ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN
KHOA CÔNG NGHỆ PHẦN MỀM

---------- o0o ----------

BÁO CÁO CUỐI KÌ

NGÔN NGỮ LẬP TRÌNH JAVA

Đề tài: GAME 2048

Giảng viên hướng dẫn : ThS. Huỳnh Tuấn Anh

Nhóm thực hiện:

Đào Đức Huy 15520295

Lê Thanh Quang 15520065

Tp. Hồ Chí Minh, tháng 6/2019

NHẬN XÉT CỦA GIẢNG VIÊN

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

LỜI CẢM ƠN

Trong học kỳ này, chúng em được tiếp cận với môn học "Ngôn ngữ lập trình Java" – môn học rất hữu ích đối với sinh viên Khoa Công nghệ phần mềm. Do đó, để nhóm chúng em hoàn thành được tốt đồ án môn học này, không thể không nói đến công lao của thầy Huỳnh Tuấn Anh. Chúng em xin chân thành cảm ơn thầy đã tận tâm hướng dẫn chúng em qua từng buổi học trên lớp. Bên cạnh đó, nhóm cũng xin gửi lời cảm ơn chân thành đến các anh chị khóa trên, các bạn trong và ngoài lớp đã sẵn lòng chia sẻ tài liệu cũng như kinh nghiệm từng trải của bản thân để nhóm chúng em học tập và tránh mắc những sai lầm, tiết kiệm được thời gian trong quá trình thực hiện đồ án.

Tuy nhiên, do kiến thức và khả năng của chúng em còn nhiều hạn chế, do đó không tránh khỏi những thiếu sót, yếu kém. Chúng em rất mong nhận được những ý kiến đóng góp quý báu của thầy cô và các bạn học cùng lớp để đồ án được hoàn thiện hơn và rút ra được kinh nghiệm cho quá trình làm game sau này.

Sau cùng, chúng em xin kính chúc quý thầy cô ở Khoa Công nghệ Phần mềm, đặc biệt là thầy Huỳnh Tuấn Anh thật dồi dào sức khỏe để tiếp tục thực hiện sứ mệnh cao đẹp của mình là truyền đạt kiến thức cho thế hệ mai sau.

Nhóm xin chân thành cảm ơn!

MỤC LỤC

CHƯƠNG 1. GIỚI THIỆU

1.1. Giới thiệu game 2048

1.1.1. Lịch sử ra đời

2048 là game giải đố được tạo ra bởi nhà thiết kế web người Italy tên là Gabriele Cirulli. Nhiệm vụ chính của trò chơi là trượt ô để kết hợp chúng lại tạo thành ô 2048. Tuy vậy người chơi vẫn có thể tiếp tục chơi sau khi đạt được ô 2048 để tạo ra số lớn hơn.

2048 ban đầu được viết bằng ngôn ngữ JavaScript và CSS trong ngày cuối tuần, và game được ra mắt vào ngày 9 tháng 3 dưới dạng phần mềm mã nguồn mỡ dưới giấy phép MIT.

Circulli 19 tuổi đã tạo ra game trong lúc đặt ra thử thách rằng anh ta có thể tạo game từ ban đầu hay không. Anh ấy ngạc nhiên khi trò chơi của anh ấy đạt 4 triệu lượt tham gia trong vòng chưa đầy hơn 1 tuần, đặc biệt hơn nó chỉ là dự án làm game trong vòng cuối tuần. “Nó chỉ là cách để bán thời gian “ – anh ấy nói. Trò chơi miễn phí. Cirulli nói rằng anh ấy không muốn kiềm tiền từ những thứ mà anh ấy không phát minh. Anh đã phát hành trò chơi lên thiết bị di động iOS và Android vào tháng 5 năm 2014.

2048 là trò chơi gây nghiện và tạo ra cú hit lớn trong cộng đồng thời bấy giờ. Trò chơi được tò báo Wall Street mô tả như “game Candy Crush cho con nghiện toán”, và tờ Business Insider gọi trò chơi là "Threes on steroids".

