[ICITAIC-2019]

ISSN 2348 – 8034
Impact Factor- 5.070

GLOBAL JOURNAL OF ENGINEERING SCIENCE AND RESEARCHES

SNAKE GAME

Mugdhama Patil[1], R.Neha[2], Pravachana Mallapu[3], Mr.Rajasekhar Sastry[4], Dr.B.V Ramana Murthy[5] and Mr. C Kishor Kumar Reddy[6]
[1,2,3,4,5,6]Stanley College of Engineering and Technology for Women, Hyderabad

ABSTRACT

Video game development is the process of creating a video game. The effort is undertaken by a game developer, who may range from a single person to an international team dispersed across the globe. A video game can be developed in many different languages, one of them being Java. Java is one of the best languages in which one can develop a game. Coding in Java allows one to make games that will run on all the desktop operating systems like Windows, OSX and Linux but also it is the native language for making games for all Android devices. A game can be developed in Java by using IDE's. Eclipse is an Integrated Development Environment for Java. We can use Eclipse for coding and executing the games. The Snake Game is one of the games that can be implemented in Java. Here the snake is moved around by the player, for it to reach the apple. It takes us one step forward to learning more about the computer languages.

Keywords: *Video game development, Java, Eclipse, Snake Game.*

I. INTRODUCTION

Java language is a general-purpose programming language that is concurrent, class-based, object-oriented, and specifically designed to have as few implementation dependencies as possible. The applications of Java are usually compiled to byte code that can run on any Java virtual machine (JVM) regardless of computer architecture. Java programming language was developed by James Gosling, at Sun Microsystems and was released in the year 1995 as a core component of Sun Microsystems' Java platform.

Video game development is the process of making a video game. You take an idea or a concept for a game, and you develop, program, engineer, render, record, mix, produce, test, etc. until you have a full-fledged game. It's grown quite a bit in the past ten years. There are a number of reasons for this. Some of them are: the continued growth of the gaming industry. As video games become more accessible and gain a huge following, more people consider participating in what goes on behind the scenes. The popularity boom means more opportunities, as college educators rush to satisfy the demand. It also means that a lot of competition as there is now in a pool of thousands of like-minded students hoping to nap the "golden ticket" that is a degree in game development. The first video games made were non-commercial, and were developed in the 1960s. They required mainframe computers to run and were not available to the public. Commercial game development began in the 1970s with the advent of consoles and early home computers like the Apple I. Due to low costs and capabilities of computers, a programmer could develop a full game. Approaching the 21st century, the increasing computer processing power and heightened consumer expectations made it difficult for a single person to produce a mainstream console or PC game. The average cost of producing a triple-A video game slowly rose from US$1–4 million in 2000 to over $5 million in 2006, then to over $20 million by 2010.
Snake game is the common name for a video game where the player manoeuvres a line which grows in length, with the line itself being a primary obstacle. This concept was originated in the 1976 arcade, and the ease of implementing Snake game has led to hundreds of versions (some of which have the word snake or worm in the title) for many platforms. After a variant was loaded on Nokia mobile phones in 1998, there was a resurgence of interest in the snake concept as it found a larger audience.

The rest of the thesis is organized as follows:
Chapter-2 depicts the relevant work on game development and snake game.
Chapter-3 proposes the Snake Game development.

Chapter-4 discusses the procedure and implementation
Chapter-5 concludes the thesis followed by references

II. LITERATURE SURVEY

Java is a programming language with unique features. Java has been the foundation for developing and delivery of embedded and mobile applications and java game programming as well as Web content and enterprise software. Java has up to 9 million developers worldwide. From laptops and PCs to gaming consoles and supercomputers, Java is just about everywhere.

Object-oriented programming concepts (which includes abstraction, containment, inheritance, encapsulation, and polymorphism).The reasons why Java is a big scale platform independent language which can be run on all OS like Mac, Windows, and Unix. Because of its Manner and nature of Java Virtual Machine. Understanding the Java collection framework. In-depth knowledge of the data types and few of the java language classes like String, System, Math etc.
There are four main OOP concepts in Java. These are:

- Abstraction. Abstraction means using simple things to represent complexity. We all know how to turn a TV on, but we don't need to know how it works in order to enjoy it. In Java language, abstraction means simple things like objects, classes,and variables represent more complex underlying code and data. This is important because it lets avoid repeating the same work several times.

