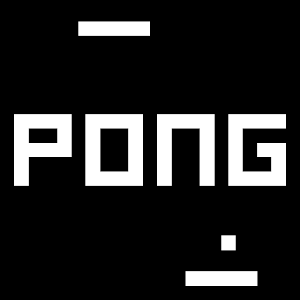


Ciaran Adams

[17005609]  [2017/18]

[Computer Science core] [Coursework]



Contents

[Introduction to my Project 2](#_Toc508940905)

[Design and final result 3](#_Toc508940906)

[The Structure (**Algorithm**) 3](#_Toc508940907)

[The Main Menu Screen 4](#_Toc508940908)

[The Controls-Screen 6](#_Toc508940909)

[Game-Screen 8](#_Toc508940910)

[Implementation 14](#_Toc508940911)

[Hardware Requirements: 14](#_Toc508940912)

[Software Requirements: 14](#_Toc508940913)

[Opening the Game (Pictures included) 15](#_Toc508940914)

[Problems I had with this project 18](#_Toc508940915)

[Conclusions 20](#_Toc508940916)

[Summary of The Project (Achieved and not achieved) 20](#_Toc508940917)

[Future of The Project 20](#_Toc508940918)

[References 21](#_Toc508940919)

## Introduction to my Project

For my project I have chosen to create a pong game made from java on processing version 3. The aim is to make good use of object orientated programming particularly collision in this case. There has been a good use of text and Images in the code for the game. A main menu screen, pause screen, win screen and a game screen have all been implemented.

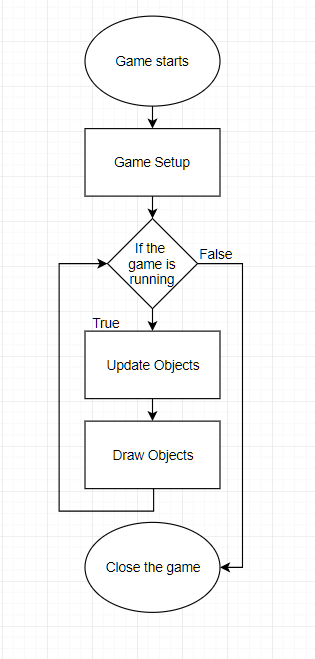
Pong is an old arcade game, one of the first ever made. It’s a table tennis like game with the use of two-dimensional graphics in this case. The game originally was released in 1972 by Atari it quickly became a success and helped establish the video game industry.[[1]](#endnote-1)

The name of the project is Pong, it’s like the base version of the pong game with the addition of some colours. It’s a 2-player based game for a single PC (or laptop) one player will make use of one paddle while player 2 makes use of the other. The first of six points wins and displays in the colour of the winner to make the game more visually pleasing to the user.

## Design and final result

### The Structure (**Algorithm**)

#### Game Loop



### The Main Menu Screen

The main menu page may not appear important however toi the user this may have an affect on them because should they think it seems boring making them not want to play the game. It will have different options “Play”, “Credits” and “Exit”. The “Play” screen will take you directly to the Game-Screen. It will make use of a background image that I have created in MS paint. I made use of “BNMachine” font with a size of “40” in white colour.

The background will be a custom image that has been created in MS paint. It will be 800 x 600 pixels to fit the window

Here is where the font will be showing the options for the user to play, view controls and Exit

Background image with title here

Font with options of what to do with instructions

Background

As the text is size “40” its quite clear to see this is a benefit for the user especially is they are visually impaired. Not only this as the colour is white and the background black it further adds to the ability for the user to see correctly.

**The colour scheme**

This was done to somewhat have a similarity to that of the original pong game however throughout the game I added my own little variations of colour.

**Positioning the text**

fill(255);

textAlign(CORNER);

text ("Press Enter to play",width/2-30,height/2- 40);

text ("Press Ctrl for Controls",width/2-30,height/2+ 20);

text ("Press Esc to exit",width/2-30,height/2+ 80);

**Key Pressed**

“Press E to Exit” will exit the application entirely.

“Press Ctrl for Controls” will bring the user to the controls page and show them the controls.

