Coding Game
Programming based educational application

Adrian George BOSTAN
Teacher Coordinator: Lect. dr. eng. Adina COCU

Abstract

In the present paper, I will introduce an educational application in the form of a computer puzzle game, which allows the user to test his programming aptitudes in a user-friendly graphical environment. The game can be used as a tool for better understanding a particular programming language. The versatility of the application stems from the ability to easily import the structures of any programming language in the game. As a result, the user has the power to choose from a wide range of programming languages. The application comes with levels for three predefined languages: C#, C++ and Java.

The **Coding Game** bundle consists of the game along with two tools for adding, removing and modifying the content of the game. The main application has three game modes, each mode targeting a different aspect of learning a programming language. The three game modes are: *Puzzle*, *Order* and *Test*, and together, they can make the process of learning a programming language more pleasant and more fun.

The goal of this application is not to replace IDE's or programming books, but to serve as an additional tool for the user, allowing him to improve his skills and knowledge through playing a game, thus making the whole experience more enjoyable.

1. Introduction

The main purpose of educational systems is to help a particular person learn faster and more efficiently. It's common knowledge that the process of learning is difficult and it's based on the action of continually accumulating information. This process can be improved by using these educational systems in the phases of accumulating and testing knowledge.

In the last couple of years, there have been developments in educational systems based on computer games. However, the use of this kind of system is still isolated and there is a general misconception that educational games are only suitable for children. The presented application targets people who want to test their knowledge of programming. Consequently, the user can be any person interested in learning how to program.

Computer games have the ability to keep the player focused for long periods of time while entertaining him. They enhance cognitive skills, such as problem solving, decision making, organization, critical thinking, etc. This is the reason why games can be used for educational purposes, by challenging the player to acquire the presented information, without getting him bored. The visual elements specific to computer games can make learning a pleasant experience while providing a proper level of relaxation and can motivate the player to further improve his skills.

Puzzle games are perfectly suitable for learning because they don't require the undivided attention of the player. This means that the user is not exhausted after playing the game, but actually relaxed. **Coding Game** is a puzzle game at its core, intended for testing and improving abilities related to programming. The application can help in better understanding the general structure of a program, in memorizing the syntax of a programming language and in adopting a particular programming style. By going through the levels of the game, the player observes different techniques used for achieving different things, techniques which he can acquire for future use.
The tools provided with the actual application have the role to extend the game by adding levels for the predefined programming languages (C#, C++, Java), or by adding in new ones. This feature makes **Coding Game** a universal learning tool for programming languages.

2. Application description

2.1 General description

The application and the tools were programmed in C#, using the XNA Game Library developed by Microsoft. The IDE I used for writing the application is Microsoft's Visual C# 2010 Express Edition.

Coding Game consists of the main application (the game) and two graphical tools for loading content in the game: the *Language Tool* and the *Level Tool*. The *Language Tool* is used for importing programming languages in the game, and the *Level Tool* is used for loading levels for a particular programming language and a particular game mode.
The game has three game modes: *Puzzle*, *Order* and *Test*. All the game modes are timed. The time in which a particular level is to be completed is set through the *Level Tool,* when loading new levels in the game.
Each programming language in the game has its own three game modes. So, the languages are separated from one another. Before choosing a game mode from the three mentioned above, you have to choose a programming language from the list of languages detected by the game.
Every sequence of code present in the game, depending on the programming language, has syntax colouring. The colours used to highlight a particular language are set through the *Language Tool*.

