Journal of Information Engineering and Applications
ISSN 2224-5782 (print) ISSN 2225-0506 (online)
Vol.8, No.7, 2018

Application and Development of Java Gaming Engine: A Case of Leaping Frog Game

Muhammad Nadeem
Department of Computer Science and Information Technology, Ghazi University, Dera Ghazi Khan

Amjad Rasool Farazi
Department of Computer Science, Bahauddin Zakariya University Multan

Muhammad Tariq
Department of Computer Science, Government College University Faisalabad, Layyah Campus, Pakistan

Abstract
As it has been seen, in modern computers the processing power has grown to absolute level, it made easier to develop accurate, graphical simulation in two and three dimension. The aim of the game is to swap the frogs over using the simple set of rules. The java provides sound playback, graphics manipulation, and a game loop along with supporting classes. Java Gaming engine is the strongest tool which provides multi functionality for the development of games. Nowadays, games are not only enjoyment tools, but also used for learning and improving abilities. It is concluded that there must be a balance between realism and speed.
Keywords: Java programing, System design, Leaping frog, Graphic manipulation

Introduction

A deep connections between the idea of game and the idea of computation, and indeed that games should be thought of as a valid and important model of computation. When it comes into single-player games, also called Console games, I think which are more significant in individuals‟ life although multi-player games are getting more and more popular in more than one aspect. Players can play the games flexibly without internet access, unlike a game with multiple players competing with or against each other to reach the game's goal.

However, it is a curious fact that various kinds of games seem to be in especially direct correspondence with particular models of computation.

This game is to develop the intellectual ability of child. It is simple for play rule. It is the basic level of game for child to improve the mental ability. This game have three levels Easy, Medium and hard. In which some red and green frog sitting on the stands. The aim of the game is to swap the frogs over using the simple set of rules. The red frogs only can move to the right and the green frogs only can move to the left sides. A series of 4 articles by Chris Hecker [1, 2, 3, 4] present a simple method for creating a 3D physics simulator capable of collisions for rigid, polygonal solids.

The java provides sound playback, graphics manipulation, and a game loop along with supporting classes. Java Gaming engine is the strongest tool which provides multi functionality for the development of games. Nowadays, games are not only enjoyment tools, but also used for learning and improving abilities.

The Java Instructional Gaming Engine was developed to support interactive curriculum. It is a java-based 2D engine that simplifies the process of creating 2D games and game-based curriculum. The engine provides sound playback, graphics manipulation, and a game loop along with supporting classes.

Recently, there are a multitude of new features provided to make people more convenient and familiar to use their mobile phones like GPS, built-in camera, Infrared, Bluetooth, and WLAN connections.

Literature Review

Evolutionary algorithms (EAs) are stochastic search methods that mimic the metaphor of natural biological evolution and/or the social behavior of species. The optimized behavior of such species is guided by learning, adaptation and evolution [5] “Improving particle swarm optimization by hybridization of stochastic search heuristics and self-organized criticality”. Denmark: Aarhus University. Researchers have developed computational systems that mimic the efficient behavior of species such as ants, bees, birds and frogs, as a means to seek faster and more robust solutions to complex optimization problems. The first evolutionary-based technique introduced in the literature was the genetic algorithm [6]. *Adaptation in Natural and Artificial Systems*, Ann Arbor, MI: University of Michigan Press. In an attempt to reduce processing time and improve the quality of solutions, particularly to avoid local optima, various genetic algorithm improvements and other EAs have been proposed during the past 10 years, with the latest and perhaps most promising technique being the shuffled frog-leaping (SFL) algorithm [7]. Optimization of water distribution network design using the shuffled frog leaping algorithm [8].

The shuffled frog-leaping algorithm is a memetic meta-heuristic that is designed to seek a global optimal

solution by performing a heuristic search. It is based on the evolution of memes carried by individuals and a global exchange of information among the population [7]. Optimization of water distribution network design using the shuffled frog leaping algorithm. In essence, it combines the benefits of the local search tool of the particle swarm optimization [9]. "Particle swarm optimization". In *Proceedings IEEE International Conference on Neural Networks*, 1942–1948. Piscataway, NJ: IEEE Service Center and the idea of mixing information from parallel local searches to move toward a global solution [11]. The SFL algorithm has been tested on several combinatorial problems and found to be efficient in finding global solutions [7].