"Press Enter to play" will let the user start the game.

#### Final Result



### The Controls-Screen

This will take you to the Controls Screen which will show the user how to play the game. I made use of “BNMachine” font with a size of “40” in white colour.

Background

Here the instructions will explain how to win the game and how to control the paddles for player 1 and 2. As well as this it will also explain how to pause the game

Font with controls / instructions on how to play the game

Lets the user know how to return to the previous screen

B to go back

As the text is size “40” its quite clear to see this is a benefit for the user especially is they are visually impaired. Not only this as the colour is white and the background black it further adds to the ability for the user to see correctly.

**Key Pressed**

"Press B to return to main menu", this will bring the user back to the main menu.

**Positioning the text**

fill(255);

textAlign(CORNER);

text ("To pause press P ",width/2-200,height/2 -160);

text ("Player 1,",width/2-200,height/2 - 110);

text ("W,A,S,D",width/2-200,height/2 - 70);

text ("Player 2,",width/2 + 70,height/2 - 110);

text ("Arrow keys",width/2 + 70,height/2 - 70);

text ("Press B to return to main menu",width-width + 10 ,height- 30);

#### Final Result



### Game-Screen

This screen consists of 3 things a Pause-Screen, Win-Screen and the game itself. The background will once again be an Image I have created in MS paint. I made use of “BNMachine” font with a size of “40” in white colour.

Scores

The ball

Paddles

Ball Boundaries

Background

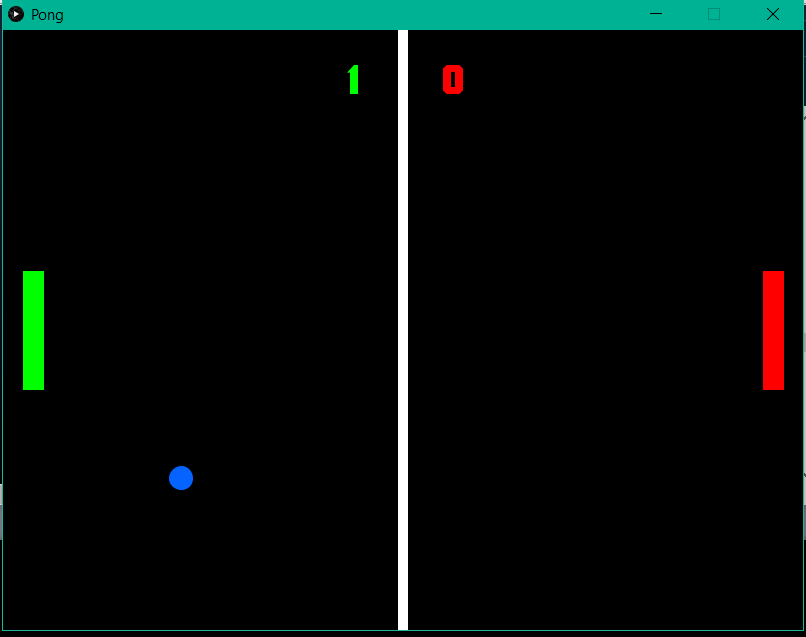
**Ball Boundaries**

**Scores**

**Paddles**

**Background**

#### Final Result



#### The Pause-Screen

This will let the user Continue the game or Exit the application entirely. The pause screen will pause the game as is and show the game in the background of the text. I made use of “BNMachine” font with a size of “40” in blue colour.

Here the font will explain how the user can continue playing and how to exit. It will just be on top of the current gamescreen as I personally enjoyed this style however to make a separate screen for the pause would have been a simple task

Declares that the game is paused

**Centring the text**

fill(5,100,255);

textAlign(CENTER,CENTER);

text("Paused, press C to continue.",width/2,height/3-20);

text("or Esc to exit", width/2, height/3 +20);

**Stopping the paddles from moving**

At first there was trouble with the pause screen as the paddles would still be able to move when the game was paused this had to be removed as it would spoil the game for the user and encourage cheating once this bug was found. The solution was to create a temp variable “**tempPaddle**” save the paddles speed in this variable then put the paddle speed to 0 removing any ability to move the paddle.

