APPLICATION FOR STUDY TESTS, THE DRIVING TEST IN JAVA TECHNOLOGY

Professional paper

Nihad Karajko, Osman Ćehajić

Faculty of Technical Studies, University in Travnik, Bosnia and Herzegovina

Abstract

Driving test is almost an obligation of modern man. Preparation for the exam and only has been legal provided. This paper describes the development and implementation of an application that allows to learn and prepare for the test exam. Treated simulation, unregulated intersections, simulation intersection with traffic lights and trams, as well as all legally prescribed tests and examination result of learning. The implementation is done in Java technology so that it is ready to install in the Internet environment as well as the desktop version.

Keywords: *test study, driving, application, Java*

Introduction

Driving schools prepare exams for study tests and driving test. Schools are interested for better quality in education in less time and no waste. Authors have interviewed many driving schools and analysed these cases of usage:

- Driving simulation
- Study tests
- Analisys of exam results

User authentication – case of usage 1. Application authentication and evidence of registered and anonymous users is made in this case of usage.

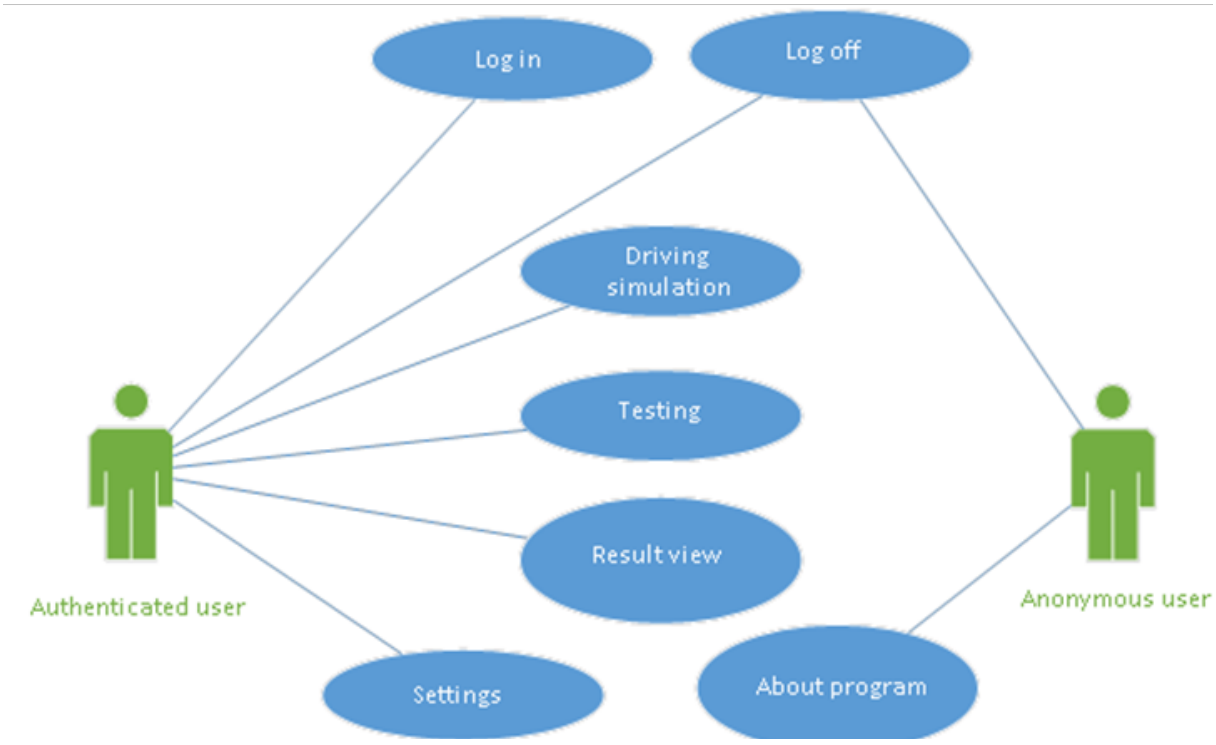

Figure 1: Regular application usage

Driving simulation – case of usage 2. Driving simulation and problem solving in traffic should let user manage situations on non-regulated crossroad, crossroad with traffic lights and crossroad with tram.