- Encapsulation. This is the practice of keeping fields within a class private, then providing access to them via public methods. It's a protective barrier that keeps the data and code secure within the class itself. This way, we can re-use objects like code components or variables without allowing open access to the data system-wide.

- Inheritance. This is a special feature of Object Oriented Programming in Java. It lets programmers create new classes that share some of the attributes of existing classes. This lets us build on the previous work without reinventing the wheel.

- Polymorphism. This Java OOP concept lets programmers use the same word to imply different things in different contexts. One of the forms polymorphism in Java is method overloading. That's when different meanings are implied by the code itself. The other form is method overriding. Mastering Java is all about how one can put theory into practice. If you have to get the programming of Java perfectly, one needs to try out different types of logic exercises (for example finding all the prime numbers between 1 to 2000 and creating a Fibonacci series and also computing number factorials and much more of such kind) and file input/output exercises such as listing files, reading and displaying files on console, creating file with content). String manipulation exercises such as parsing numbers from strings, replacing part of the string and building number pyramids or creating 2 player text-based games can also be done. One can write Java programming language programs and software and run it on any platform. Users can create programs running within a web browser and accessing is available in the web services apart from developing server-side applications for online polls, commerce firms. Customized apps can also be created using Java programming and one can write efficient applications for every type of electronic device they need including wireless modules and mobile phones. Developers can sharpen their skills and further learn Java language features on how to come up by reading the Java web developer site. There are so many visual education tools such as BlueJ and Alice to impart training in Java programming language to developers. Java has significant language features which offer benefits to users. Platform independence means compilers do not have to produce native object code for platforms and instead come up with byte code instructions for Java Virtual Machine. Java is easy to master too. And there are numerous classes and methods in the programming language which have to be learned. Input/output classes are there to read and write data from numerous sources and networking permits communication across computers online or LAN can also be used. Platform independent GUI applications are created through Java's Abstract Window Toolkit. Java Applet is a special class that lets one to come up with downloadable simple Java programs that can be run on client browsers. Java can be used anywhere and everywhere. Education, embedded systems, application programming, and simulation are few of the many areas where Java language basics can be applied. Areas of application also include network apps, WWW Applets, Cross-platform app development and many more. Java code is a programming language as well as a virtual machine tool and API specification. One of the biggest advantages of this language is making it easy for use across numerous settings and its high level of security and safety. It performs legal object field access and only legal data conversions. All up code parameter types undergo checking so that all of them are legal. Another property of Java development which lends itself to multiple platforms is its top performance. Java Programming Language basics environment compiles the byte code into native machine code during runtime.

Many different games can be developed using Java. Few of them are-

- Puzzle
- Snake
- Breakout
- Tetris
- Pacman
- Space Invaders
- Minesweeper
- Sokoban

III. SNAKE GAME

We used Eclipse as the IDE to develop the Snake Game. This code of the game is written in a package which six consists of classes.

1. BoardPanel Class
2. SidePanel Class
3. Direction Class
4. Clock Class
5. TileType Class
6. Snake Game

In the SidePanel Class, the Statics and Score and the Controls are to be displayed. The dimensions of the side panel are set. Background is set to white and the font colour is set to black. Headings and Subheadings are displayed using the function setFont(). Object of Snake Game Class is created here for the statistics. Display the Total Score, Total Apples Eaten and the Apple Score

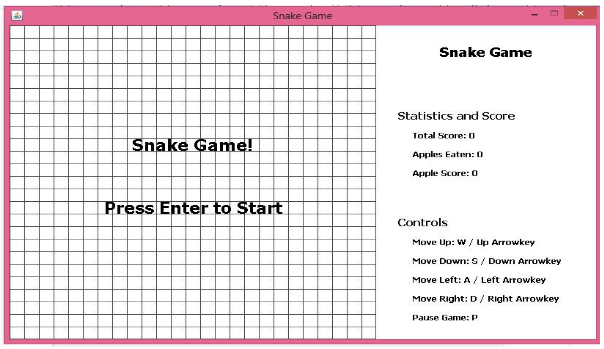
The Clock class is responsible for tracking the number of cycles that have been elapsed over time. This class pauses the game when commanded to and send s value to the called classes. It also resets the game when it comes to an end.

The Direction class is used to determine which way the snake is moving.
The TileType class is represents the different types of tiles that can be displayed on the screen. They are apple, snake's head and the snake's body.