1.1.2. Gameplay

The objective of 2048 is to achieve “2048” on one of your tiles. To achieve “2048”, you must make matches on the board. To make a match, you must slide two tiles that have the same number on them together. So if you have two “2” tiles next to one another and you slide them so that they would be next to each other at the end of that move, the two tiles will combine. The two tiles will combine to make one tile with the number “4” in it. This new “4” tile will only be able to match with another tile with the number “4” in it.

At the start of the game, there are two tiles on your board. The two tiles will have a low number on them, most likely a “2” or a “4”. The board has the potential to hold 16 tiles. To play, you have to slide the screen either vertically or horizontally. If you slide horizontally, all of the tiles will move as far as they can to the horizontal wall. If you slide vertically, all of the tiles will move as far as they can to the vertical wall. A tile can only go as far as there are open spaces for it to slide into.

Every time you slide the tiles to one side of the board, or the wall, it counts as a move. With each move you make in 2048, a new tile is added to the board. The new tile will be a low number. So, hypothetically, if you make a lot of moves without making any matches, the board will continue to fill up until you have no empty spaces for you tiles to move into. Once this happens, and you cannot make any matches, you lose the game. That is basically all you need to know for how to play 2048.

2048 is a puzzle game that is fueled by simple addition. These are the addition problems you should know that are the core of how to play the game:

2 + 2 = 4

4 + 4 = 8

8 + 8 = 16

16 + 16 = 32

32 + 32 = 64

64 + 64 = 128

128 + 128 = 256

256 + 256 = 512

512 + 512 = 1,024

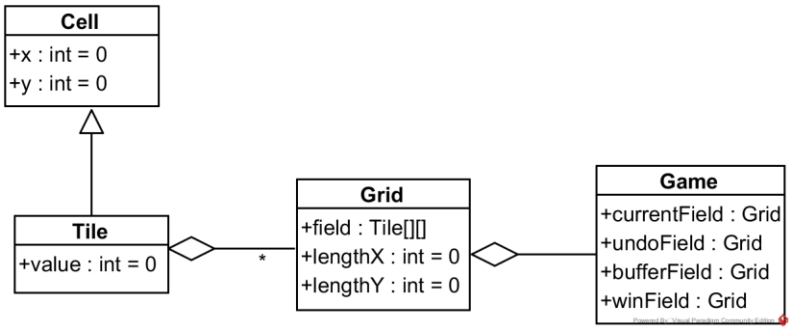
1,024 + 1,024 = 2,048

And it’s not a game changer if you don’t know math. When it comes down to it, 2048 is a matching game. You just want to slide the tiles will the same number into each other. Be careful: when you get higher numbered tiles, they’ll get in your way of matching other tiles and your board can easily become congested. Good Luck and have fun!

CHƯƠNG 2. THIẾT KẾ

2.1. Thiết kế cấu trúc dữ liệu trò chơi

Sơ đồ lớp:



Cell: là một ô có vị trí cụ thể trong tọa độ 2 chiều.

Tile: kế thừa **Cell**, là một ô có giá trị cụ thể.

Grid: là mảng hai chiều các **Tile**, là thể hiện của bảng số.

Game có 4 **Grid** chính:

currentField: bảng số hiện tại của trò chơi.

bufferField: bảng số tạm của trò chơi.

undoField: bảng số trạng thái trước đó của trò chơi.

winField: bảng số trạng thái đích.

2.2. Thiết kế giải thuật trò chơi

2.2.1. Giải thuật tạo màn chơi

Giải thuật tạo bản số ban đầu:

Input: Kích cỡ bản số lengthX, lengthY

Output: Bản số ban đầu (currentField)

Lưu đồ giải thuật:

2.2.2. Giải thuật di chuyển

Giải thuật tìm vector di chuyển:

Input: Hướng di chuyển (direction(left, right, up, down))

Output: Vector di chuyển (vector)