System Design

According to the method of usage, we can divide games into two main types, one is called multi-player games, and the other is called single-player games. However, in terms of the form of representation, which can be classified into text-mode games and graphic-mode games as well. Speaking of text-mode games, which is a form of games through swapping characteristics. We can play the games by replying the information according to the tips from mobile games. Furthermore, there are two basic approaches to achieve text games, SMS games and WAP games.

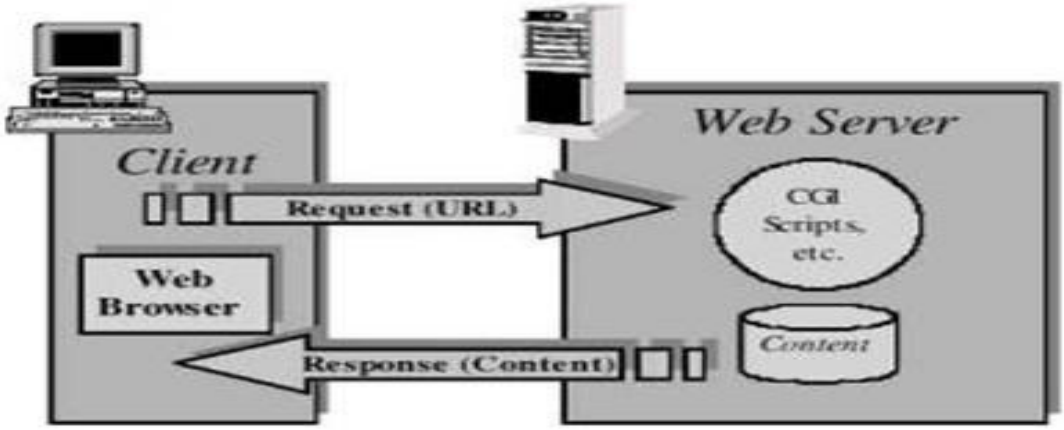

Figure 1: Working process of WAP games.

Graphic-mode Games

As far as graphic-mode games, which are the electronic games that involve interaction with a user interface to generate graphic? They are more like animations. More specifically, they can be divided into several areas like embedded games, J2ME games and some others.

Java Programming Language

With the rapid development of society, Java technology has become one of the most significant parts in our daily lives. From mobile phones to laptops, navigation systems to games, we can use Java technology to make them more functional, entertaining and convenient. The number of Java enabled mobile phones worldwide is over 250 million according to a press release from Sun titled: "Java technology is everywhere, surpasses 1.5 billion devices worldwide", and the number will continue to increase considerably.

Characteristics

Java is an object-oriented, simple, distributed, secure, dynamic and high performance programming language developed by Sun Microsystems, a company best known for its high-end UNIX workstations. But now, it is a part of Oracle Corporation. Java language is designed to be small, simple, and portable across platforms and operating systems which derives much of its syntax from C and C++ but has a simpler object model and fewer low-level facilities.

Classification

Recently, Java technology has become more and more popular due to its high qualities, and owing to its extendibility feature, we can applied it into various kinds of fields. There are several Java platforms in correspondence with different goals of development for programmers, offering them numerous proper resolutions they require.

Below figure demonstrates an overview of the components of different Java platforms and the areas of various applications.

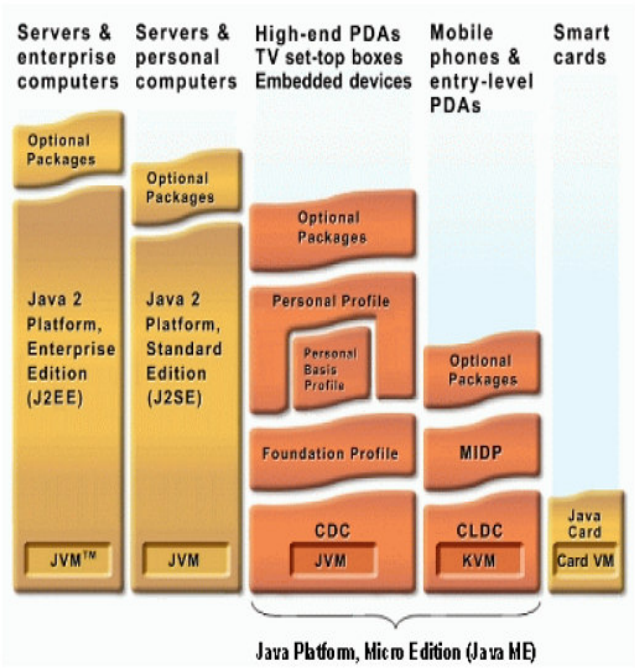

Figure 2: Different Java platforms

As can be seen from the graph, Java can be divided into three types of edition: Java Standard Edition (Java SE), Java Micro Edition (Java ME), and Java Enterprise Edition (Java EE).