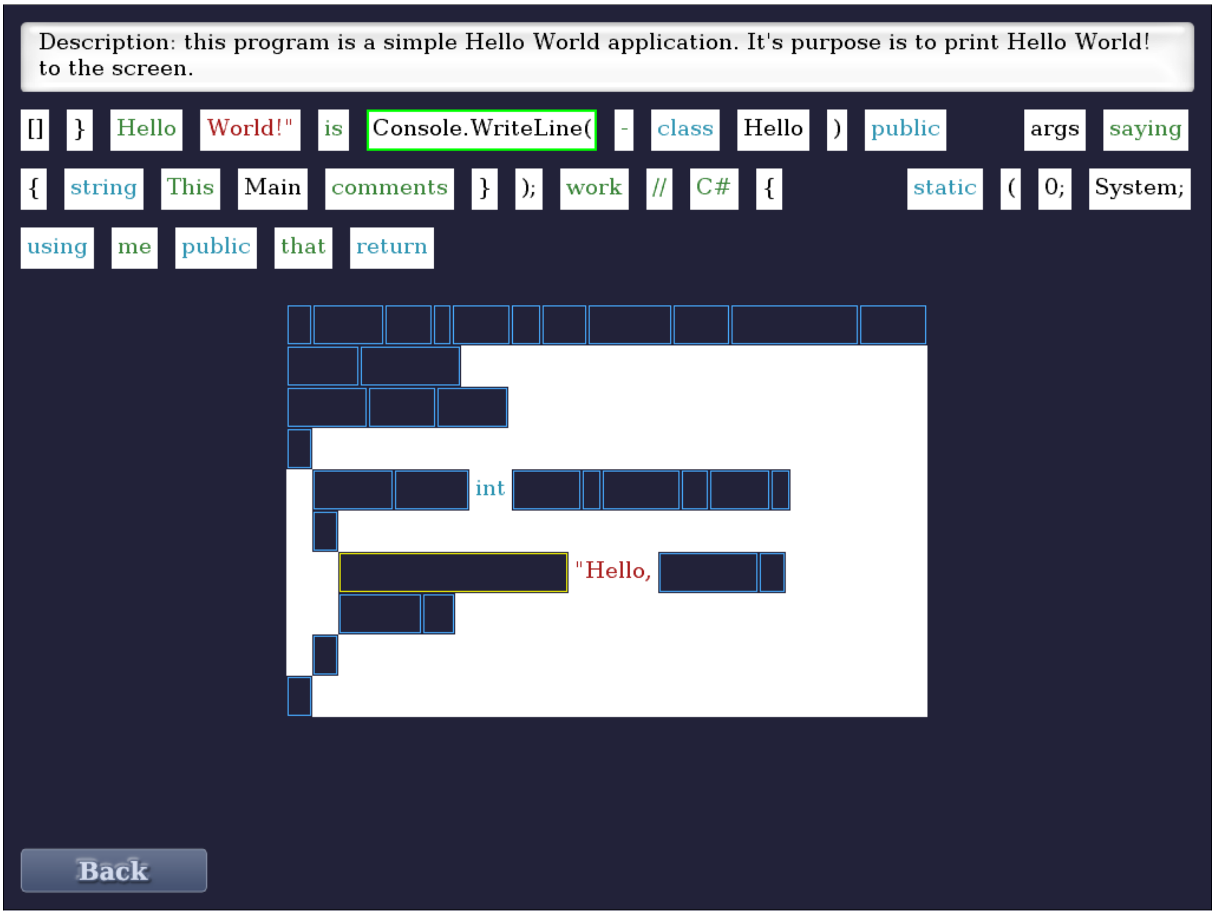
2.2 Important classes

- ScreenManager - Controls all the game states.
- GameState - Creates a basic structure for all the game states. All the game states extend this class.
- TextureManager - Static class used for loading all texture assets in the game.
- Game States:
    - Menu - The menu game state. It's the active game state when launching the application.
    - Help - Provides some information about the game modes.
    - SelectLanguage - Allows you to select a programming language to play with.
    - ChooseGame - Select a game mode: *Puzzle*, *Order*, *Test*

    - Puzzle - Loads all *Puzzle* levels and controls them, making them active one at a time.
    - Order - Loads all *Order* levels and controls them, making them active one at a time.
    - Test - Loads all *Test* levels and controls them, making them active one at a time.
- Levels:
  - PuzzleLevel - a *Puzzle* level. The Puzzle class contains and controls all the *Puzzle* levels.
  - OrderLevel - an *Order* level. The Order class contains and controls all the *Order* levels.
  - TestLevel - a *Test* level. The Test class contains and controls all the *Test* levels.

