FRIMAN

Erik Parso, Dominik Suroviak, Jaroslav Ondrejak, Marek Dragula,
Matej Jesensky, Patrik Sensel, Peter Sedlacek, Monika Vaclavkova

***Abstract*—**The learning process of information management on Faculty of management science and informatics is quite long and hard procedure. To be a successful manager on a field of informatics that is able to lead team of developers creating some sort of information system requires to understand at least basics of programming and algorithmic thinking. Therefore, there are some subjects focusing on these abilities and skills. To make it easier to get base knowledge of programming there was founded a team to develop an application allowing students to learn faster.

***Keywords*—**information management, unified modelling language, object oriented programming, FRIMAN, integrated development environment, flowchart diagram

I. INTRODUCTION

The purpose of information management is to:

- *design, develop, manage, and use information with insight and innovation*
- *support decision making and create value for individuals, organizations, communities, and societies [1]*

The Faculty of management Science and Informatics helps students to develop a deep understanding of the users of information and the organizational and social goals information management serves. As a result, they are equipped to use information both as a competitive tool and a means to create positive organizational change. Nowadays, these people should have a good understanding of information technologies. These technologies are used by many people to maximize efficiency of development process. Except for the basic knowledge of operation systems and office applications, these students should have also the basic know-how about the design, development and maintenance of information systems and some knowledge about the software engineering. Therefore, students studying information management at our faculty have to pass through few subjects oriented on software development. Many of them have little or none experience with programming and their way of thinking is different. It's hard for them to understand the principles of program thinking. Many software companies use an object oriented paradigm in development process. That is the main reason for students to learn to program application in Java language which is one of the most used object oriented programming language nowadays. The problem is that it is really hard for them to understand how to write a code using this language and how the program is executed. Flowchart is one of the ways to teach them the principles of algorithm thinking. But, this approach has one big disadvantage: the relation between flowchart and code is not obvious for some of them. It would be useful to provide students tool allowing them to create simple programs without having the knowledge of programming language syntax.

II. COMPARISON OF EXISTING DEVELOPMENT ENVIRONMENTS

At the beginning of the project one year ago, there was an idea to create a Java library providing simple API for application development. The purpose of this API was to make some operations like writing or reading text from console or file simpler than Java API is providing now. However, after analyses we find out that this approach is not suitable. One of our project leaders

D. Suroviak, Faculty of management science and informatics, University of Zilina, (e-mail:suroviakdominik@gmail.com)
M. Jesensky, Faculty of management science and informatics, University of Zilina, (e-mail:m.jesensky@centrum.sk)
M. Dragula, Faculty of management science and informatics, University of Zilina, (e-mail: marek.dragula@gmail.com)
J. Ondrejak, Faculty of management science and informatics, University of Zilina, (e-mail: ondrejakjaro@gmail.com)
P. Sensel, Faculty of management science and informatics, University of Zilina, (e-mail: sensel@st.fri.uniza.sk)
E. Parso, Faculty of management science and informatics, University of Zilina, (e-mail: parsoerik@gmail.com)
P. Sedlacek, Faculty of management science and informatics, University of Zilina, (e-mail: kefas555@gmail.com)
M. Vaclavkova, , Faculty of management science and informatics, University of Zilina, (e-mail: monika.vaclavkova@fri.uniza.sk)

teaches these students programming and after a few conversations with her we have identified, that these students have problem to understand program statements and how the program works. Then, we were searching for development environment that would help beginners to get better understanding of programming logic. We found many IDE's, but all of these environments expect that you have at least a basic knowledge of programming language syntax. We can now introduce some of these solutions and mention some of their advantages and disadvantages.

A. BlueJ

BlueJ is a Java development environment that is being developed and maintained at the University of Kent at Canterbury, UK, and La Trobe University, Melbourne. [2] This software is created with the purpose of introducing object-oriented programming to beginners, and its design differs from other development environments as a result. The user interface is much simpler unlike the other development environments such as NetBeans, Eclipse, etc. The environment supports interesting tools that are not available in other environments to make teaching object oriented programming faster. The best tool included in there is visualization of class structure using UML diagram. You can create the instances of objects which can be tested. You can inspect their values, invoke their methods, pass them as parameters and more. You can also directly invoke Java expressions without compiling. Thus, BlueJ is a powerful graphical shell/REPL for Java.

