RESEARCHERID
THOMSON REUTERS

[ICITAIC-2019]

ISSN 2348 – 8034
Impact Factor- 5.070

GLOBAL JOURNAL OF ENGINEERING SCIENCE AND RESEARCHES

BRICK BREAKOUT

Anjana Kamana[*1], Melissa Mary Smith[2], Nikhitha Beecharaju[3], Mr.Rajasekhar Sastry[4], Dr.B V RamanaMurthy[5] & Mr.C Kishor Kumar Reddy[6]

[*1,2,3,4,5&6]Stanley College of Engineering and Technology for Women, Hyderabad

ABSTRACT

A game design document is the blueprint from which a game is developed. The research design is a case study which consists of semi-structured interviews, questionnaires and observation used to generate the data. Java is more accepted in the game development community these days, and with good commercial-quality Java games are on the market, it will become a definitive choice. Practical Java Game Programming identifies the technological path developers need to Introduction to Java Programming with Games follows a spiral approach to introduce concepts and enable them to write game programs. The code is designed for games and are generally transformable and can be prolonged for a longer duration. Games are generally separated by the different versions that are devised by the researchers. Scientists delineate games on the basis of analysing the divergent patterns in the history of a particular game that came into being. Brick breakout is a classic game which is developed using java. The object of brick breakout is to break the bricks that are distributed around the top of the game screen. The bricks are broken after coming in contact with a ball that bounces around the screen with the help of a paddle.

Keywords: *Game design, java programming, code design, brick breakout, paddle*.

I. INTRODUCTION

James Gosling, Mike Sheridan, and Patrick Naughton started this Java language project work in June 1991. Java was originally designed for television interaction, but it was too advanced for the digital cable television industry at that time. Java Programming Language was written by James Gosling at Sun Microsystems for the Sun Microsystems released in 1995 as core component Microsystems' in Java platform. Sun Microsystems released their first public implementation as Java 1.0 in 1996.Java is a general purpose computer programming language that is class based, object oriented. It is planned to let the application developers "Write once, Run anywhere" (WORA). This means Java code can run on all platforms that support java and there is no need of recompilation. Java is typically compiled to byte code that can run on any Java virtual machine(JVM) regardless of architecture of java. Java is derived from Smalltalk, is similar to C and C++.Java is portable so it is preferred. Java is currently used in many different applications including for developing Android apps, Java web applications, software tools, etc. Java can also be used to create games.

Java follows some principles which makes it mostly used and preferred programming language to write and understand. The five principles that are to be followed while creating a java program are: -

- It must be "simple, object-oriented, and familiar".
- It must be "robust and secure".
- It must be "architecture-neutral and portable".
- It must execute with "high performance".

It must be "interpreted, threaded, and dynamic".

For a 'game design' it can mean "emotional engineering" or "largely communication" whilst for the other "everything that goes into a game is more or less game design". On a more formal matter, many designers have gone extra miles to explicate their conceptions on game design. Due to the young age on the field, it is not so rare to see these design guidebooks written by the industry actors cited in academic papers. Some of the books can be considered as part of the "canon" of the game studies. Katie Salen's& Eric Zimmerman's book Rules of Play: Game Design Fundamentals discussing the different schemes of game designs (rules, play, culture) is perhaps one of these. Written for the designers, the book is widely4used by the young field of researchers – and often perhaps even

ISSN 2348 – 8034
Impact Factor- 5.070

misunderstood by their search community due to the differing knowledge needs of the industry and the academics (inspiration vs. foundation). Conceivably one of their most cited original notion is the notion of "second order design". This is to illustrate the indirect nature of game designing as computer mediated experiences: "game designis a second-order design problem. A game designer designs the rules of the game directly but designs the player's experience indirectly". However, the lack of theoretical exposure to other design rules might lead to the overemphasis of the "specialnature" of a design domain close to the examiner. In a discussion over Twitter in 2012 (Figure 1), Eric Zimmerman did retrospectively contemplate that "Perhaps design is always about second-order problems."

In this game, we have one paddle, one ball and 30 bricks. An image of ball has been created, a paddle and a brick in Inkscape. A timer is created for game cycle. They do not work with angles; It simply changes directions Top, bottom, left and right. It is inspired by the pybreakout game. This brick breaker game was developed in PyGame library by Nathan Dawson.

The game consists of seven files namely Commons.java, Sprite.java, Ball.java, Paddle.java, Brick.java, Board.java, and Breakout.java. In this game the player moves the bottom bar (paddle) displayed on the screen by using the buttons on the keyboard to deflect a bouncing ball to smash the bricks on the upper part of screen. There are several rules in the movements of the ball and how the score is calculated.