if(key == 'p' ||key == 'P'){

while (speedx !=0){

pause = true;

tempSpeedX = speedx;

tempSpeedY = speedy;

**tempPaddle = paddleS;**

**paddleS = 0;**

speedx = 0;

speedy= 0;

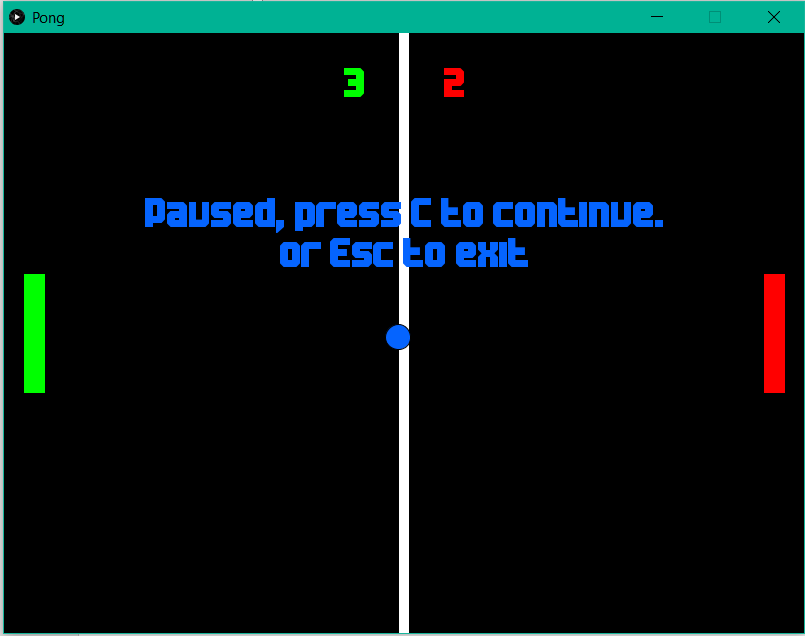
}

}

**KeyPressed**

Should the user press “Esc” it will exit the game they also have the option to play again. The user can do this by clicking on the game screen.

##### Final Result



#### The Win-Screen

This will let the user Play again or Exit the application entirely. I made use of “BNMachine” font with a size of “40” in blue colour. However, depending on which user wins the colour may change. This screen essentially used a similar sense of coding from the pause screen with a few minor adjustments as shown below.

The text will be displayed player “x” wins and will be in the colour of the player for example if player one wins the text will be displayed in green and player 2 would have the text display in red.

The winner will be displayed along with instructions to play again and exit the game.

**Centring the text**

Getting the text centred is a key role in this project the code I used was

text ("Game Over",width/2,height/3- 40);

text("Click to play again or Esc to exit", width/2, height/3 + 40);

This code got the text centred as I had hoped with no issues at all and it was used with the method “endScreen()”

**Font colour differing upon winner**

This adds to the satisfaction of the user when/if they win not only this but it makes it clearer for users who may have bad vision who won the game. The code was fairly simple as shown here

**KeyPressed**

Should the user press “Esc” it will exit the game they also have the option to play again. The user can do this by clicking on the game screen.

##### Final Result



## Implementation

### Hardware Requirements:

The Processing development environment should run on most modern computers. It will run on any computer that can run the Java Runtime Environment (JRE). The Processing website does not specify any particular minimum system requirements for hardware.

### Software Requirements:

Processing runs on Mac OS X, Windows and Linux. The actual supported platforms can be found on the Processing wiki article titled: [Supported Platforms](https://github.com/processing/processing/wiki/Supported-Platforms). If you have gotten the file in a “rar” file or any other kind of zipped file you may require to download winrar.