This IDE is widely used by our faculty in learning process. One of the reasons for choosing BlueJ was that it allows an approach where teachers and students deal with objects representing by graphical component. The pure Java language does not make learning process of object oriented programming very easy, because of numerous syntax and language details.

With BlueJ student can create an object and call its methods as the very first activity. Because users can create and interact with objects directly, concepts such as classes, objects, methods and parameters can easily be discussed in a concrete manner before looking at the first line of Java syntax. [2]

After a few years of experiences with our faculty learning process, the teachers know that this environment is not suitable for beginners, who have never met with programming so far. Program syntax, statements, input and output operations are really hard for them to understand. In conclusion, this IDE is good enough for people having at least some experience with programming. It will be much easier to learn principles of object oriented programming by using this IDE for them.

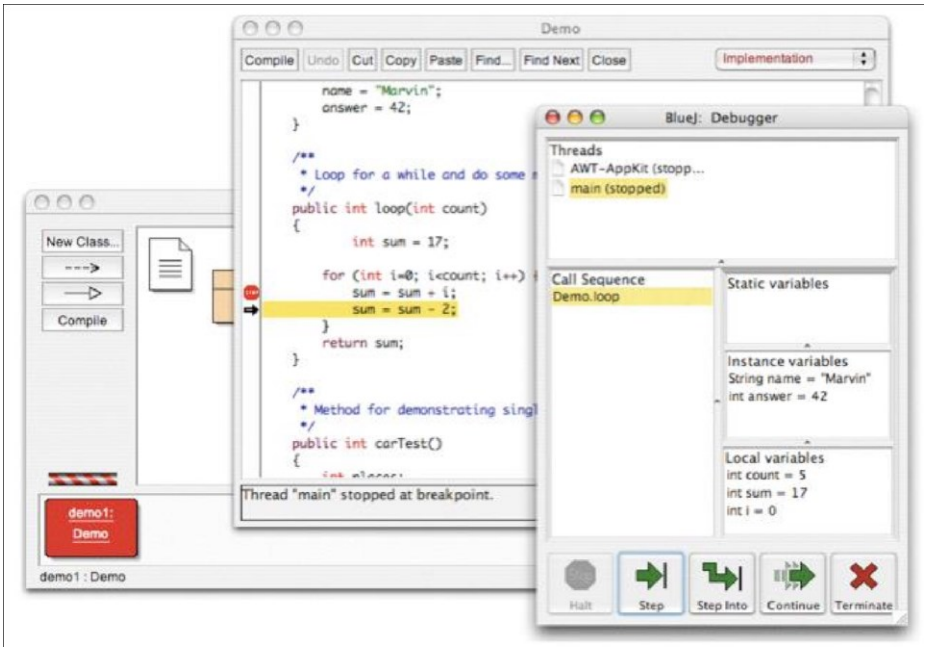

Fig 1 - BlueJ interface

Advantages:

- possibility to create instances and call methods without running whole project
- view class diagram and relationships between classes
- command interpreter
- good for better understanding of object oriented paradigm

Disadvantages:

- not suitable for big projects
- no abstraction from Java syntax
- no code generation
- intellisense is missing

B. GreenFoot

The GreenFoot is educational software designed to make learning Java programming easy and fun. This IDE is based on BlueJ. Greenfoot allows users to easily create simple graphics or visuals by providing predefined Java classes. *Greenfoot should be used to encourage the students to not give up programming if they find it difficult as it lets them immediately create the thing they were trying to do. There are many pre-defined methods such as turnLeft() or setLocation() which allow the programmer to easily move an actor around. This actor can have its controls mapped to the keyboard so that the student will be able to move his or her actor around. When the students see that they have managed to make a moving character with a few simple lines of code, they would probably be encouraged to make their Greenfoot application even more interactive. Since Greenfoot uses the actual Java code, the students are learning Java in fun way compared to the usual text based programs such as when creating a simple calculator program. [3]*

Although this environment is much simpler than BlueJ, beginners can still have problems with Java syntax. There are many environments and online solutions similar to GreenFoot. This approach of learning is better, because it is funny to create a simple game and see how objects behave when you call a method on them. The problem is, that there is no abstraction from Java syntax and you use text form of code to create program.