Study test – case of usage 3. Enables knowledge testing with tests made by the law, tests Test1... Test13 and Test C and Test D

1. Research

Application works under these technical conditions:

- 95% of the time online: web page must be available at least 95% of the time. Software update, hardware changes, failures and other problems should not affect this availability.
- Security identification and protected authentication: user names and passwords must not be saved to regular text fields or files.
- Protected personal user data: personal user data, such as addresses, telephone numbers and numbers of credit card must not be available to anonymous users.

Scalability: It is not yet inspected in this early stage because expected number of connected users or database size storage can not be predicted. However, architecture of the web aplication hosting place must be ready to work with scaling when great number of new users register and begins to use.

2. Problem solving

This program is built using Eclipse Software Development Kit tool. To go to the screen shown on the image below, import project „Auto Škola“ (File – Import Project and find the file). Then find „Default Package“ and open class „PrikazRezultata“ on the left side of the tool on Package Explorer.

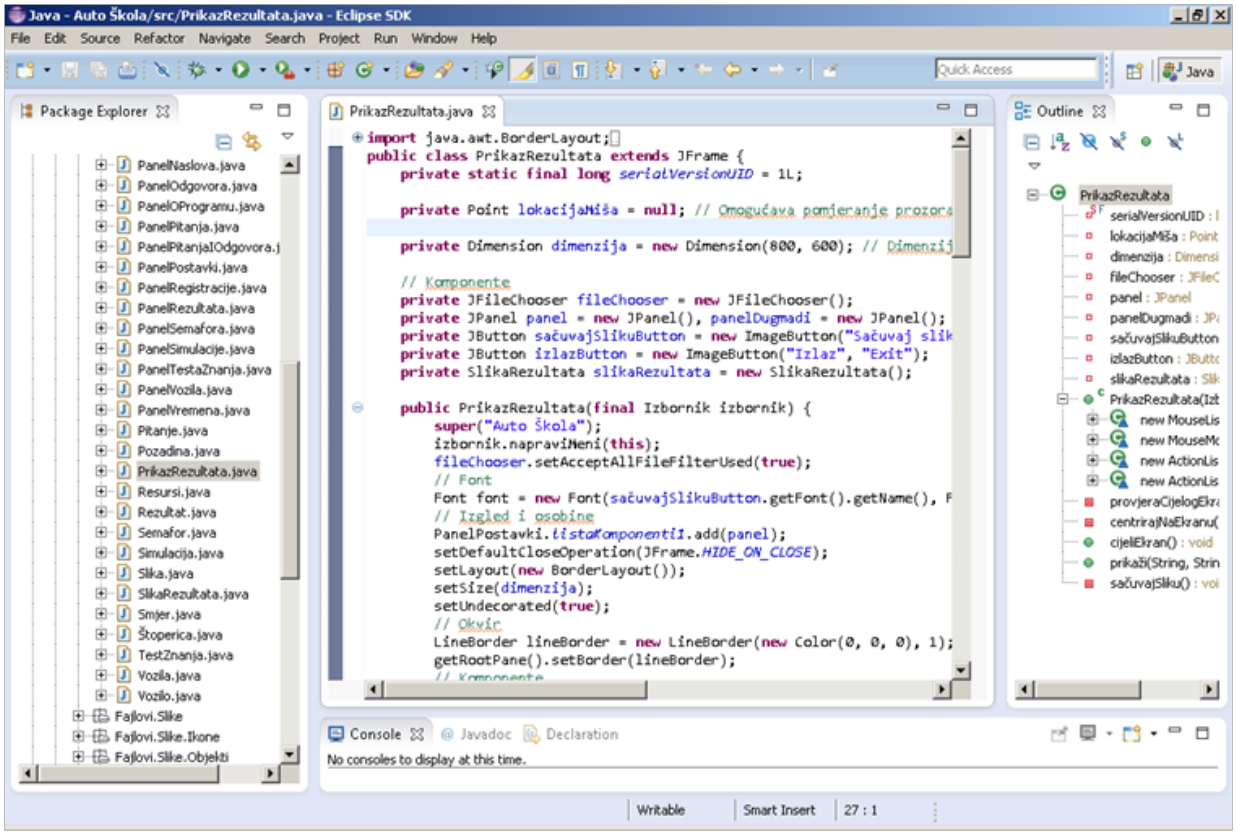

Figure 2: Eclipse Software Development Kit

Program contains 49 classes. Class *Prikaz Rezultata* (*Result View*) enables viewing of the results from the tests. Program is designed without using visual editor (component properties and layouts are made by code).