2.3 The game modes

Puzzle: the first part of **Coding Game,** in which the player receives the description of a program and a set of puzzle pieces belonging to a source file. The structure of the source file is rendered on the screen, but without any code (only white space is shown). Every line of code in the original source file is divided into puzzle pieces. The player must use the puzzle pieces to recreate the original source with the working program, specified by the description. The puzzle pieces are shuffled each time, so that the main focus is to understand how and why particular pieces of code are used. A *Puzzle* level is done when the source file has been recreated using the puzzle pieces, or when the time set for the level is up.
Additional levels for this game mode can be loaded with the *Level Tool.*



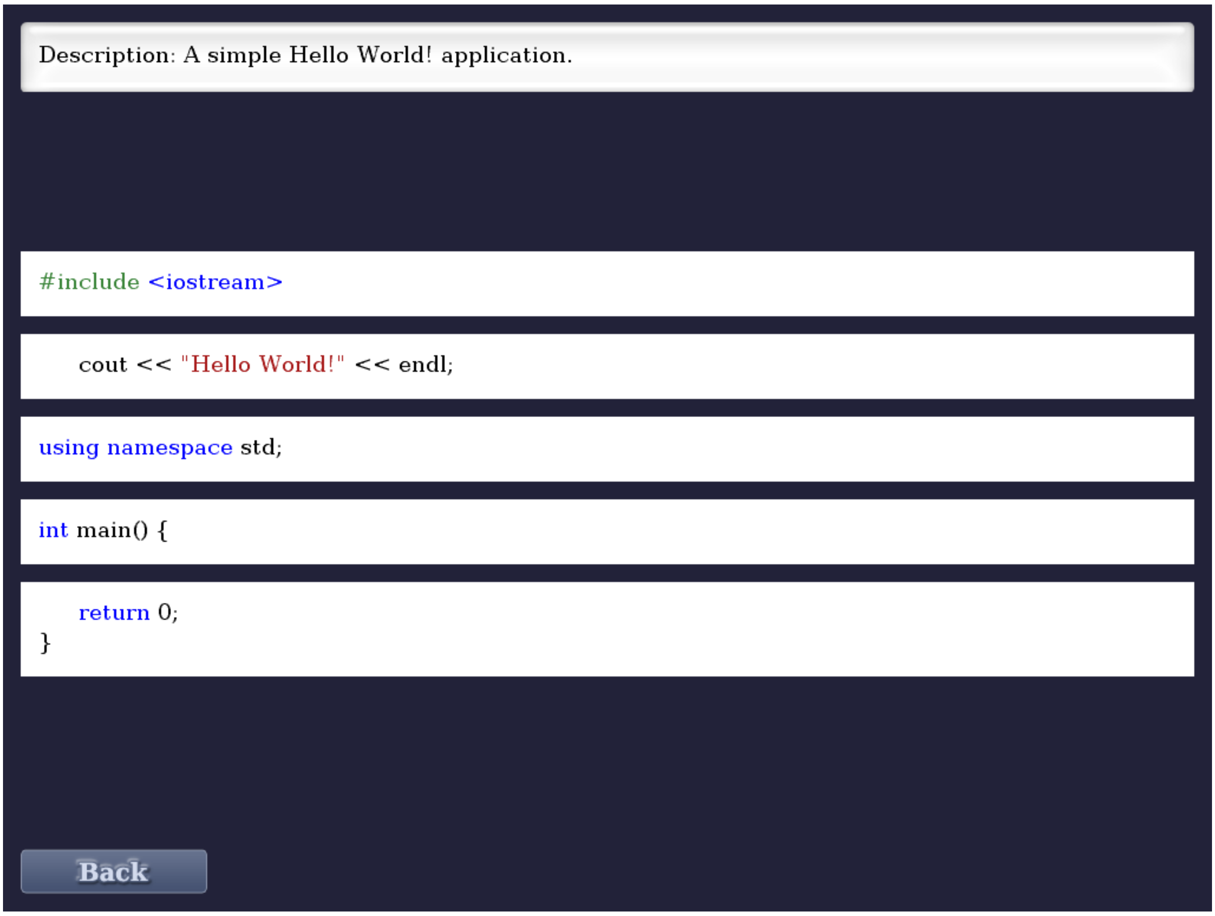
Description: this program is a simple Hello World application. It's purpose is to print Hello World! to the screen.