Java Development Tools

Java can be implemented on a variety of platforms, but there are two main development tools that a number of people would like to use, one is called Eclipse and the other is called Net Beans.

System Development

The environments continue to improve with each version including features such as global search. However IDEs like eclipse have a much greater feature set for managing large projects. The game or simulation must run inside the environment's window. While these windows help with interaction, debugging, and code design, they can also detract from the actual game once it is complete.

Sometimes a game or interactive curriculum is most effective when the screen is not cluttered with menus and options around it.

These windows can detract from the lesson being taught, or reduce the cinematic experience a game may be trying to create. Another major drawback is the students do not get to use the features of a popular integrated development environment such as Eclipse. One of the really nice features of Eclipse is how it handles compile errors. Eclipse compiles code as you are typing and highlights errors as well as gives suggestions on how to fix them.

Packages

Packages are used in Java in-order to prevent naming conflicts, to control access, to make searching/locating and usage of classes, interfaces, enumerations and annotations easier etc.

A Package can be defined as a grouping of related types (classes, interfaces, enumerations and annotations) providing access protection and name space management.

Some of the existing packages in Java are.

- **Java. Lang** - bundles the fundamental classes
- **java.io** - classes for input , output functions are bundled in this package

Programmers can define their own packages to bundle group of classes/interfaces etc. It is a good practice to group related classes implemented by you so that a programmers can easily determine that the classes, interfaces, enumerations, annotations are related.

Since the package creates a new namespace there won't be any name conflicts with names in other packages. Using packages, it is easier to provide access control and it is also easier to locate the related classed

Creating a package

When creating a package, you should choose a name for the package and put a **package** statement with that name at the top of every source file that contains the classes, interfaces, enumerations, and annotation types that you want to include in the package.

The package statement should be the first line in the source file. There can be only one package statement in each source file, and it applies to all types in the file.

If a package statement is not used then the class, interfaces, enumerations, and annotation types will be put into an unnamed package.

The import Keyword

If a class wants to use another class in the same package, the package name does not need to be used. Classes in the same package find each other without any special syntax.

Source Code of Game

In the source code of the game some packages are imports from java libraries for example

```
 import java.io.*;
import java.util.ArrayList;
import java.applet.*;
import java.awt.*;
import java.awt.event.*
```

After this we develop a class leaping frog in which we define all variables and functions which are used in the game for different purposes.

```
public class leapingFrog extends Applet
{
        int numRed; // number of red frogs
        int numGreen; // number of green frogs
        int numPlaces;
        ArrayList <Integer> frogs;
      int autoCounter = 0;
    // The images of the frogs
        Image redPic, greenPic, Pic, Win, moon, grass, pond, rst;
        AudioClip ac, jump;
        //Button reset;
        int displayMode = 10;
        Font textFont = new Font("Helvetica", Font.BOLD, 17);
```

Now we use the init( ) function. The init( ) compile the code sequentially up to down. We set the color black for background color, set the font text and color for font text. Then use the getImage( ) method to get the images from the specific location.

To get the audio clip we use the method getAudioclip( ), from this method we can get the audio file from the workspace.

```
                public void init()
 {
                    setBackground(Color.BLACK);
                    setFont(textFont);
                    setForeground(Color.blue);
                    Pic = getImage(getDocumentBase(), "mzl.ucenpfct.jpg");
                    Win = getImage(getDocumentBase(), "winner.jpg");
                    redPic = getImage(getDocumentBase(), "redpic.gif");
                    greenPic = getImage(getDocumentBase(), "greenpic.gif");
                    moon = getImage(getDocumentBase(), "moon.jpg");
                    grass = getImage(getDocumentBase(), "grass.png");
                    pond = getImage(getDocumentBase(), "pond.png");
                    rst = getImage(getDocumentBase(), "reset.png");
                    ac = getAudioClip(getCodeBase(), "welcome.wav");
                    jump = getAudioClip(getCodeBase(), "frog.wav");
                    ac.play();
        }
```