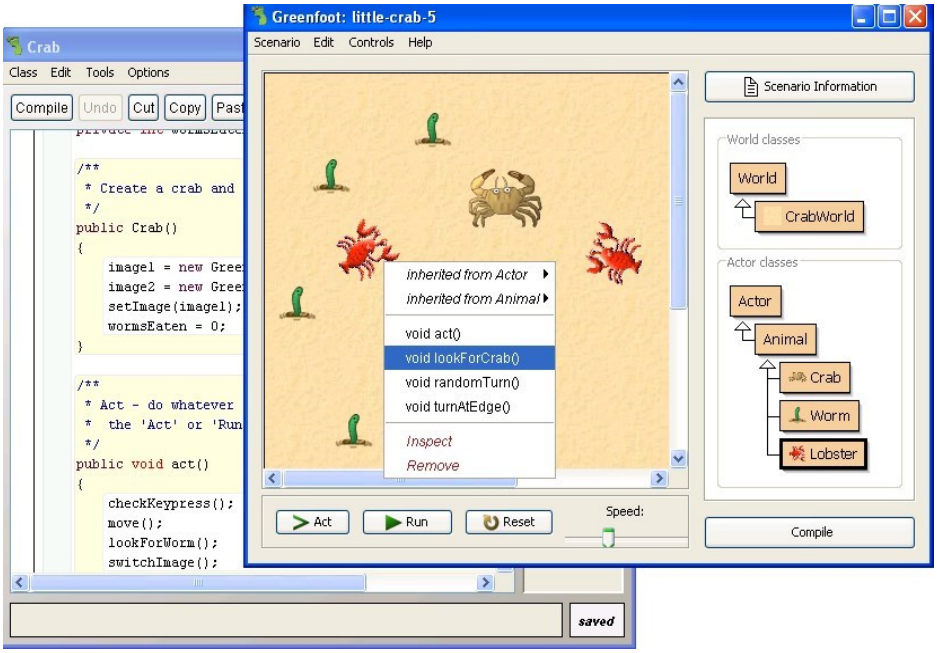

Fig 2 - Greenfoot interface

Advantages:

- possibility to create instances and call methods without running whole project
- predefined methods
- command interpreter
- visual representation of object and possibility to see the interaction with objects using methods
- predefined Java statements
- auto-completion
- syntax highlighting
- plugin system
- funny way of teaching programming

Disadvantages:

- not suitable for big projects
- no abstraction from Java syntax
- the actors are programmed in standard textual Java code
- no code generation

III. FRIMAN

There are many development environments similar to GreenFoot and BlueJ, but we have not found any that meet our expectations. And that is the main reason why we made a decision to create FRIMAN.FRIMAN will be an open source software, that has been developing within school subject since 2016. The main feature of IDE is possibility to create simple applications using flowcharts, without the knowledge of Java or any other programming language. The next image you can see shows the most important parts of this program. By using the main menu, we can create a new project, open or save existing one, run program, start debugging and many more. In the middle we can see editor. When we choose the class and method, there will be showed a flowchart diagram describing code inside of the selected method. User have a possibility to switch between Java code and flowchart representation of method. Because of it is hard for us to create diagram from Java code, code can be created just by using flowchart diagram and not by writing Java code. Diagram is created by drag and drop operations. All operations that user can do are represented by graphical item. A few of them you can see in the left down menu in picture below. These operations should provide a satisfactory support to create any functionality. In the future, we would like to provide some kind of intellisence that would make programming even easier.