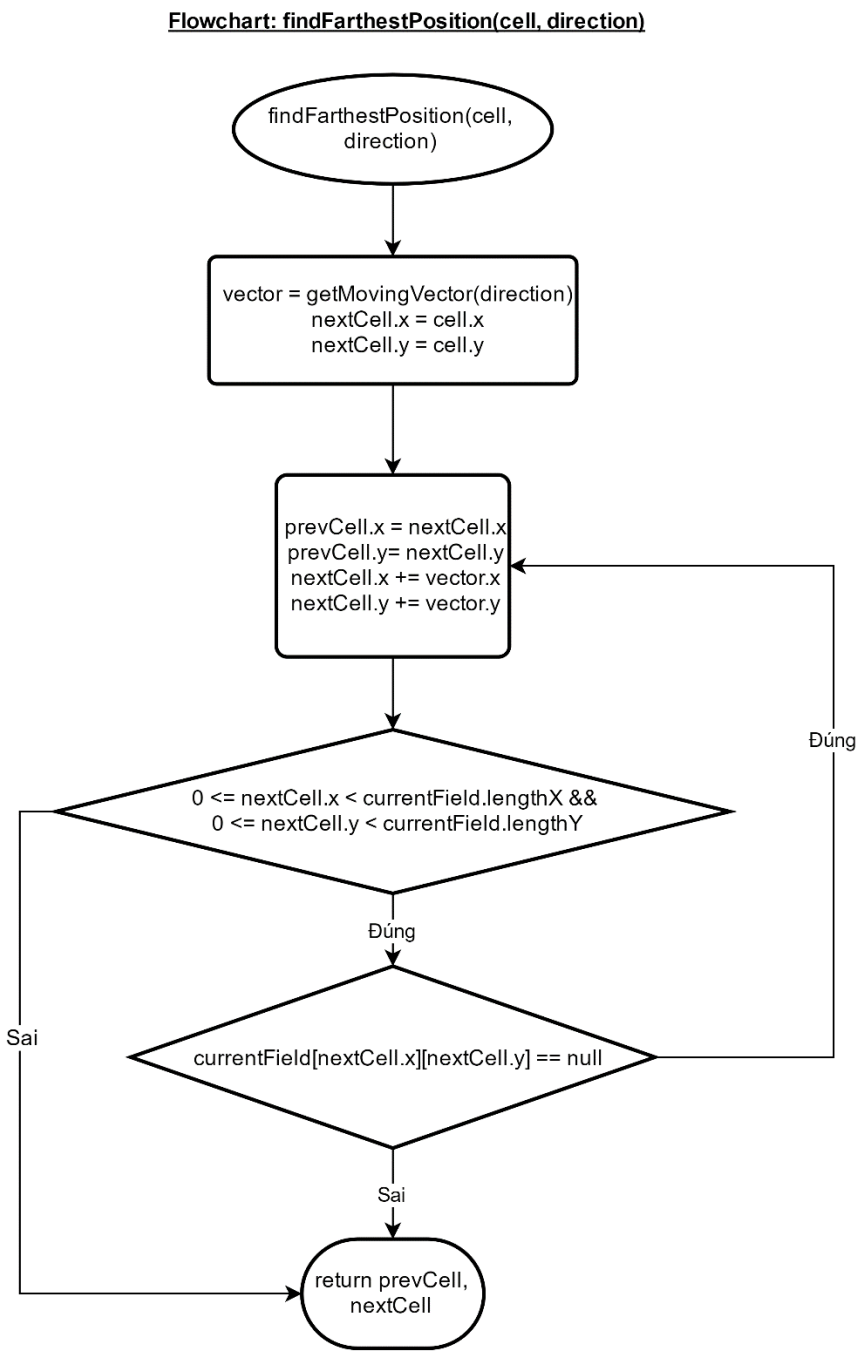
Lưu đồ giải thuật:

Giải thuật tìm vị trí ô đầu tiên chạm phải:

Input: Vị trị của ô (cell), hướng di chuyển (direction(left, right, up, down))

Output: Vị trí của ô đầu tiên chạm phải(nextCell) và vị trí ô trước đó(prevCell)

Lưu đồ giải thuật:

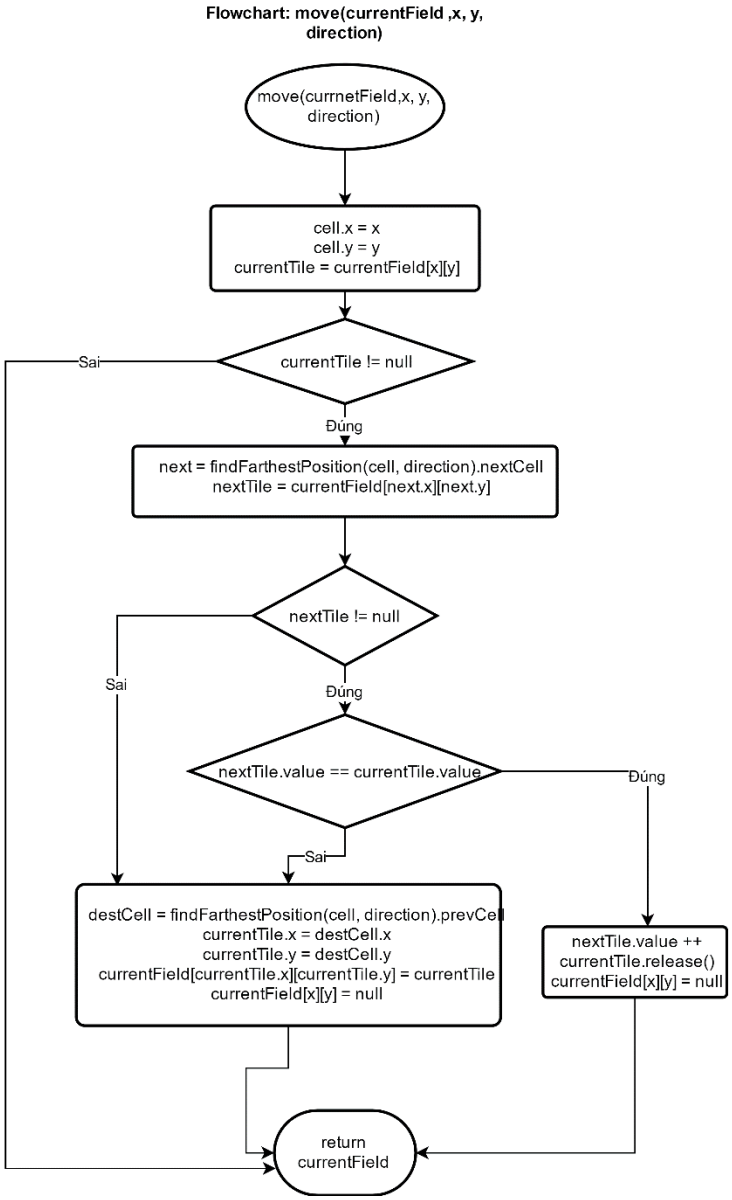


Giải thuật di chuyển:

Input: Vị trị của ô (cell), hướng di chuyển (direction(left, right, up, down))bảng số (currentField)

Output: Bảng số đã di chuyển (currentField)

Lưu đồ giải thuật:

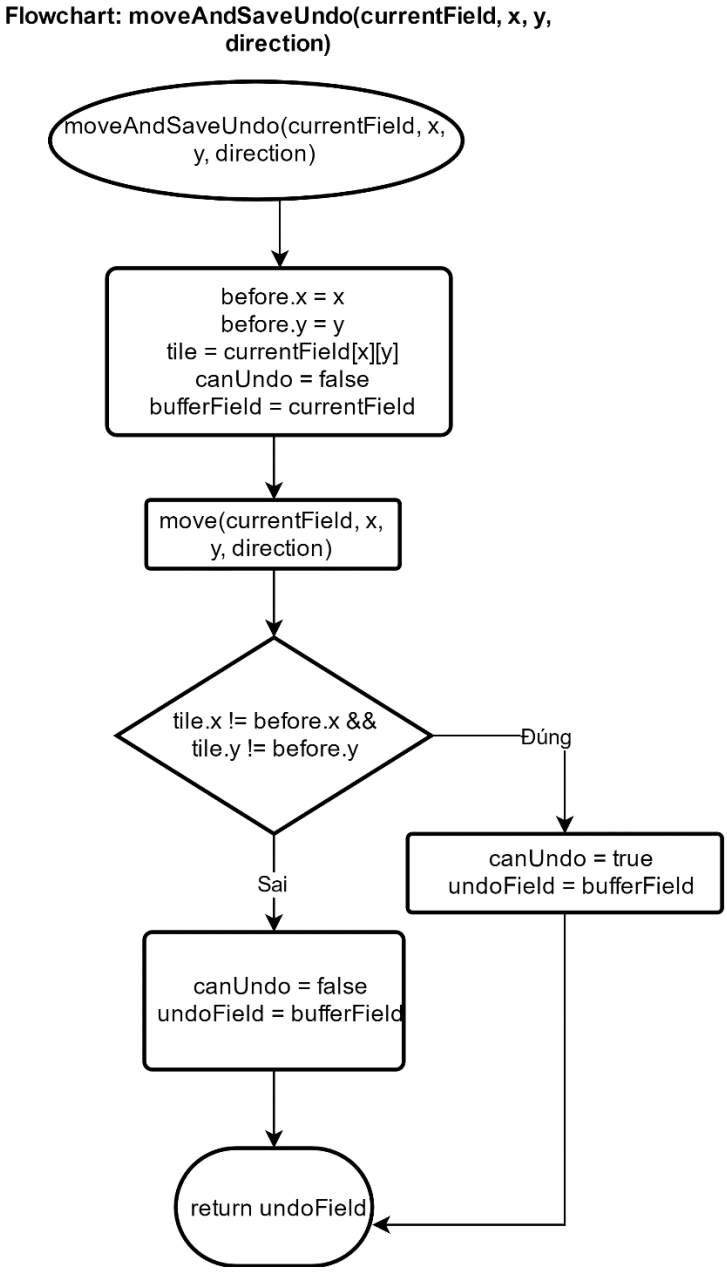


2.2.3. Giải thuật lưu bảng hoàn tác

Input: Bảng số hiện tại (currentField), vị trí ô di chuyển (x,y), hướng di chuyển (direction(left, right, up, down)).

Output: Bảng số hoàn tác (undoField).

Lưu đồ giải thuật:

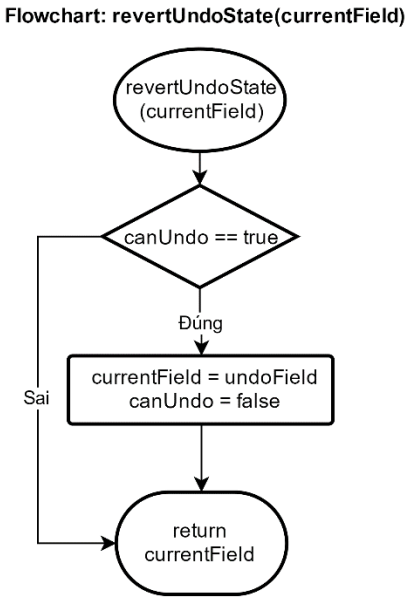


2.2.4. Giải thuật hoàn tác

Input: Bảng số hiện tại (currentField).

Output: Bảng số đã hoàn tác (currentField).

Sơ đồ giải thuật:



2.2.5. Các thông số field hiện tại

2.2.6. Các thuật toán

2.2.6.1. Minimax

2.2.6.2. Monocinoty

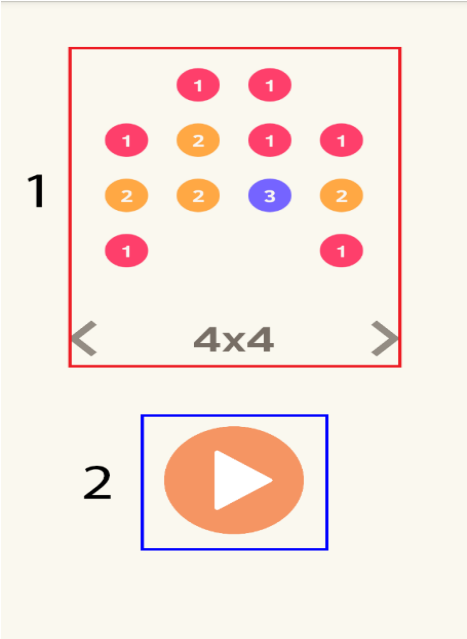
2.2.6.3. Smoothness

2.2.6.4. EmplyCell

2.3. Thiết kế giao diện trò chơi

2.3.1. Màn hình menu

Giao diện



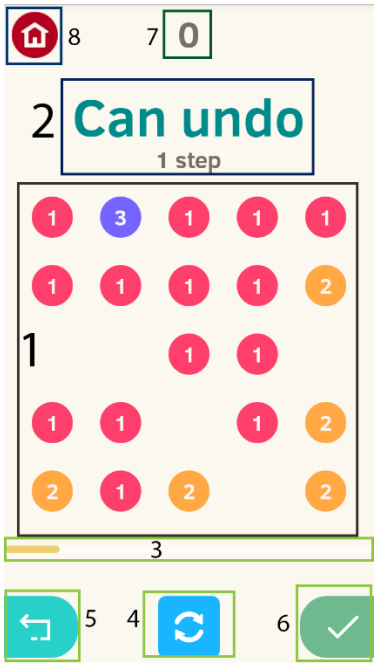
Hình 1: Giao diện menu

Chức năng

STT	Chức năng	Mô tả
1	Chọn mức độ	Người dùng có thể vuốt qua trái, qua phải hoặc nhấn nút mũi để chọn mức độ
2	Vào game	Người chơi nhấn nút để vào game với độ khó đã chọn

2.3.2. Màn hình trò chơi

Giao diện



Hình 2: Giao diện trò chơi

Chức năng

STT	Chức năng	Mô tả
1	Bảng số	Vùng chơi chính, có thể vuốt qua trái, qua phải, lên trên, xuống dưới để di chuyển các ô.
2	Thông báo	Cung cấp trạng thái của trò chơi cũng như gợi ý trong suốt quá trình chơi.
3	Giới hạn thời gian	Cho biết thời gian còn lại để hoàn thành màn chơi.
4	Khởi động lại	Kết thúc màn chơi hiện tại và qua màn chơi mới.
5	Hoàn tác	Trả về bảng số về trạng thái trước đó.
6	Kiểm tra	Cho biết những ô nào đúng và sai.
7	Điểm số	Cho biết điểm số hiện tại.

8	Home	Trở về màn hình Menu.

2.4. Thiết kế giải thuật AI

CHƯƠNG 3. CÀI ĐẶT VÀ THỬ NGHIỆM

3.1.1. Nền tảng công nghệ:

- Để phục vụ cho việc thực hiện đồ án, nhóm đã sử dụng các công cụ hỗ trợ như:

+ IntelliJ IDEA

+ Android Studio

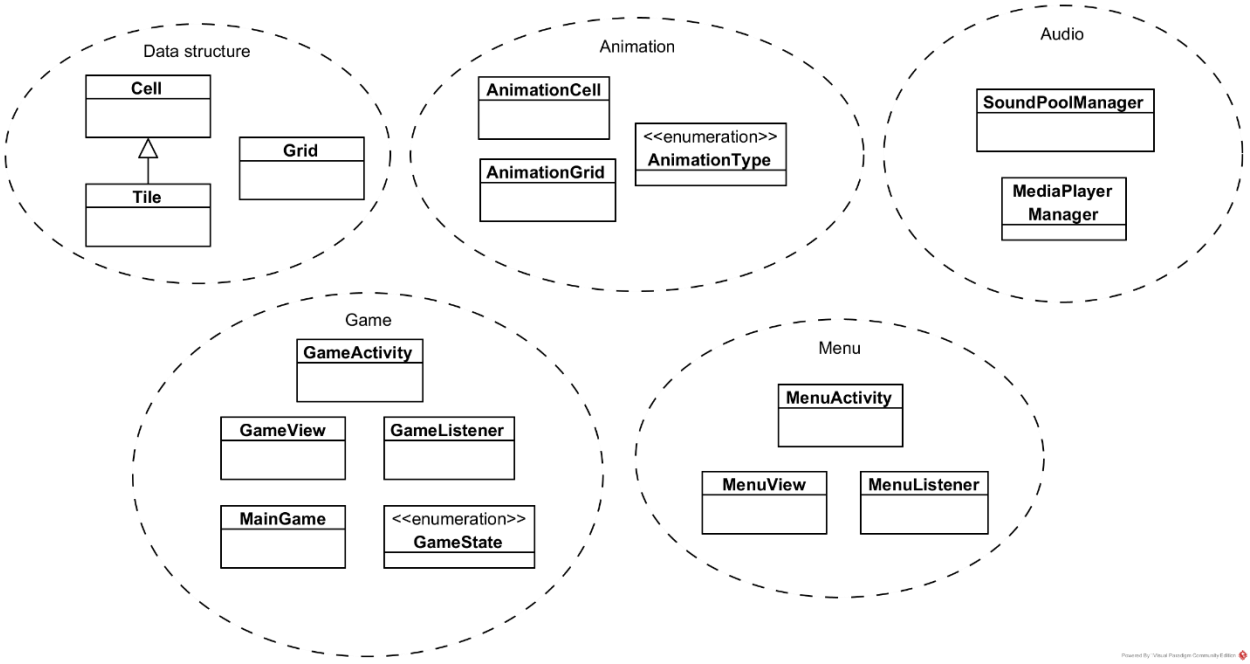
+ Genymotion/Nox

+ Microsoft Office 2016 / Microsoft Word Online

+ GitHub

- Ngôn ngữ sử dụng lập trình: Java. Framework: LibGDX.

3.1.2. Sơ đồ lớp



3.1.3. Cài đặt chương trình

3.1.4. Cell

Class Cell dùng để biểu diễn một ô trong game, có 2 giá trị là hoành độ x và tung độ y, để xác định vị trí:

-private int x;

-private int y;

Phương thức chính của class là Constructor, các hàm getter, setter.

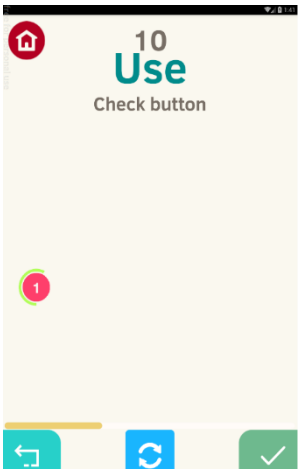
3.1.5. Tile

Một Cell trong game có thể có chứa giá trị hoặc không. Một Cell có chứa giá trị được gọi là một Tile. Thành phần gồm một cell và giá trị.

3.1.6. Animation Type

Class Animation Type chứa một Enum các giá trị đại diện cho một kiểu Animation:

- SPAWN: Animation khi ô mới được thêm vào
- MOVE: Là Animation khi thực hiện việc di chuyển các ô trong game
- MERGE: Là Animation khi thực hiện gộp 2 ô có giá trị giống nhau và va chạm nhau.
- CHECK_RIGHT_CIRCLE/ CHECK_WRONG_CIRCLE: Là Animation biểu diễn vòng tròn kiểm tra một ô có đúng hay chưa. Một ô được gọi là đúng nếu như vị trí của ô đó có vị trí và giá trị bằng với ô tương ứng trong yêu cầu. Các ô đúng sẽ được bao quanh bởi một vòng tròn (circle) màu xanh, các ô sai sẽ tạm ẩn đi một thời gian ngắn.



Hình 3:Animation check

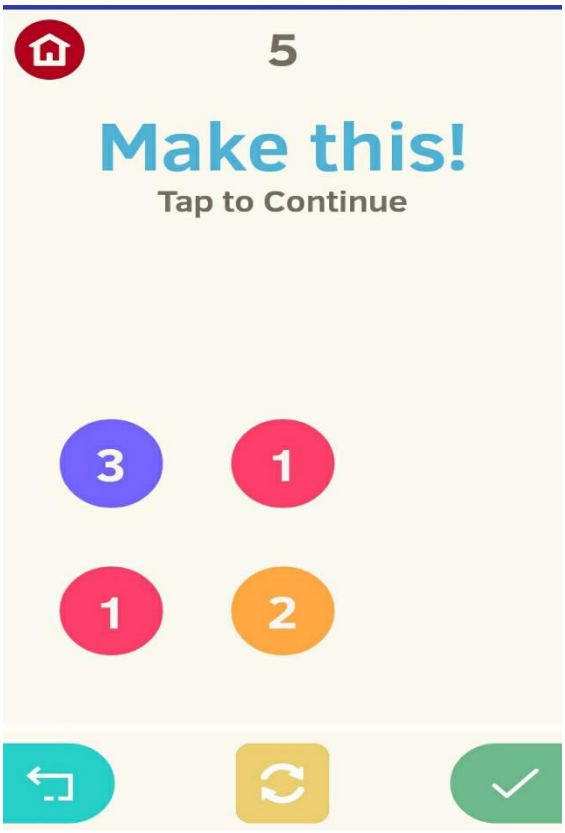
- SWIPE_LEFT/SWIPE_RIGHT: Là Animation biểu diễn việc thay đổi độ khó bằng cách lướt màn hình về bên trái/phải. Độ khó trong game được đánh giá bằng kích thước của grid. Kích thước càng lớn thì game được xem là càng khó.
- NONE: Không có Animation.