Variable *Lokacija Miša* (*Mouse Location*) enables window movement with left mouse click because window does not have control bar. *Dimenzija (dimension)* is the size of the window which is 800 pixels in width and 600 pixels in height.

Description of the components in the class:

JFileChooser is used in case if user wants to save the picture of the results.

Panel and *PanelKomponenti* (*Panel of Components*) are the background panels for components.

SačuvajSlikuButton (Save Image Button) – clicking on this button will show *JFileChooser* dialog and that is how user can save the picture of the results.

IzlazButton (Exit Button) – used for closing the current window and going back to previously opened window.

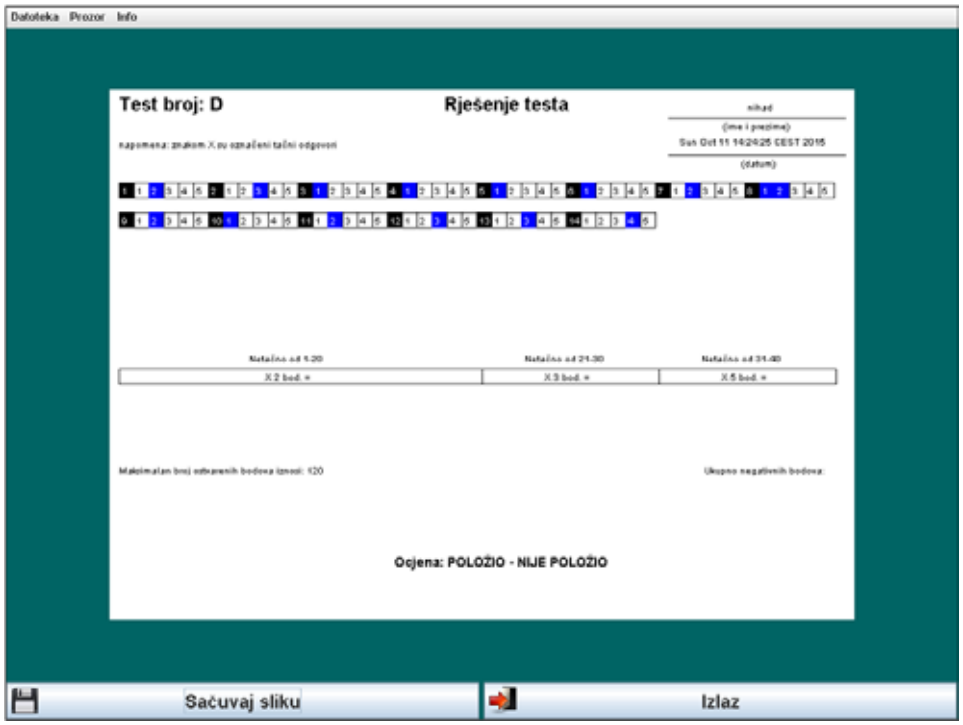

Figure 3: Results of the test

Command super („*Auto Škola*“) is added in constructor of *„PrikazRezultata“* class which is called for changing the title of the window. Command *izbornik.napraviMeni(this)* calls the function for creating window menu.

Next code shows layout, window properties, font, border...

Image of results of the test shows user name, date of the test and score. Every user can see if he has passed or failed the test.