At this moment, FRIMAN is divided into logic parts that communicate with each other to provide full functionality.

These parts are creating the core of FRIMAN, which consists of compiler, debugger, editor, controller, view providers, command provider etc. These parts have to communicate with each other, therefore it was necessary to design a way how these components will be connected. We have decided to use special approach how the commands will travel between application parts. Every part containing some logic is a command provider. When there is a user interaction with graphical interface, new command is created. This command is passed to controller, that will make a decision which command provider will be used to process command.

After the command is processed, the command return value will return to controller. Then controller choose the view provider that updates user interface. The same approach can be used for communication between command providers. As we mentioned before, the user of this environment does not have to have almost any knowledge of programming language syntax. This environment could be the first step, if someone wants to start programming. It will help

beginner to get better understanding of program logic. When user gets necessary knowledge how the program work, he can start to use GreenFoot or BlueJ and he can proceed to use professional environments.

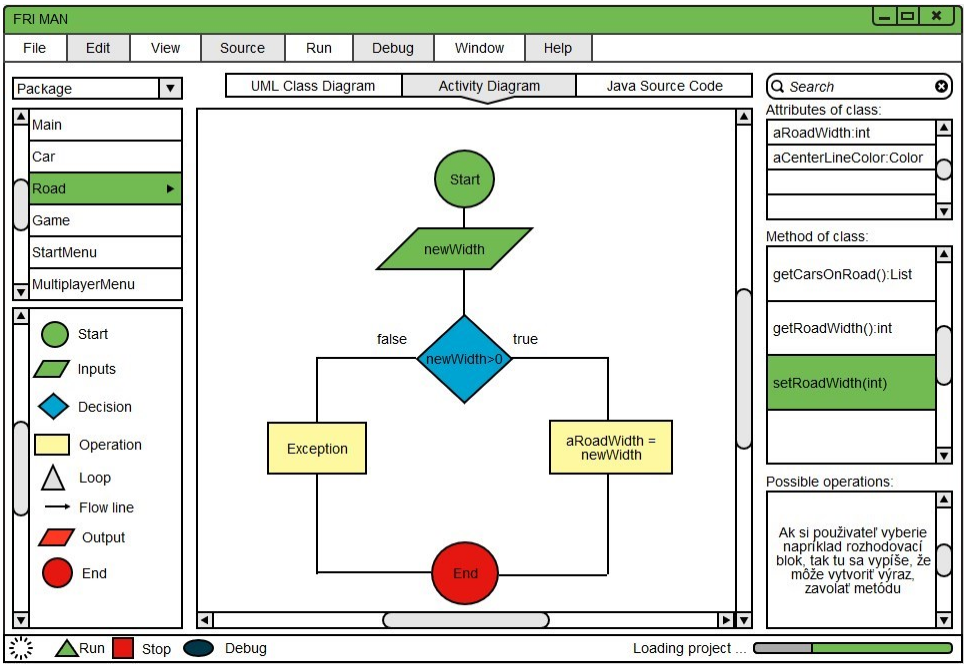

Fig. 3 - FRIMAN interface predesign

This programming IDE will provide some really useful features. Some of these features have been implemented in other environments already. One of them is the possibility to show existing classes in graphical form and to create instances of them. Then you can call methods on this instances and see the changes of these objects will be made. You can also see the return value of the method. This is the same feature that BlueJ provides right now, but we would like to make some graphical changes and make it more simple. You can see the current state of this feature in the following picture.