Earlier releases have been removed because we can only support the current versions of the software. To update old code, read the [changes](https://github.com/processing/processing/wiki/Changes) page. Changes for each release can be found in [revisions.txt](https://raw.github.com/processing/processing/master/build/shared/revisions.txt). If you have problems with the current release, please [file a bug](https://github.com/processing/processing/issues?state=open) so that we can fix it. Older releases can also be [built from the source](https://github.com/processing/processing). [Read More](https://github.com/processing/processing/wiki/FAQ#whats-with-the-version-numbers) about the releases and their numbering. To use [Android Mode](http://android.processing.org/), Processing 3 or later is required.[[2]](#endnote-2)

### Opening the Game (Pictures included)

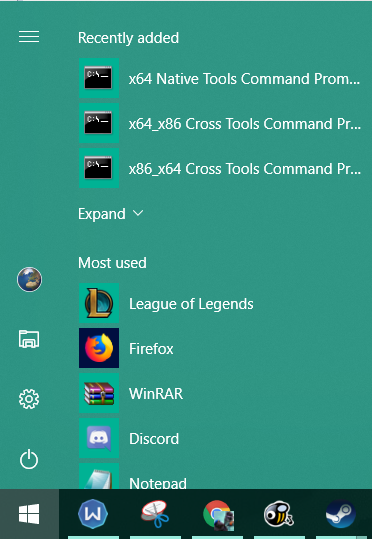
\*\*\*It is strongly recommended that you do not run the game while using an external flash drive\*\*\*

#### If you are Getting the zip file onto your pc (If the file is already on your pc skip to step 2):

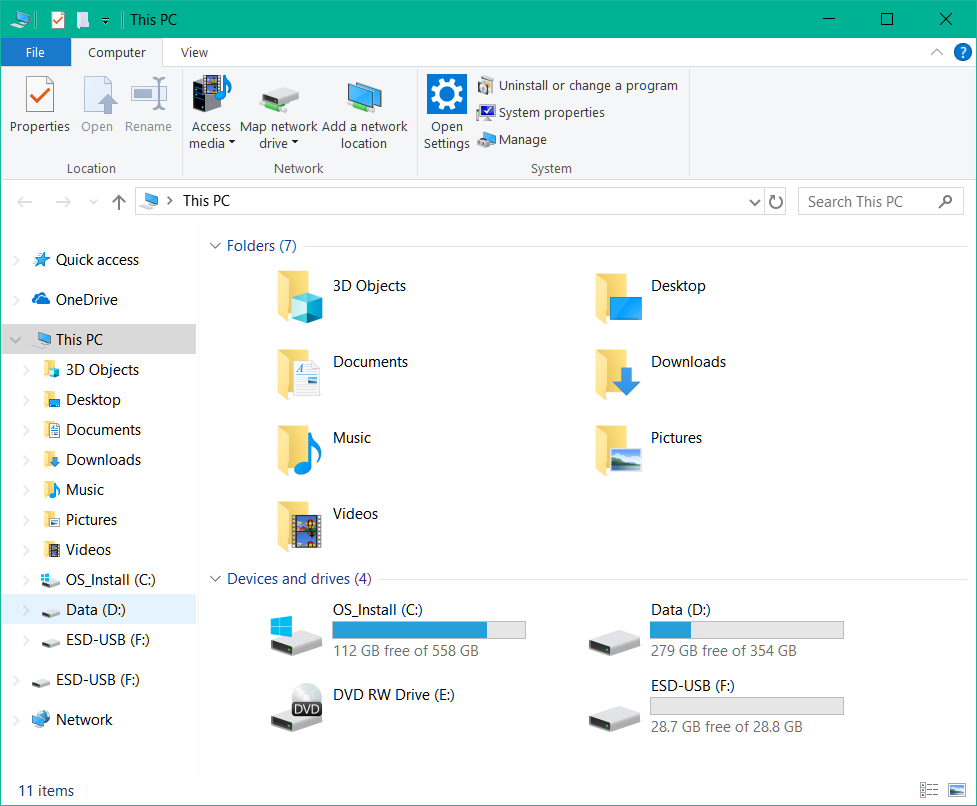
* Insert the drive into a USB port.
* Click the windows icon



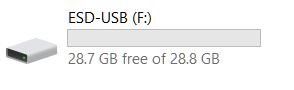
* Then click the windows explorer icon



* this should take you to “this pc” however it may be named something else on your device