TheBoardPanel Class is responsible for managing and displaying the contents of the game board. First, the background of the game is set to white. A grid is drawn on the board. This makes it easier to see where we are in relation with the apple. The messages displayed at the beginning of the game and at the end of the game are written is coded here and are made to be written in the middle of the screen. The snake and apple are created in the colours green and red respectively. The head of the snake is different compared to its body. The head has eyes which should always face the direction in which the snake is moving. Vertical lines represent that its facing north or south. Horizontal lines indicate that it is facing east or west. The apple is set to and random direction and the snake is set to move in different direction

The Snake Game Class is responsible for handling much of the game's logic. Objects of all the other classes are created here. The logic when the snake hits the border or its own body is also written here. If the snake collides with a fruit, increment the number of fruits eaten , update the score and spawn a new fruit. If snake collides with its body or the wall, flag that the game is over and pause the game. If no collision occurred, decrement the number of points that the next fruit will give us if it's high enough. When the snake collides with the fruit increase the length of the snake. When the game terminates reset the score and statics.

IV. RESULTS AND DISCUSSION


Snake Game

Snake Game!

Press Enter to Start

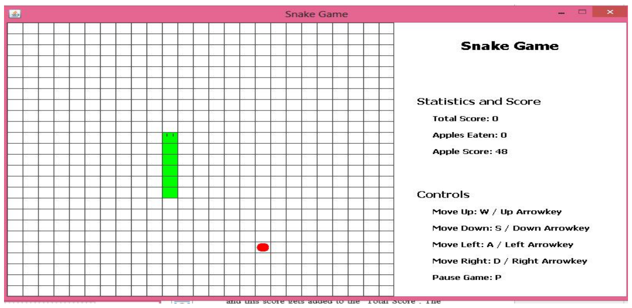
Snake Game

Statistics and Score
Total Score: 0
Apples Eaten: 0
Apple Score: 0

Controls
Move Up: W / Up Arrowkey
Move Down: S / Down Arrowkey
Move Left: A / Left Arrowkey
Move Right: D / Right Arrowkey
Pause Game: P

Fig 1. Start window

The following picture is the result of project. When the program is run, the game window is displayed. The name of the game is displayed on the top. The game consist of a game panel and a side panel. The game panel is where the actual game takes place and is in the form of a grid. The side panel tells you about the statistics and the controls; the statics show the 'Total Score', 'Total Apples Eaten' and 'Apple Score'. It controls tell you which keys to use while playing the game .i.e. to move up use W or up arrow key, to move down use S or down arrow key, to move left use A or left arrow key , to move right use D or right arrow key. To start the game 'Enter' key must be pressed.

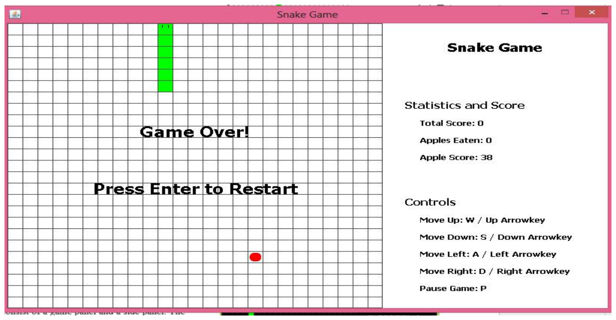

Snake Game

Statistics and Score
Total Score: 0
Apples Eaten: 0
Apple Score: 48

Controls
Move Up: W / Up Arrowkey
Move Down: S / Down Arrowkey
Move Left: A / Left Arrowkey
Move Right: D / Right Arrowkey
Pause Game: P

Fig 2. Running Game

The snake is in green colour while the fruit is in red. The snake has to move around and eat the fruit, to get points. Every fruit has a score; it starts from 100 and decreases with time. The 'Fruit Score' at the time when the fruit gets eaten, is the score achieved and this score gets added to the 'Total Score'. The body of the snake is increased when it eats the fruit. The 'Fruit Score' can go to minimum of 10 points.


Game Over!

