Binary freestyle pong

TOC \o 2-2 \t "Heading, 2,Subtitle, 3,Title, 4"

Defining and understanding PAGEREF \_Toc \h 2

Introduction PAGEREF \_Toc1 \h 2

Requirements PAGEREF \_Toc2 \h 2

Context diagram PAGEREF \_Toc3 \h 2

Planing and designing PAGEREF \_Toc4 \h 3

Gantt Chart PAGEREF \_Toc5 \h 3

Interface design PAGEREF \_Toc6 \h 3

Algorithm PAGEREF \_Toc7 \h 4

This is the code for detecting if player two has hit a boundary PAGEREF \_Toc8 \h 4

Test data PAGEREF \_Toc9 \h 4

Data dictionary PAGEREF \_Toc10 \h 4

Implementing PAGEREF \_Toc11 \h 5

Source code PAGEREF \_Toc12 \h 5

Testing and evaluating PAGEREF \_Toc13 \h 5

Peer tests PAGEREF \_Toc14 \h 5

Evaluation PAGEREF \_Toc15 \h 5

Maintaining PAGEREF \_Toc16 \h 5

Enhancements review PAGEREF \_Toc17 \h 5

content versioning PAGEREF \_Toc18 \h 5

Defining and understanding

## 

## Introduction

The project that will be created is a 2 player pong like game made in Visual C#. The main twist in this game is that points are awarded for hitting the sides increasing in powers of 2 without hitting the ball yourself again until your opponent has hit it and started gaining points for them. There will also be a title screen with high scores and a instruction screen

## 

## Requirements

* + - 1 or more moving balls that
      * correctly rebound off surfaces
      * move in straight line
    - paddles that
      * controllable
      * rotate
      * cant escape field
      * only current next hitter can move through other player
    - score system that
      * correctly count
      * record scores
      * finish game after score limit is reached
      * best of 3
    - Sides that
      * reflect ball
      * numbered
      * add score
      * act as boundary
    - Extras
      * instruction screen
      * menu screen
      * Gems that
        + add score multiplayer
        + modify bat
        + modify ball

## Context diagram

Planing and designing

## Gantt Chart

## 

## Interface design

## 

## 

## Algorithm

if (GoingUpPlayer2)

if (playerTwo.Bounds.IntersectsWith(topWall.Bounds)== false)

playerTwo.Location = new Point(playerTwo.Location.X, playerTwo.Location.Y - 2);

if (GoingRightPlayer2)

if (playerTwo.Bounds.IntersectsWith(rightWall.Bounds) == false)

playerTwo.Location = new Point(playerTwo.Location.X + 2, playerTwo.Location.Y);

if (GoingDownPlayer2)

if (playerTwo.Bounds.IntersectsWith(bottomWall.Bounds) == false)

playerTwo.Location = new Point(playerTwo.Location.X, playerTwo.Location.Y + 2);

if (GoingLeftPlayer2)

if (playerTwo.Bounds.IntersectsWith(leftWall.Bounds) == false)

playerTwo.Location = new Point(playerTwo.Location.X - 2, playerTwo.Location.Y);

# This is the code for detecting if player two has hit a boundary

## Test data

|  |  |
| --- | --- |
| Press D | Player one moves right |
| Press Down key | Player Two moves down |
| Press enter | Player two rotates |
| Hits wall | Ball rebounds |
| Hits paddle | ball rebounds |

## Data dictionary

|  | Type | Description |
| --- | --- | --- |
| Score 1 | Integer | Current score for player 1 |
| Score 2 | Integer | Current score for player 2 |
| PlayerIsScoring | String | Who is currently scoring |
| PowerName1 | String | The Power on player 1 |
| PowerName2 | String | The power on player 2 |
| Player1Rotation | Integer | The rotation of player 1 |
| Player2Rotation | Integer | The rotation of player 2 |
| Player1Location | Integer | The location of player 1 |
| layer2Location | Integer | The location of player 2 |
| ballLocation | Integer | The location of the ball |

Implementing

## Source code

Source code can be Found at [www.github.com/ollyroo/binaryfreestylepong/](http://www.github.com/ollyroo/binaryfreestylepong/)

Testing and evaluating

## Peer tests

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Peer tester | Press D | Down key | Enter | Hits wall | Hits paddle |
| Bailey | Yes | Yes | Yes | Yes | Yes |
| Father | Yes | Yes | Yes | Yes | Yes |

## 

## Evaluation

This prodject was only a semi-success. The game is functional to a high degree but some of the clients requests like a end game and best of three functionality. But the base system is there and could be worked on for a couple more weeks to obtain a more finalised product. Next time maybe more accurate time allocation could be made to make more efficient work time.

[[1]](#footnote-1)

Maintaining

## Enhancements review

An enhancement that can be made is to add powers that effect the ball or bats in different ways. A nearby friend suggested that the ball currently moves to rigidly and could benefit from having more angles to bounce at.

## content versioning

Content versioning was semi-made through www.github.com/ollyroo/binaryfreestylepong/

1. By Olly Clissold [↑](#footnote-ref-1)