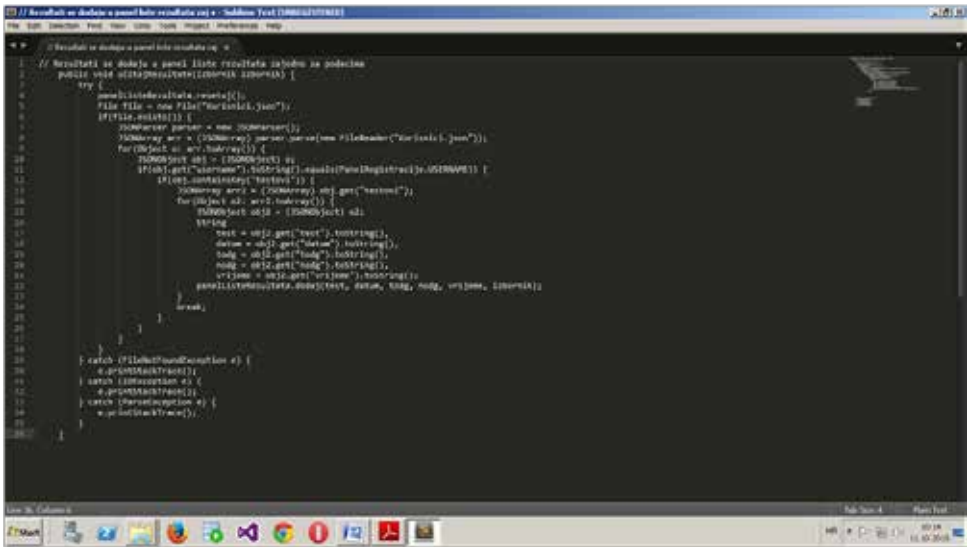

Figure 4: Function for loading user data from JSON database

The above image shows function which loads all results of single logged user from *JSON database*. Function uses class *„Izbornik“ (Menu)* as an argument which is the main form of the program. To free the program from possible errors which could make program stop responding we must use try – catch function for loading data. Errors which may show up are: *FileNotFoundException* (in case if the file does not exist), *IOException* (in case if loading of the file fails), *ParseException* (in case if the file could not be parsed to JSON format type).

The loading is free from errors which could occur so here goes the code for loading data from the file. *JSONParser* is the file parser for parsing files to *JSON data* format type. *JSONArray* is data array contained in parser when process of the parsing is completed. The goes for – each function which takes all data from the array one by one and creates structure (extraction) of necessary data for specified user.

Required data for database:

1. Test – name of the test,
2. Datum – date of the test,
3. Todg – number of correct answers,
4. Nodg – number of wrong answers,
5. Vrijeme – elapsed time.

When the process of data searching is over, found data is added to the list of the results for viewing.

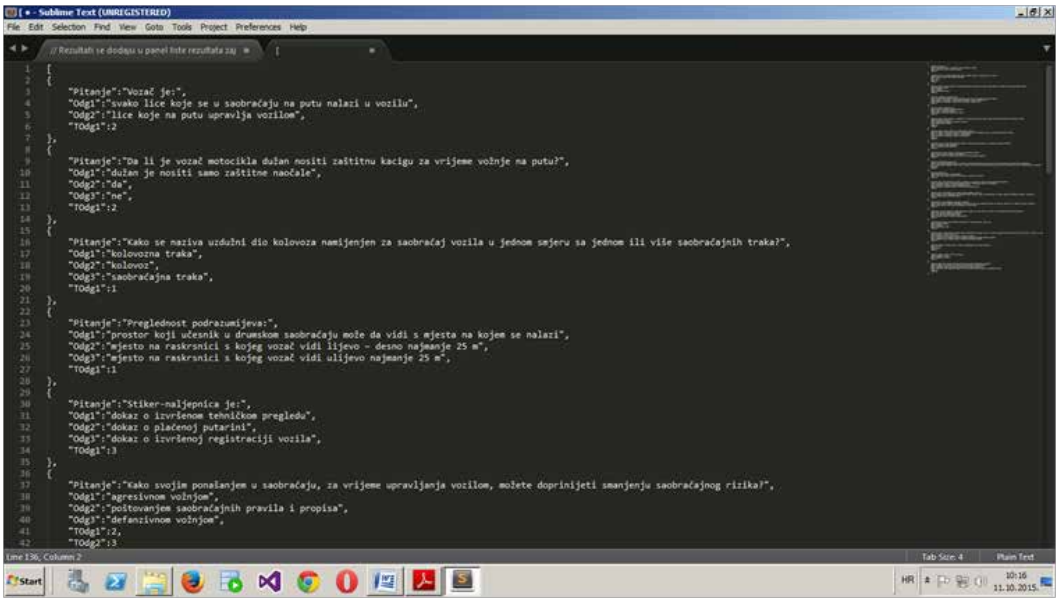

Figure 5: Primary database structure for tests

While working on this project, authors thought of using XML or JSON database type. JSON database type fits application needs better because it is more readable and data searching is faster comparing to XML database type.