* look for the removeable drive you just connected in my case its “ESO-USB(F:)” this varies depending on what your drive is named



* Double click it and here you will see the file (in this case it’s a zip file so you would progress to step 2)

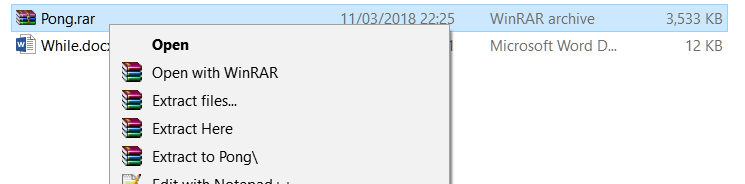


You will then want to drag the file onto your pc to the destination of your choice. #

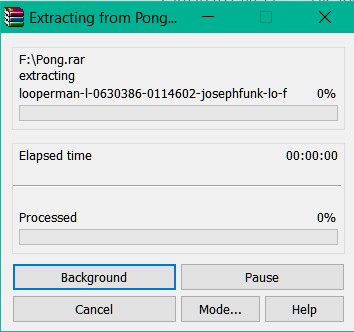
#### Unzipping the file:

if the file is not a zip file skip to step 3.

* Right click the zipped file



* Then click “Extract here” a window will appear similar to this wait…



* And when its finished you should see a folder like this



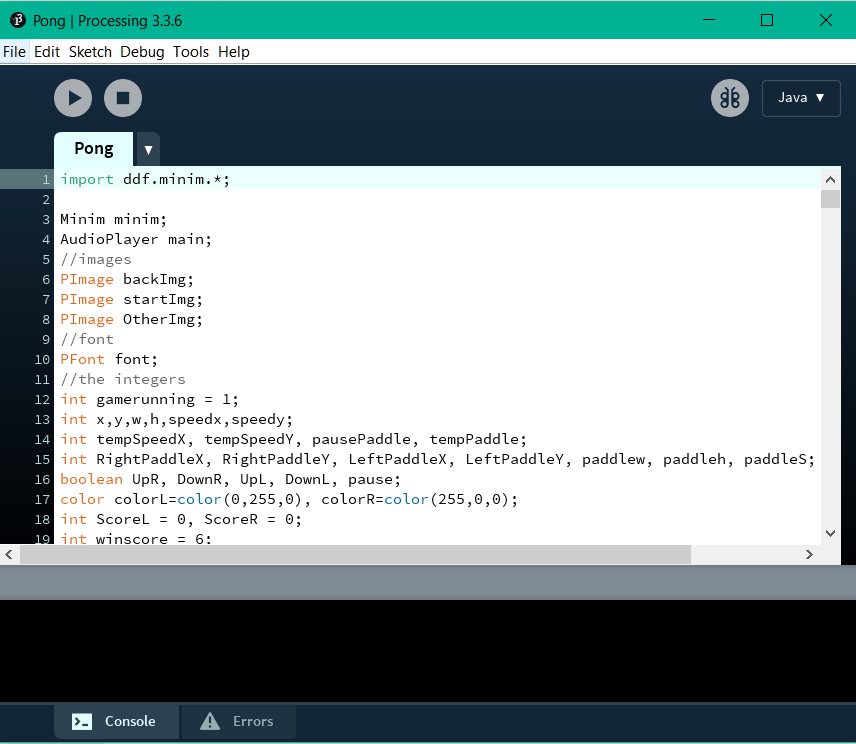
* Congratulations the game is now on your device.

#### Running the Game:

* Open  and in the folder you should find a file named “Pong.pde.



* Double click and the processing menu/ Code will Show up



* Press Play at the top left corner
* Now the main menu page will be displayed Enjoy!