II. LITERATURE SURVEY

The first step is to write a code which puts the various pieces on the playing board. Itprobably makes sense to implement procedure to run (which drives the game) as two methodcalls: one that sets up the game and the other that plays it. The most important part of the setup consists of creating rows of bricks on the top of the game. The number of dimensions andspacing of the bricks, and the distance from the top of the window to the first line of bricks, are specified using some named constants given in class Breakout.

The next step is creating a paddle. You will need to reference the paddle often.Be careful that the paddle stays completely on the board even if the mouse moves off the board. The next step is an interesting part. In order to make brick breaker or breakout into a real game, you have to be able to tell when the ball collides with another object in the window. However, the ballis not a single point. It occupies physical area, so it may collide with something on the screen even though its centre does not. The easiest thing to do is typically of the simplifying assumptions made in real computer games are to check a few carefully chosen points on the outside of the ball and see whether any of those points has collided with anything. As soon as you find something at one of those points (other than the ball of course) you can declare that the ball has collided with that object.

Java should be fast enough for most games, so where is the catch? There are some reasons:

- Many game developers lack expertise in java.
- Lack of good game development frameworks is common.
- Programmers don't want to accept Java as a games programming language. Most only accept C++ as that?
- No support for game consoles (though the PC market still exists)

In terms of machine speed, Java beats C++ in several scientific computing benchmarks these days. You'll write pathological code in either language that performs badly if you would like to, however over-all, they're at par and are for an extended time. In terms of memory usage, Java will have some over-head. Hello World could be a 4K program in java. However, that overhead is pretty nonsensical in today's multi GB systems. Finally, Java will have additional of a start-up time. i might not advocate mistreatment Java for brief run-time utilities like UNIX system program line commands. In those cases, start-up can dominate your performance. In an exceedingly game but, it's fairly insignificant. Properly written Java game code doesn't suffer gigahertz pauses. a bit like C/C++ code, it will need some active memory management however to not the amount C/C++ will. As long as you retain your memory usage to either long lived objects (persist for a complete level or game) and really short lived objects (vectors and such, passed around and quickly destroyed when calculation) gigahertz mustn't be an apparent issue.

ISSN 2348 – 8034
Impact Factor- 5.070

Java is nice for business logic, servers, and platform freelance code that has got to run faithfully. There area unit many factors why Java is not typically employed in games:

- Bounds checking safety mechanisms (marginal performance distinction these days).
- Having to convert between C++ knowledge structures and Java knowledge structures (can't simply copy memory between buffers)
- Several of the books and tutorials follow the gang thus it's arduous to search out non-C++ game dev info
- The core graphics libraries (DirectX and OpenGL) and plenty of off-the-rack engines area unit C/C++ primarily based
- several games try and run as quick as potential so that they will add additional visually appealing options. It's rough to figure with C++ libraries from bytecode languages like Java (writing a JNI layer) and .net

(Lots of marshalling/unmarshalling, api/structure attributes). So it adds quite a bit of work for little benefit.

Game developers wish to be getting ready to the metal and infrequently can write their tight inner loops in assembly. Java does not provide identical level of potential performance, each in terms of consistent speed or memory use.

Game programmers conjointly unremarkably use Java, as a result of Java supports multithreading and sockets. Multithreading uses less memory and makes the foremost of obtainable mainframe, while not block the user out once serious processes ar running within the background. Sockets facilitate in building multiplayer games. Plus, Java runs on a virtual machine, thus your game is easier to distribute.

In our game, we've got one paddle, one ball and thirty bricks. I actually have created a picture for a ball, paddle and a brick over Inkscape. We have a tendency to use a timer to form a game cycle. we have a tendency to don't work with angles, we have a tendency to merely amendment directions. Top, bottom, left and right. I used to be galvanized by the pybreakout game. it absolutely was developed in PyGame library by Nathan Dawson.

First, we'd like to determine on the categories and also the knowledge and ways in them. we have a tendency to will demonstrate Associate in Nursing approach that was 1st projected by Abbott in 1983. The approach states to identify the categories from the matter description of the matter. Staring at the outline, we will produce the jailbreak category wherever the most technique is written. We'll conjointly include classes for the most participants of the game: Ball, Brick, Paddle, and Player. We have a tendency to will create 2 enumeration types: Ball Color and Speed. There, we'll store the possibilities for the ball color and ball speed. We'll have to be compelled to produce 2 extra categories that related to Java: BreakoutFrame and BreakoutPanel. These are the window and also the drawing panel for our application, severally. Lastly, we'll produce a Breakout Shape category that will be the taxon of all the shapes which will be displayed. Since drawing the paddle, ball, and brick are similar, it's cheap to form one taxon.