All data such as user personal data and data of the tests which contain questions and answers is created in JSON database.

This picture shows one of the study test data structure. That structure contains: *pitanje (question), answers: odg1, odg2, odg3 and todg1 (correct answer).* So, every question can have more answers. Question structures are connected into one database which is an array and that array is the one group of the test. Tests can be divided to 3 groups: theoretical questions, traffic signs and rules on the crossroads.

3. Implementation

Main menu contains eight elements:

1. Simulation – opens simulation menu with settings,
2. Study test – opens study test menu which shows test names,
3. Fullscreen – toggles fullscreen mode,
4. Results – shows the list of results of the current user account,
5. Settings – options for managing graphic user interface (GUI) and account password,
6. About program – shows author information, version and other program info,
7. Log off – switching user account,
8. Exit – closing the program.

Figure 6: Application main menu

Figure 7: Crossroad with traffic lights

Explanation of how to use the simulation:

- *Brzina (speed)*: shows the vehicle movement speed which can be set from 1 to 10,
- *Dodavanje vozila (adding vehicles):* Combo-Box list shows names of the vehicles which can be added to the crossroad, user can add vehicles clicking on the arrow buttons, and clicking on x button will remove all vehicles from the cross-road,
- *Semafori (traffic lights):* click on the arrow button to change traffic light state (left – previous state, right – next state) and on *„isključi“* (turn off) option to turn off all traffic lights and vehicles will act like there are no traffic lights on the crossroad.
- *Kontrola (control):* clicking on the button which shows all directions will move all vehicles de-spite of the rules on the crossroad, clicking on the door will close the simulation window.

Every vehicle has its own id or number which is shown on the top. Press keys from 1 to 9 on keyboard to move vehicles through crossroad. If the vehicle can be moved by the rule of traffic it will be moved otherwise it won’t move through the crossroad.

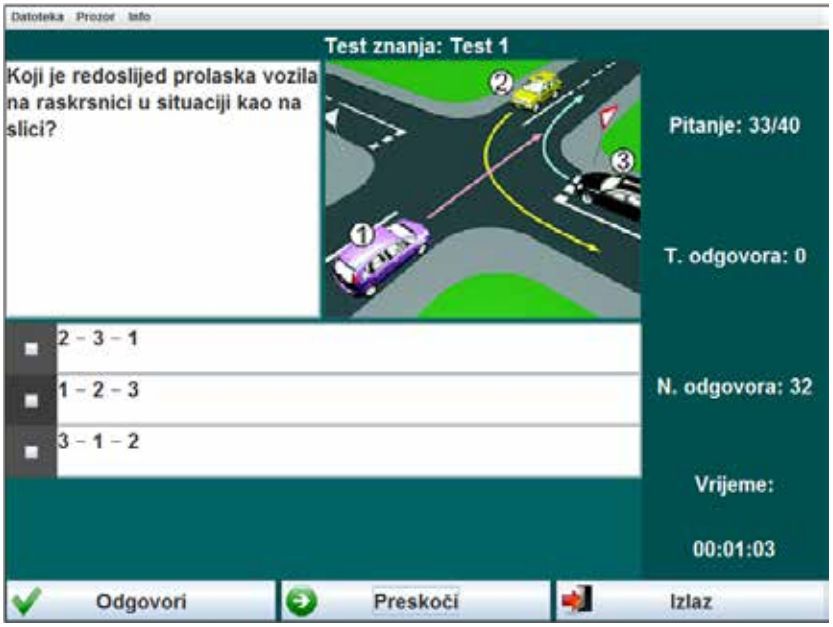

Figure 8: Study test

Test window contains question, picture (which can be showing traffic sign or traffic on crossroad), answers and test stats like number of question, number of correct and wrong answers and elapsed time.

