particular query are shown in the page bottom part.

Each task has events associated with it. Events describe attempts to solve a particular problem. It is possible for the teacher to obtain information about the number of times a particular task or group of tasks was solved or tried unsuccessfully. Figure e shows an example for level 3 problems.

TNumber	Level	Proposed	Solved	NotSolved
3	3	100	60	40
6	3	100	10	90
9	3	80	10	70
12	3	70	20	50
15	3	40	5	45
18	3	50	15	35
21	3	50	5	45

Figure 3 – task events.

We can observe the task numbers, how many times each was tried, solved and attempted (incorrectly solved). For instance, task number 21 was tried 50 times, but only 5 students succeeded until the moment. The teacher can identify problematic areas that may need an extra clarification.

The technologies used to develop ELJT are JSP, JSTL, EJB and the Java programming language. The application also uses a MySQL database server (v4.1.11) and a Tomcat 5.5.4 as a Web server. The database is used to store all the information, namely student and teacher accounts, proposed tasks, solved tasks, etc.

CONCLUSIONS AND FUTURE WORK

In this paper we described a web-based system to support programming students' training using Java as the programming language. The system works in two modes, one for teachers, to provide problems and monitor students' work, and the other for students to automatically test their solutions to selected or proposed problems in the form of computer programs. From the point of view of the students such a tool can offer:

- an auto-testing mechanism that can enhance conscience about their own learning,
- feedback about errors made,
- challenge towards themselves when trying to reach higher proficiency levels
- competition among the students of the class
- etc.

From the point of view of the teacher it can offer a simple way to:

- provide more problems to students,
- know her/his students evolution,
- identify students' difficulties,
- improve student support,
- etc.

At present ELJT is in an advanced stage of development. Our aim is to finish a first version soon and perform some preliminary tests in order to use it with students in the next academic year. At the same time we will conduct a first evaluation study where we will use common evaluation techniques, like usability interface testing, questionnaire passing and user interviewing.

Future enhancements of the tool are already being planned and include:

- The extension to accommodate other languages, like C or C++.
- The integration with algorithm and program animation tools that we have developed, namely SICAS and OOP-anim [2, 4].
- Assisted problem resolution.
- The development of a collaborative version.

REFERENCES

[1] Naps, T. L., Eagan and J. R., Norton, L. L., JHAVÉ – An Environment to Actively Engage Students in Web-based Algorithm Visualizations, in Proceedings of the 31st SIGCSE Technical Symposium on Computer Science Education, 2000.

[2] Marcelino, M., Gomes, A., Dimitrov, N. and Mendes, A. Using a computer-based interactive system for the development of basic algorithmic and programming skills, in Proceedings of International Conference on Computer Systems and Technologies (CompSysTech'2004), Sofia, Bulgaria, 2004.

[3] Costelloe, E., The Use of a Software Enabled Scaffolding Environment to Aid Novice Programmers, MSc Thesis, University of Dublin, 2004.

[4] Esteves, M. and Mendes, A., A Simulation Tool to Help Learning of Object Oriented Programming Basics, in Proceedings of the 34th ASEE/IEEE Frontiers in Education Conference, Savannah, Georgia, USA, 2004.

[5] Becker, B. W., Teaching CS1 with Karel the Robot in Java, in Proceedings of the 32nd SIGCSE Technical Symposium on Computer Science Education, 2001.

[6] Pardo, A., A Multi-Agent Platform for Automatic Assignment Management, in Proceedings of Innovation and Technology in Computer Science Education – ITiCSE02, Aarhus, Denmark, 2002.

[7] Wu, S., Tsai, S. and Yang, P., JAVALAB – A Java Tutorial and Programming Laboratory System, in Proceedings of Exploring Innovation in Education and Research, Taiwan, 2005.

[8] Foubister, S. P., Michaelson, G. J. and Tomes, N., Automatic assessment of elementary standard programs using Ceilidh, Journal of Computer Assisted Learning, vol. 13, pp. 99-108, 1997.

[9] Korhonen, A., Malmi, L., Nikander, J. and Tenhunen, P., Interaction and Feedback in Automatically Assessed Algorithm Simulation Exercises, Journal of Information Technology Education, vol. 2, pp. 241-255, 2003.

ABOUT THE AUTHORS

Assist. Prof. António Mendes, PhD, Department of Informatics Engineering, University of Coimbra, Phone: +351 239 790000, E-mail: toze@dei.uc.pt.

Valentino Ivanov, BSc, Department of Computer Systems, University of Rousse, E-mail: valentino82bg@yahoo.com.

Assist. Prof. Maria Marcelino, PhD, Department of Informatics Engineering, University of Coimbra, Phone: +351 239 790000, E-mail: zemar@dei.uc.pt.