Figure 1 – A Puzzle level

Order: is the second game mode of **Coding Game** and consists of a number of code sequences that form a program. The code sequences are not in the correct order. Along with the sequences, a description of what the program is supposed to do is provided. The player has to put the chunks of code in the correct order, to create a working program that fits the received description.

The main difference between the *Puzzle* game mode and the *Order* game mode lies in the intended learning concept. The *Puzzle* game mode is more complex, the player has to unscramble a source file by placing the puzzle pieces in the empty table which represents the source file. The *Puzzle* game mode helps with learning the keywords and their use, memorizing the structures used in a program and with adopting a proper programming style. In contrast, the *Order* game mode provides the user with a number of chunks of source code and the emphasis falls on learning the logic behind a program, the right order in which things should be done and observing the used techniques which can serve as future references for the player. An *Order* level is over when the sequences of code are in the correct order, or when the time is up.

Additional *Order* levels can be loaded with the *Level Tool*.



Description: A simple Hello World! application.

```
#include <iostream>
```

```
    cout << "Hello World!" << endl;
```

```
using namespace std;
```

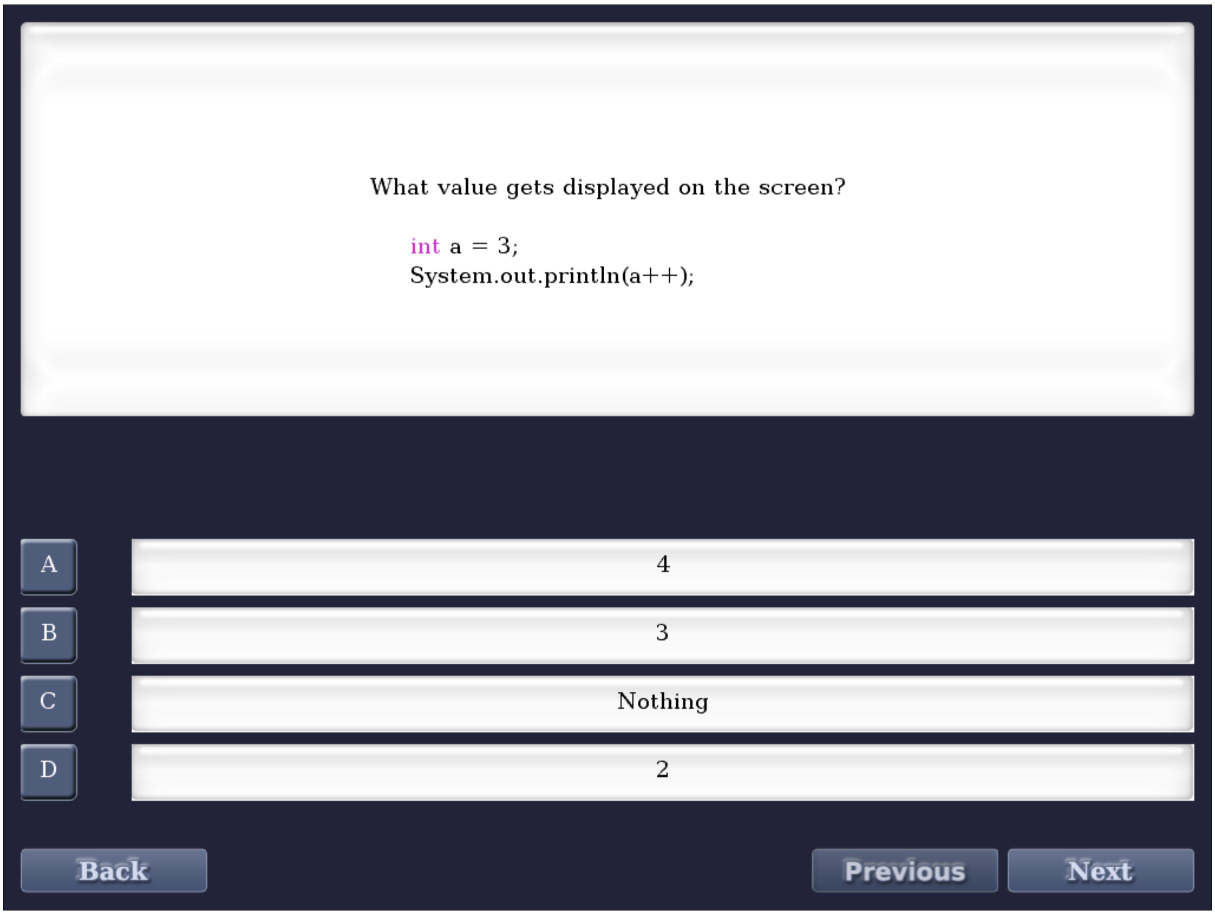
```
int main() {
```

```
    return 0;
}
```