In jailbreak, the initial configuration of the planet seems as shown on the correct. the coloured rectangles within the high a part of the screen are bricks, and also the slightly larger parallelogram at all-time low is that the paddle. The paddle is in a very mounted position within the vertical dimension, however moves back and forth across the screen in conjunction with the mouse till it reaches the sting of its house. An entire game consists of 3 turns.

On every flip, a ball is launched from the middle of the window toward very cheap of the screen at a random angle. That ball bounces off the paddle and also the walls of the planet, in accordance with the physical principle usually expressed as "the angle of incidence equals the angle of reflection" (which seems to be terribly simple to implement as mentioned later during this handout). Thus, when 2 bounces—one off the paddle and one off the proper wall—the ball might need the flight shown within the second diagram. (Note that the dotted line is there only to show the ball's path and won't appear on the screen.)

Now comes the interesting part. In order to create jailbreak into a true game, you've got to be ready to tell whether or not the ball is colliding with another object within the window. As scientists typically do, it helps to start by creating a simplifying assumption and so reposeful that assumption later. Suppose the ball were one purpose instead of a circle.

ISSN 2348 – 8034
Impact Factor- 5.070

The advantage of being outside the ball which implies that get Element At can't come back the ball itself—but yet shut enough to create it seem that collisions have occurred. Thus, for every of those four points, you wish to:

1. Decision get Element At on it location to examine whether or not something is there.
2. If the worth you retreat to isn't null, then you wish look no farther and may take that worth because the GObject with that the collision occurred.
3. If get a element that returns null for a specific corner, go on and try the next corner.
4. If you get through all four corners while not finding a collision, then no collision exists.

From here, the sole remaining issue you wish to try to is decide what to try to once a collision happens. There are only two possibilities. First, the article you retreat to could be the paddle, which you can test by checking.
If it's the paddle, you need to bounce the ball so that it starts traveling up. If it isn't the paddle, the only other thing it might be is a brick, since those are the only other objects in the world. Once again, you wish to cause a bounce within the vertical direction, but you also need to take the brick away. To do so, all you wish to try to take away it from the screen by job the remove methodology.

III. PRODUCER FOR BRICK BREAKOUT

The game which is developed by us consist of following thesis where we have, Chapter 1: Start Page, Chapter 2: Game and Chapter 3: Game over.

1. The start page was designed according to the user friendly manner which consists of single player and at one go the user can start the game and move into the created segmented.
A ball travels across the screen, bouncing off the top and side walls of the screen. When a brick is hit, the ball bounces away and the brick is destroyed. The player loses a turn when the ball touches the bottom of the screen. To prevent this from happening, the player has a movable paddle to bounce the ball upward, keeping it in play.

Figure 1: ***Start Screen***

It is very joyful game which can be played by any age group and the Start page has a START button to start the game.

2. The second window consists of the actual game to be played, it consists of a paddle where the ball rest in the start and later on paddle helps us to hit all the bricks present in the window. The concept of the game is to break all the bricks with the help of paddle and the ball, once the ball hits the boundary excluding the

ISSN 2348 – 8034
Impact Factor- 5.070

paddle then the game gets over. Hence we've to start the game again to start it all over again. Once you break all the bricks on the screen you get a pop up VICTORY or else GAME OVER.

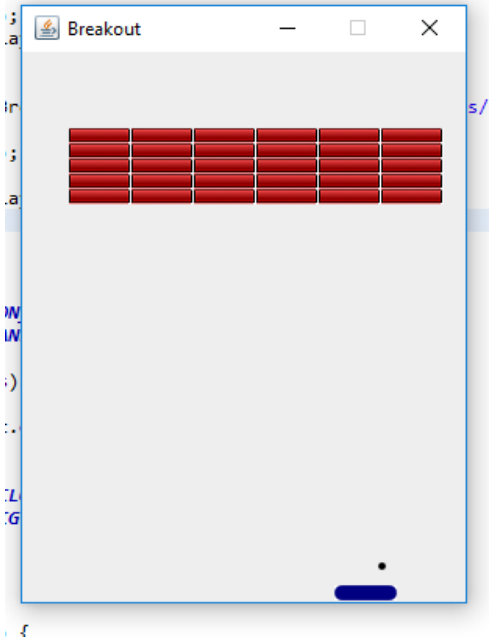

Figure 2: Game Screen

3. The last window that will appear is the Game over or the Victory sayings. Once you complete breaking all the bricks that is, you won the game and can play again freshly. If not, it appears a Game Over popup and give out the score. Score is the number of bricks that you broke. Along with the score you get a saying telling Game Over and you have to start the game again and start playing from the beginning.
The game was published in 1978, but with only six rows of bricks, and the player is given five turns to clear two walls instead of three. In the Breakthru variant, the ball does not bounce off of the bricks, but continues through them until it hits the wall.