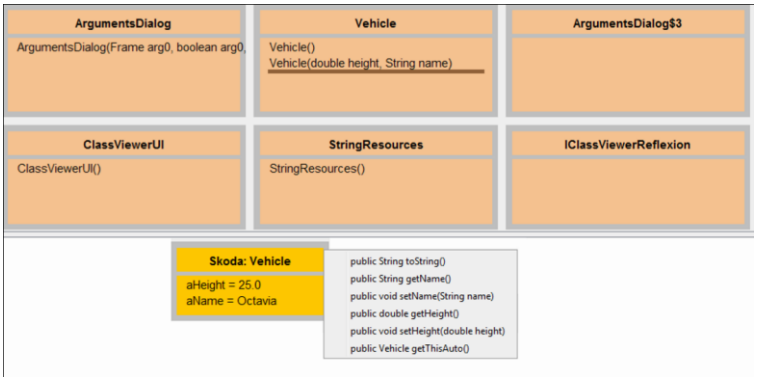

Fig. 4 - Classviewer

Moreover, there will be a code generator provided possibility to draw a flowchart and automatically create a Java code. It is one of the key parts we are working on right now and you can see its current look in image bellow.

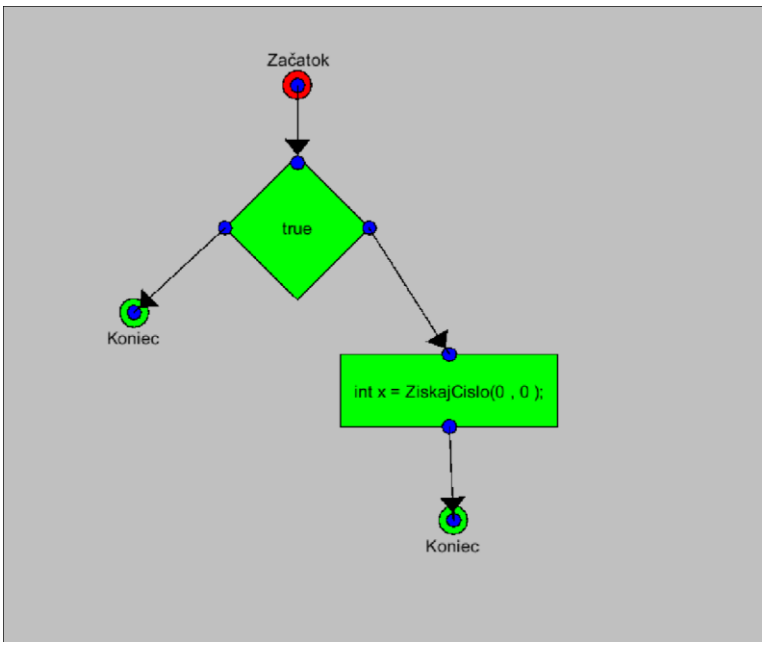

Fig. 5 - Flowchart diagram editor

As you can see at this moment this project is divided into a few parts that are developed independently but our team has already started to join them together to form a single application that allows at least some basic functionality.

IV. Conclusion

As mentioned before, this project started as an idea of creating a small plugin into BlueJ but after some time of problem analysis we have decided to push this idea on a whole new level. Because of its complexity, it would take much more time to complete than the time estimated at the beginning. There is a lot of things to do in the future. For example, one of the most significant parts we have not implemented yet is debugger. To make it easier and faster to work there could be some sort of intelisence with possibilities to complete half-written words, refactor code etc. If someone is interested in code optimalization there should be profiler providing information about memory consumption or method duration. In case of higher popularity there could be implemented to choose one of multiple languages.

References

[1] "What is information management?," 2016. [Online]. Available: https://ischool.uw.edu/academics/msim/what-is-information-management. Accessed: Dec. 30, 2016.
[2] D. J. Barnes, Objects First with java: A practical introduction using BlueJ. Harlow, United Kingdom: Pearson Education, 2016.
[3] "The benefits of using Greenfoot in class," A Journey into the Teaching Experience, 2013. [Online]. Available:https://teachtolearn2202.wordpress.com/2013/10/26/the-benefits-of-using-greenfoot-in-class/. Accessed: Dec. 30, 2016.
[4] "BlueJ,". [Online]. Available: http://bluej.org. Accessed: Dec. 30, 2016.
[5] "Greenfoot,". [Online]. Available: http://greenfoot.org. Accessed: Dec. 30, 2016