Figure 2 – An Order level

Test: is the last game mode of the application and it is based on a quiz. The player gets a number of questions about programming <egz: What is the result of the following sequence of code?> which he has to answer. Every question has four answers. Only one of the four answers is correct. This game mode tests the theoretical aspects of learning a programming language and challenges

the player to see if he has a solid grasp of the programming concepts. A *Test* level is over when all the questions have been answered, or when the time allocated for the test has passed.
Additional *Test* levels can be loaded with the *Level Tool*.



What value gets displayed on the screen?

```
int a = 3;
System.out.println(a++);
```

A 4

B 3

C Nothing

D 2

Figure 5 – The level tool

2.4 The game modes

The tools are used for adding personal content to the application. There are two tools available: the *Language Tool*, and the *Level Tool*. The tools feature an intuitive interface and are very easy to use. The tools make the application versatile, because through them, the user can load his own languages and levels for those languages.

The **Language Tool**: is used for importing languages in the game. In order to import a language, all the keywords of that language have to be provided. Also, the colours used to highlight the syntax of the loaded language have to be chosen. This tool can also remove languages from the game.

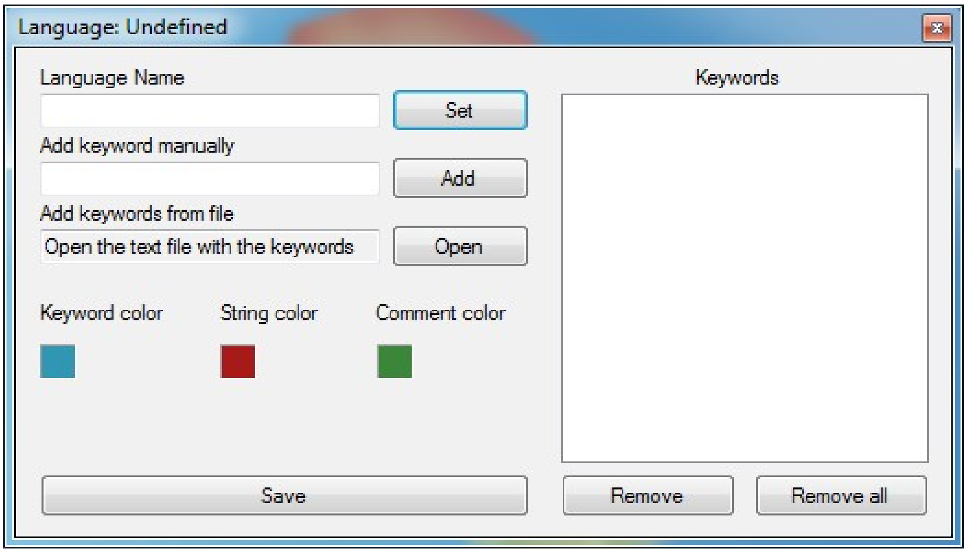


Figure 4 – The language tool

The **Level Tool**: is used for loading levels for a particular game mode (*Puzzle*, *Order*, *Test*) and for a particular programming language. The languages imported with the *Language Tool* are detected by the *Level Tool*. As each game mode is different, the process of loading a level is different for each of the game modes. Also, this tool can remove levels from the game.