ISSN 2348 – 8034
Impact Factor- 5.070

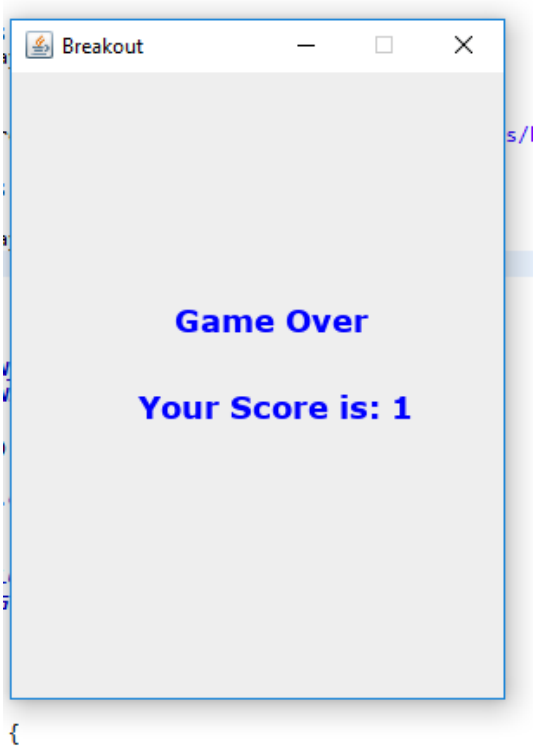

Figure 3: Game Over Screen

This game is being run through the eclipse 'run'. After we run the code or the program written, it will appear the Start screen and we can start playing the game which is user friendly and any age group can play.

IV. RESULT AND DISCUSSION

The output screen that will be displayed after the game is finished is the result to our developed game. The output that is displayed is the result to the developed program. The result shows that if a user completes the game completely the output pops out as VICTORY but if the user could not break all the bricks the output screen displays the Game Over pop out. The victory screen looks like

ISSN 2348 – 8034
Impact Factor- 5.070

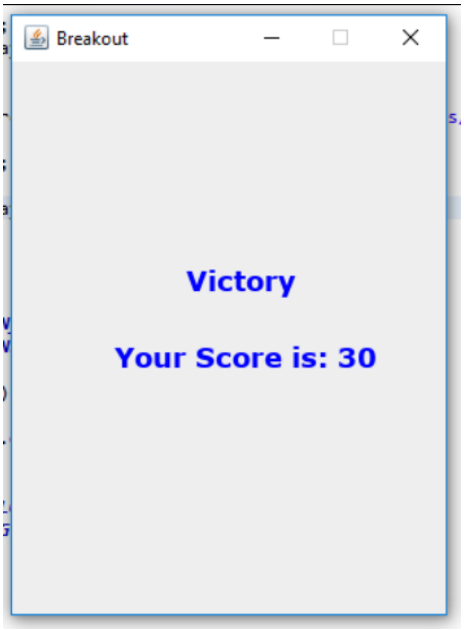

Figure 4: Victory Screen

This is how we developed the Brick Break Out game with the help of java programming language. This is how the input process takes place and gives the output in the above form.

V. CONCLUSION

In this survey paper it is shown how java is used in game development and in other applications. We have discussed about how brick breakout game is developed in java.