// ****the action of the paint function depends on the mode*****

Paint function use to paint the window. In this game we make three level of game and relate with the modes. We

have easy, medium and hard level. each level have its own mode so when we click easy mode then the paint function paint the window for easy level, if the mode is medium then the paint function paint the window for medium level and so on.

```
        public void lvlEasy ( )
{
                        numRed = 3; // number of red frogs
                        numGreen = 3; // number of green frogs
                        numPlaces = numGreen + numRed + 1;
            frogs = new ArrayList<Integer> ( );
            }
public void lvlMedium ( )
{
                        numRed = 5; // number of red frogs
                        numGreen = 5; // number of green frogs
                        numPlaces = numGreen + numRed + 1;
                        frogs = new ArrayList<Integer> ( );
            }
public void lvlHard ( )
 {
                        numRed = 8; // number of red frogs
                        numGreen = 8; // number of green frogs
                        numPlaces = numGreen + numRed + 1;
                        frogs = new ArrayList<Integer> ( );
}
public void paint(Graphics g)
```

Display the rules of the game

In this section we display the rule of the game and instructions which followed by the users and in this section we display the rule auto salve rule from which user can also get gaudiness

```
                        public void showRules(Graphics g)
```

Move a frog from a given position to a given position

In this section we use the swap function to move the position from source to destination so we make the function void movefrog( ).

```
            public void moveFrog(int source, int dest)
```

Decide whether the mouse has been clicked over a position

In the mouse down function we set the position where the mouse clicked. We set the click position easy, medium and hard levels and others buttons which are used in the game.

```
            public boolean mouseDown(Event event, int x, int y)
```

Working Principle of the Games

Description of Project

In this chapter, I mainly introduce the project I designed, and some regulations and principles are also explained so as to make individuals understand clearly.

This game is developed the mind ability of child. It is simple for play rule .it is the basic level of game for child to improve the mental ability. This game have three levels Easy, Medium and hard. In which game some red and green frog sitting on the stands. The aim of the game is to swap the frogs over using the simple set of rules. The red frogs only can move to the right and the green frogs only can move to the left sides.

Figure 3: Game display.

Level of Game

Easy level of game

In the easy level total number of frogs is six, three red and three are green. Red frogs are sitting the left side and green frogs are sitting the right sides at stands.

Red frogs can only move to the right side and green frogs can only move to the left side. Coding of the easy level is given.

```
publicvoid lvlEasy ( )
        {
                numRed = 3; // number of red frogs
                numGreen = 3; // number of green frogs
                numPlaces = numGreen + numRed + 1;// total number of places
                frogs = new ArrayList<Integer> ( );
        }
```

Figure 5.2 of the easy level given below.

Figure 4: Easy level of game leaping frog.

Medium level of game

In the easy level total number of frogs is ten, five red and five are green. Red frogs are sitting the left side and green frogs are sitting the right sides at stands.

Red frogs can only move to the right side and green frogs can only move to the left side. Coding of the medium level is given

```
        publicvoid lvlMedium ( )
        {
                numRed = 5; // number of red frogs
                numGreen = 5; // number of green frogs
                numPlaces = numGreen + numRed + 1;// total number of places
```



```
        frogs = new ArrayList<Integer> ( );
    }
```

Figure 5: Medium level of game leaping frog.

Hard level of game

In the easy level total number of frogs is twenty, ten red and ten are green. Red frogs are sitting the left side and green frogs are sitting the right sides at stands.

Red frogs can only move to the right side and green frogs can only move to the left side. Coding of the hard level is given

```
    publicvoid lvlHard ( )
    {
        numRed = 10; // number of red frogs
        numGreen = 10; // number of green frogs
        numPlaces = numGreen + numRed + 1;
        frogs = new ArrayList<Integer> ( );
    }
```