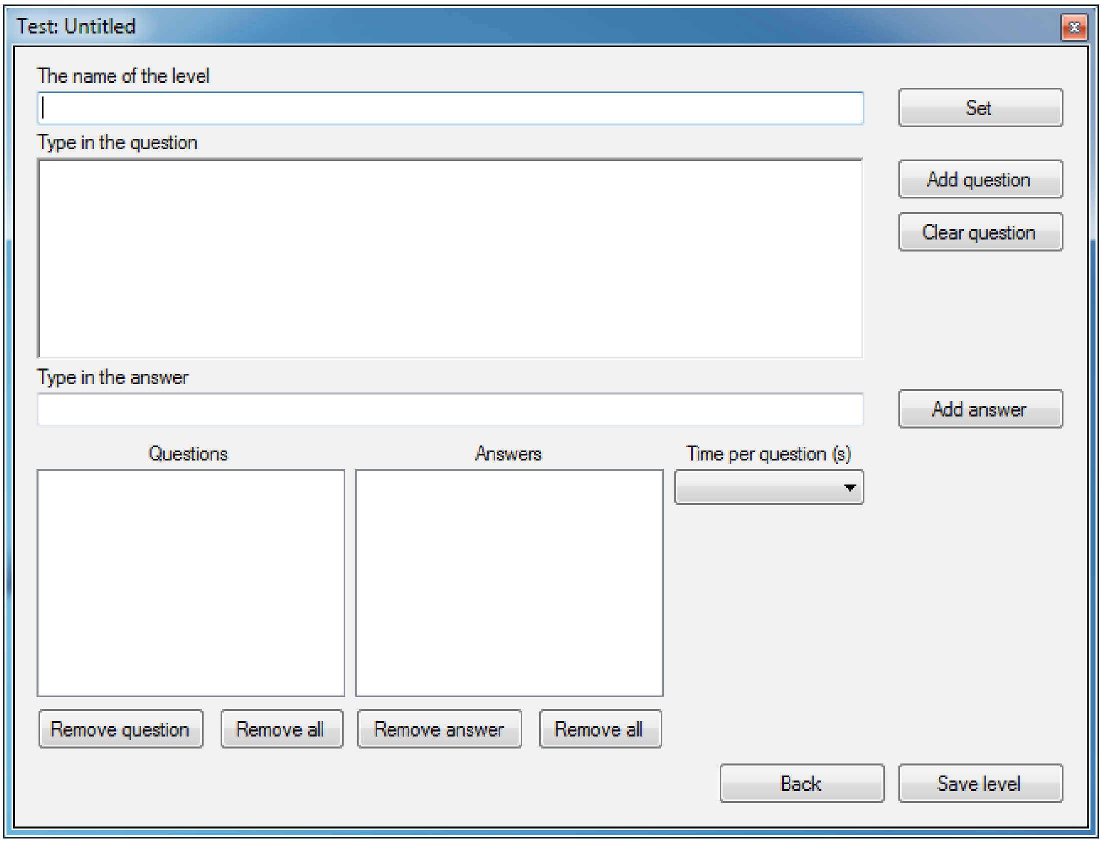


Figure 5 – The level tool

3. Conclusion

At this time, **Coding Game** only works locally. My ambition is to give the application the ability to take its resources from a server. This way, the application could even be used in schools to test students on their programming skills. The students would have access only to the executable of the application. The content of the game would be located on the school's server so that the students couldn't access it. The teachers would use the tools to create different kinds of tests, depending on the game mode, for the students. The content would be changed for every test, removing the old test, and loading a new one in the game. This kind of testing would be beneficial because it's more enjoyable for the students and would motivate them to learn.

Acknowledgements: I would like to thank dr. Adina COCU for all her patience and support.

References

[1] Professor Angela McFARLANE, *Report on the educational use of games*.

[2] dr. eng. Adina COCU, *Contributions to The Knowledge Representation Methods in Intelligent Tutoring Systems,* 2009.

ADRIAN GEORGE BOSTAN
Universitatea “Dunarea de Jos” Galati
Facultatea de Automatica, Calculatoare, Inginerie Electrica si Electronica
Str. Domneasca, nr. 47, Galati
ROMANIA
wait4.asecond@gmail.com