REFERENCES

1. *P. Schliep, "Usability of Error Messages for Introductory Students," Scholarly Horizons: University of Minnesota, Morris Undergraduate Journal, vol. 2, no. 2 April 2015.*
2. *E. Lahtinen, K. Ala-Mutka, and H-M Järvinen, "A study of the difficulties of novice programmers," ACM SIGCSE Bulletin. vol. 37. no. 3. ACM, 2005.*
3. *Milne and G Rowe. "Difficulties in learning and teaching programming—views of students and tutors." in Education and Information technologies vol 7, no. 1, pp. 55-66, 2002.*
4. *Korhonen and L Malmi, "Taxonomy of visual algorithm simulation exercises," in Proceedings of the Third Program Visualization Workshop. Warwick, UK, pp. 118-125, 2004.*
5. *M. Hristova, A. Misra, M.Rutter, & R. Mercuri. "Identifying and correcting Java programming errors for introductory computer science students" in ACM SIGCSE Bulletin, vol. 35, No. 1, pp. 153-156. ACM, 2003.*
6. *T. Boyle, C. Bradley, P. Chalk, R. Jones, & P. Pickard, "Using blended learning to improve student success rates in learning to program" in Journal of educational Media, vol 28 (2-3), pp. 165-178, 2003. [8] J. Gregory, Game Engine Architecture. CRC Press, 2009.*
7. *Aarseth, Espen. 2001. Computer Game Studies, Year One. Game Studies. The International Journal of Computer Game Research. Volume 1, Issue 1, July 2001.*
8. *Björk, Staffan. 2008. Games, Gamers, and Gaming Understanding Game Research. Mindtrek 2008, October 7–9, 2008, Tampere.*
9. *Björk, Staffan&Holopainen, Jussi. 2004. Patterns in Game Design. Charles River Media Game Development. Charles River Media.*
10. *Blessing, Lucienne T.M. &Chakrabarti, Amaresh. 2009. DRM, a Design Research Methodology. Springer.*

ISSN 2348 – 8034
Impact Factor- 5.070

11. *Bonsiepe, Gui. 2007. The Uneasy Relationship between Design and Design Research. In Michel, Ralf (Ed.) Design Research Now. Essays and Selected Papers. Birkhäuser. p. 25-41*
12. *Crookall, David. 2000. Editorial: Thirty Years of Interdisciplinarity. Simulation and Gaming. Vol. 31 No. 1, March 2000. p. 5-12. Sage Publications.*
13. *Cross, Nigel. 2007. From a Design Science to a Design Discipline: Understanding Designerly ways of Knowing and Thinking. In Michel, Ralf (Ed.) Design Research Now. Essays and Selected Papers. Birkhäuser. p. 41-55*
14. *Deterding, Sebastian. 2014. The Expectable Rise, Pyrrhic Victory, and Designerly Future of Game Studies as an Interdiscipline. In the Proceedings of Critical Evaluation of Game Studies: A seminar re-evaluating the field of game studies. Tampere: University of Tampere.*
15. *Dorst, K. &Dijkhuis, J. 1995. "Comparing paradigms for describing design activity" Design Studies Vol 16 No 228 April. p. 261-274. Elsevier Science.*
16. *Frayling, Christopher. 1993. Research in Art and Design. Royal College of Art Research Papers. Volume 1, Number 1. 1-5.*
17. *Fullerton, Tracy. 2008. Game Design Workshop, 2nd Edition: A Playcentric Approach to Creating Innovative Games. Morgan Kaufmann, February 2008.*
18. *Hagen, Ulf. 2009. "Where Do Game Design Ideas Come From? Invention and Recycling in Games Developed in Sweden". Breaking New Ground: Innovation in Games, Play, Practice and Theory. Proceedings of DiGRA 2009*
19. *Holm, Ivar. 2006. Ideas and Beliefs in Architecture and Industrial Design: How Attitudes, Orientations, and Underlying Assumptions Shape the Built Environment. Arkitektur sogdesignhøgskoleni Oslo.*
20. *Juul, Jesper. 2007. Swap Adjacent Gems to Make Sets of Three: A History of Matching Tile Games. Artifact journal. Volume 2, 2007. London: Routledge.*
21. *Juul. Jesper. 2005. Half-Real. Video Games between Real Rules and Fictional Worlds. The MIT Press.*
22. *Kuittinen, Jussi&Holopainen, Jussi. 2009. Some Notes on the Nature of Game Design. Breaking New Ground: Innovation in Games, Play, Practice and Theory. Proceedings of DiGRA 2009.*
23. *Kultima, Annakaisa. 2015. An Autopsy of the Global Game Jam 2012 Theme Committee Discussion: Deciding on Ouroboros. Proceedings of the 10th International Conference on the Foundations of Digital Games (FDG 2015), June 22- 25, 2015, Pacific Grove, CA, USA.*
24. *Kultima, Annakaisa. 2009. Casual Game Design Values. In the Proceedings of the 13th International MindTrek Conference: Everyday Life in the Ubiquitous Era. 2009. p. 58-65.*
25. *Kultima, Annakaisa. 2010. The Organic Nature of Game Ideation: Game Ideas Arise from Solitude and Mature by Bouncing. Proceedings of the International Academic Conference on the Future of Game Design and Technology.*