To answer the question, user must choose correct answers and click on the button *„Odgovori" (Answer)*. Study tests may contain five answers. If user doesn't know the answer or simply wants to skip the question, he can click on the button *„Preskoči" (Skip)*. When user finishes the test, the results are writing into database so they can be viewed later.

4. Conclusion

Program for preparing and testing a study test is made in this work. Driving licence is the necessarity of everyone so this program makes learning and understanding traffic rules a lot easier. Every user can test its knowledge on more interesting way by using simulations and tests before taking the real test. This program is created in Java programming language and that made this application available to large number of operating systems (cross - platform). A lot of program improvement has come from analisys of this problem. One of the possible improvements could be work on 3D simulation and creating one virtual reality before the driver really sits in the vehicle and starts driving with the driving instructor. This area opens possibility of using artificial intelligence for simulating reality.

References

[1] Elipse SDK 4.2.2.; https://eclipse.org, pristupljeno dana 10. 10. 2015.

[2] Pravlnik o načinima i uvjetima organizacije ispita za vozače; http://www.mkt.gov.ba, Ministarstvo komunikacija i prometa BiH, pristupljeno dana 10. 10. 2015.

[3] Schildt, H.[2006] JAVA J2SE 5, Zagreb, Mikro knjiga.

[4] Bloch, J.[2006] Efikasno programiranje na Javi. Zagreb, Mikro knjiga.

[5] Chapman, S. J. [2000] Java for Engineers and Scientist. New York, Prentice Hall.

[6] Tutorial; http://java.sun.com/docs/books/tutorial/pristupljeno dana 10. 10. 2015.

[7] Booch, G.; Rumbaugh, J.; Jacobson, I. [1999] The Unified Modeling Language User Guide. Addison-Wesley.

[8] Fowler, M.; K. Scott, K. [2000] UML Distilled, 2nd ed., Addison-Wesley.

[9] Fowler, M. [2004] UML Distilled: a brief guide to the Standard object modeling language, 3rd ed., Addison-Wesley.

[10] Holub, A. I. [2012] Allen Holub's UML Quick Reference (UML 2.0), version 2.1.3, dostupno na:http://www.holub.com/goodies/uml/, pristupljeno dana 10. 10. 2015.

[11] Lethbridge , T. C.; Laganiere, R. [2005] Object-Oriented Software Engineering, McGraw-HillEducation.

[12] OMG [2011], Object Management Group, OMG Unified Modeling Language (OMG UML),Infrastructure, Version 2.4.1, 2011, http://www.omg.org/spec/UML/2.4.1/Infrastructure/PDF/, pristupljeno dana 10. 10. 2015.

[13] Rational [2001], Rational Software Corporation, Artifact: Use-Case Model, dostupno na:http://www.ts.mah.se/RUP/RationalUnifiedProcess/process/artifact/ar_ucmod.htm, pristupljeno dana 10. 10. 2015.

APLIKACIJA ZA UČENJE TESTOVA VOZAČKOG ISPITA U JAVA TEHNOLOGIJI

Sažetak

Polaganje vozačkog ispita je skoro pa obaveza modernog čovjeka. Priprema za polaganje i samo polaganje ispita je zakoski propisano. U ovom radu je opisan razvoj i implementacija aplikacije koja omogućava učenje i pripremu za polaganje testova za ispit. Obrađena je simulacija neregulirane raskrsnice, simulacija raskrsnice sa semaforima i tramvajim, te svi zakonom predviđeni testovi te pregled rezultat učenja. Implementacija je urađena u JAVA tehnologiji tako da je spremna za instalaciju u Internet okruženju kao i desktop izvedba.

Ključne riječi: *učenje testova, vožnja, aplikacija, Java*

Received: October 12, 2015

Accepted: December 14, 2015

Correspondence to:

Osman Čehajić

Faculty of Technical Studies, University in Travnik

Bosnia and Herzegovina

E-mail: ocehajic@gmail.com