## Problems I had with this project

|  |  |  |
| --- | --- | --- |
| Problem Description | The Problem or Error without the fix | The Fix for the Problem |
| The collision would not work how it was supposed to be. They ball would not pass the X – axis of the paddle resulting in an endless game of it bouncing back and forth with no score of a never-ending game. | void collisionPaddle(){  if(x - w/2 < LeftPaddleX+paddlew/2 ){  if(speedx<0){  speedx = speedx-1;  speedx=-speedx;  }  }  else if(x + w/2 > RightPaddleX-paddlew/2 ){  if(speedx>0){  speedx=speedx+1;  speedx=-speedx;  }  }    } | void collisionPaddle(){  if(x - w/2 < LeftPaddleX+paddlew/2  && y - h/2 < LeftPaddleY+paddleh/2  && y + h/2 > LeftPaddleY-paddleh/2 ){  if(speedx<0){  speedx = speedx-1;  speedx=-speedx;  }  }    else if(x + w/2 > RightPaddleX-paddlew/2  && y - h/2 < RightPaddleY+paddleh/2  && y + h/2 > RightPaddleY-paddleh/2 ){  if(speedx>0){  speedx=speedx+1;  speedx=-speedx;  }  }    } |
| The background image would not load. I forgot to initialize the image | The game would not launch | PImage backImg; |
| Paddles would go off Screen however the paddles where moving correctly I decided to add a method that would stop them from going off screen | void movePaddle(){  //"if(UpL)" is equivelent to "if(UpL= true)"  if(UpL){  LeftPaddleY = LeftPaddleY -paddleS;  }  if(DownL){  LeftPaddleY = LeftPaddleY +paddleS;  }  if(UpR){  RightPaddleY = RightPaddleY -paddleS;  }  if(DownR){  RightPaddleY = RightPaddleY +paddleS;  }  } | void boundryPaddle(){  if(LeftPaddleY - paddleh/2 <0) {  LeftPaddleY = LeftPaddleY + paddleS;  }  if(LeftPaddleY + paddleh/2 >height){  LeftPaddleY = LeftPaddleY - paddleS;  }  if(RightPaddleY - paddleh/2 <0) {  RightPaddleY = RightPaddleY + paddleS;  }  if(RightPaddleY + paddleh/2 >height){  RightPaddleY = RightPaddleY - paddleS;  }  } |
| Could still move paddles when the game was paused. This is because I did not add any changed to the paddle when the game was paused however if it where set to 0 the paddles would stay paused that’s why I added a tempPaddle. | if(key == 'p' ||key == 'P'){  while (speedx !=0){  pause = true;  tempSpeedX = speedx;  tempSpeedY = speedy;  speedx = 0;  speedy= 0;  }  } | if(key == 'p' ||key == 'P'){  while (speedx !=0){  pause = true;  tempSpeedX = speedx;  tempSpeedY = speedy;  **tempPaddle = paddleS;**  **paddleS = 0;**  speedx = 0;  speedy= 0;  }  } |
| The ball would slow down upon collision on the Player 1 paddle. | The collision was to set to:  speedy = speedy + 1  speedx = speedx + 1 | speedy= -speedy;  speedx= -speedx; |

## Conclusions

### Summary of The Project (Achieved and not achieved)

Overall This project has achieved everything that was planned, a main menus screen was added, a pause screen, and a win screen. The collision worked perfectly the text showed correctly and even the scores where working everything that I had hoped to add.

However, with additional time many things could have been implemented to improve the programme and make it feel much more professional things like adding collision sound, background music maybe even replace the main menu with some GUI buttons.

### Future of The Project

This was my first major project in java I intend to make good use of it, add to it and maybe even optimise the code more than it already has been. It would be nice to add additional game modes such as “1 v bot” where a user would play against artificial intelligence, a wall mode where the user could practice their skills in the game to add a more competitive field to the game.

The general idea is to merge this programme to a html-based website, on this website it will have a few different games all classic arcade style games such as space invaders, Tetris, Pac-man, Helicopter, Asteroids, Snake and maybe even a simple fighting game like Street Fighter. The Idea came from the popularity of Consoles that are being re-made with a modern touch such as the SNES mini.

This will require a lot of time, effort and dedication however the hope is that it would be completed within a year.

## References

1. Ellis, David (2004). "A Brief History of Video Games". Official Price Guide to Classic Video Games. [↑](#endnote-ref-1)
2. https://processing.org/download/ [↑](#endnote-ref-2)