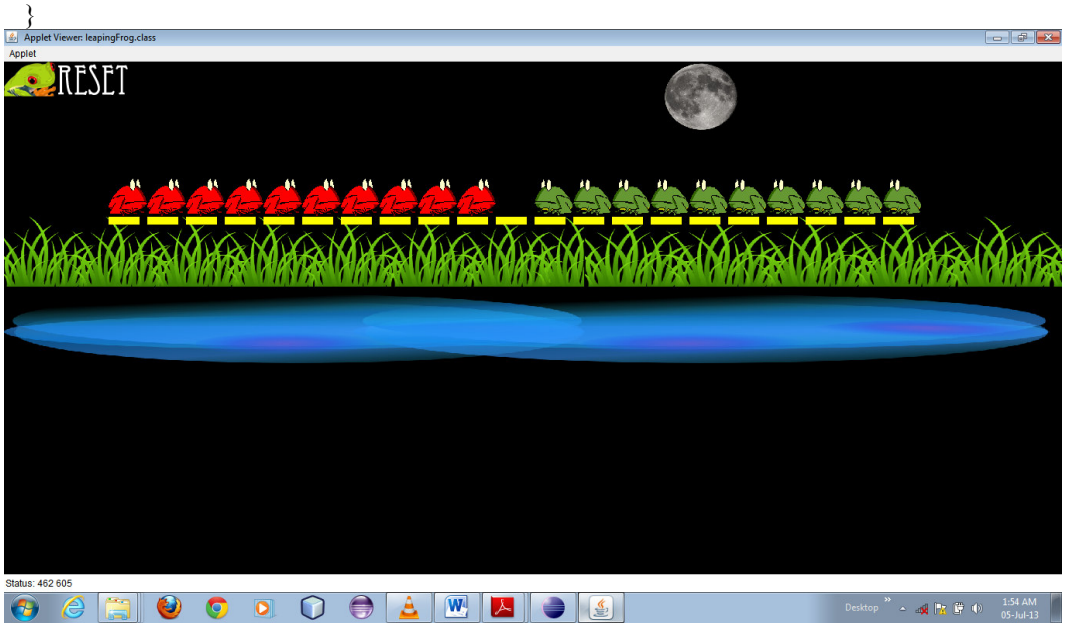

Figure 6: Hard level of game leaping frog.

Conclusion

Taking into account of all the aspects which are mentioned in previous chapters, the conclusion can be drawn is obviously that I have successfully developed a Leaping Frog game and tested it on the emulator. Actually, the ultimate purpose of my final thesis is to get more familiar with Java programming, build a game by using Java, and utilize it to implement a Leaping Frog game.

Challenges

In this section, I would demonstrate some challenges and difficulties I faced, and the trouble-shootings with regard to my study. The critical part of my thesis is to figure out how to build a user interface.

Eventually, this Leaping Frog game can be displayed successfully. there are some challenges related to my java tool that my application cannot be displayed on my system screen, which means there is an error message demonstrated onto the screen. I guess the main reason is that my system is not that some class packages are missing in my java toolkit so I download these class packages then my game run successfully.

Anyway, I think this huge project makes me more familiar with Java programming and helps me to grasp an essential skill for my Master degree and even my career although there are some troubles with regard to it.

Future Development

Last but not least, I need to discuss some prospects and further work which should be done in the near and far future. Nowadays, Java game has become more and more popular in more than one aspect. If it is possible, I guess it is necessary to apply this Leaping Frog game into multi-player mode and with the best improvement mental ability of children.

I believe this type of mobile phone games will have a wide marketing perspective in the near future.

References

[1] Hecker, Chris. "Physics Part 1: The Next Frontier." Game Developers Magazine. Oct.-Nov. 1996: 12-20.
[2] Hecker, Chris. "Physics Part 2: Angular Effects." Game Developers Magazine. Dec.-Jan. 1996: 14-22.
[3] Hecker, Chris. "Physics Part 3: Collision Response." Game Developers Magazine. Feb.-Mar. 1997: 11-18.
[4] Hecker, Chris. "Physics Part 4: The Third Dimension." Game Developers Magazine. June 1997: 15-26.
[5] Lovbjerg, M. 2002. “Improving particle swarm optimization by hybridization of stochastic search heuristics and self-organized criticality”. Denmark: Aarhus University.
[6] Holland, J. 1975. *Adaptation in Natural and Artificial Systems*, Ann Arbor, MI: University of Michigan Press.
[7] Eusuff, M. M. and Lansey, K. E. 2003. Optimization of water distribution network design using the shuffled frog leaping algorithm. *J. Water Resour. Planning Mgmt*, 129: 210–225.
[8] Elbeltagi, E., Hegazy, T. and Grierson, D. 2005. Comparison among five evolutionary-based optimization algorithms. *J. Adv. Engng. Informatics*, 19: 43–53.
[9] Kennedy, J. and Eberhart, R. 1995. “Particle swarm optimization”. In *Proceedings IEEE International Conference on Neural Networks*, 1942–1948. Piscataway, NJ: IEEE Service Center.
[10] Feng, C., Liu, L. and Burns, S. 1997. Using genetic algorithms to solve construction time – cost trade-off problems. *J. Comput. Civil Engng*, 11: 184–189.
[11] Duan, Q. Y., Gupta, V. K. and Sorooshian, S. 1993. Shuffled complex evolution approach for effective and efficient global minimization. *J. Optimization Theory Appns*, 76: 502–521.