Press Enter to Restart

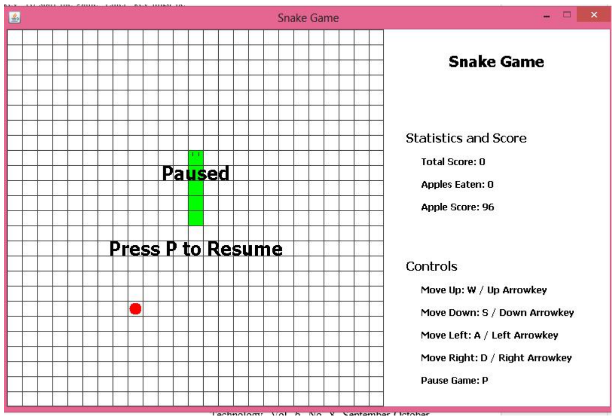
Snake Game

Statistics and Score
Total Score: 0
Apples Eaten: 0
Apple Score: 38

Controls
Move Up: W / Up Arrowkey
Move Down: S / Down Arrowkey
Move Left: A / Left Arrowkey
Move Right: D / Right Arrowkey
Pause Game: P

Fig 3. Game Over window

When the snake collides with the boundaries or when it collides with its own body, the game is over and the following message is displayed. We can press enter to restart the game.


Paused

Press P to Resume

Snake Game

Statistics and Score
Total Score: 0
Apples Eaten: 0
Apple Score: 96

Controls
Move Up: W / Up Arrowkey
Move Down: S / Down Arrowkey
Move Left: A / Left Arrowkey
Move Right: D / Right Arrowkey
Pause Game: P

Fig 3. Paused Window

The game can be paused by using the 'P' key and can be resumed using the same.

V. CONCLUSION

In this paper, the implementation of the infamous "Snake Game" is described. It discusses the steps to be taken while programming this game. The language used to write the game is described. The result has been shown with the help of pictures. This Snake Game has its own features and is interesting to play. Today games are rarely made using programming languages but the result is effective and helps one to learn more in depth about the programming language. It is also a multi-platform unique programming language with inbuilt security which helps prevent hacking.

VI. REFERENCES

1. *Saloni Jain, "Developing Games in Java for Beginners", International Journal for Research in Applied Science& Engineering Technology (IJRASET), Volume 4 Issue III, March 2016.*
2. *Bian Wu and Alf Inge Wang," A Guideline for Game Development-Based Learning: A Literature Review",Hindawi Publishing Corporation International Journal of Computer Games Technology Volume 2012, Article ID 103710*
3. *S. Bjork, S. Lundgren, J. Holopainen, Game desig patterns, in:Lecture Note of the Game Design track of Game Developers Conference 2003, March 4–8, San Jose, CA, USA, 2003*
4. *S. Bjork, S. Lundgren, J. Holopainen, Game design patterns, in: Proceedings of Digital Games Research Conference 2003, Nov. 4– 6, Utrecht, The Netherlands, 2003.*
5. *Sergii Shepelenko," Creating a 2D Platform Game", University of Tartu.*
6. *Stephen Tang and Martin Hanneghan "Game Content Model: An Ontology for Documenting Serious Game Design".*
7. *David Brackeen, Bret Barker - Developing Games in Java 1 st edition.*
8. *Lakshmi Prayaga, "Introductory Game Programming Instruction with OOP", Journal Of Object Technology, Vol. 6, No. 8, September- October 2007.*
9. *Edgington, J., Leutenegger, S., (2007), A games First Approach to Teaching Introductory Programming, SIGCSE'07, March 7-10, 2007, Covington, Kentucky, USA.*
10. *Argent, L, Depper, B.; Fajardo, R.; Gjertson S.; Leutenegger, S., T.; Lopez, M., A.; and Rutenbeck , J.(2006), "Building a Game Development Program," Computer, vol. 39, no. 6, pp. 52-60.*
11. *Andrew Davison, "Games Programming with Java and Java 3D", 14th January 2003.*
12. *Hutchinson, R. , 2001. "The Evolution of Performance on the Java Platform: A Panel Discussion".*
13. *Marner, J. 2002. "Evaluating Java for Game Development", Dept. of Computer Science, Univ. of Copenhagen, Denmark, March.*
14. *Croft, D.W. 2004. Advanced Java Game Programming, Apress, April.*
15. *Eckel, B. 2006. Thinking in Java, Prentice Hall, 4th ed., February*
16. *Davison, A. 2005. Killer Game Programming in Java, O'Reilly Media, May*
17. *Twilleager, D., Kesselman, J., Goldberg, A., Petersen, D., Soto, J.C., and Melissinos, C. 2004. "Java Technologies For Games", ACM Computers in Entertainment, Vol. 2, No. 2, April.*