3.1.7. GameState

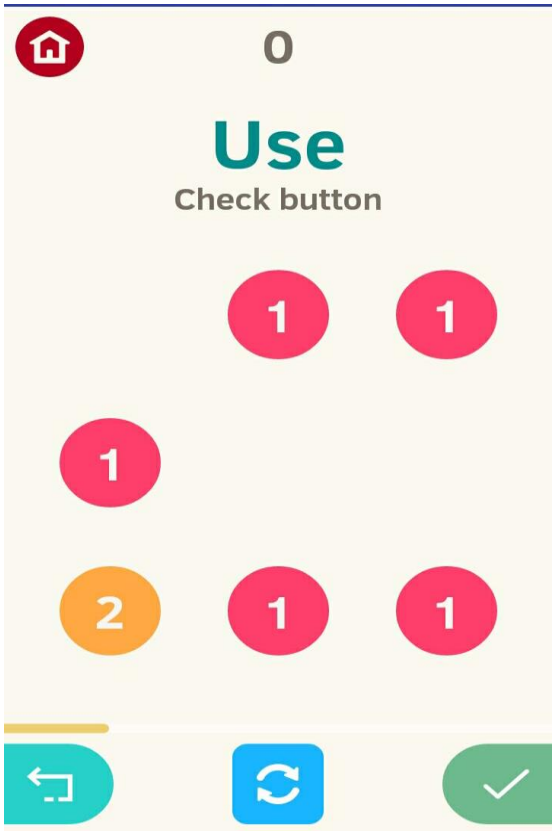
Cũng là một enum lưu trữ các trạng thái có được trong game

- NORMAL: Trạng thái bình thường của game, là trạng thái lúc người chơi đang chơi. Chương trình chỉ việc xử lý các thao tác điều khiển các ô của người chơi trong trạng thái này.
- WIN: Là trạng thái mà tất cả các ô mà người chơi tạo thành trùng khớp với yêu cầu từ game.
- LOST: Ngược lại với WIN, khi hết thời gian mà người chơi chưa WIN, game sẽ chuyển qua trạng thái LOST.

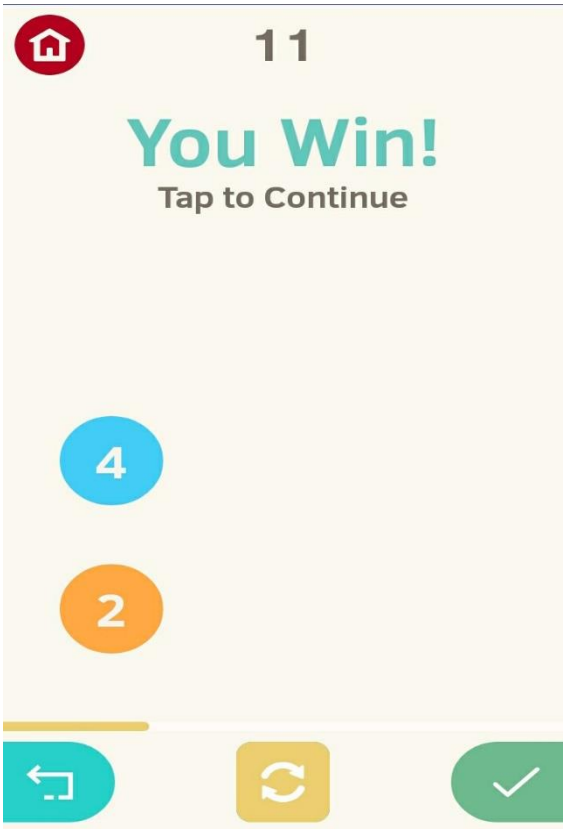
- READY: Là trạng thái mà cho người chơi quan sát yêu cầu game, và ghi nhớ, trước khi bắt đầu chơi.



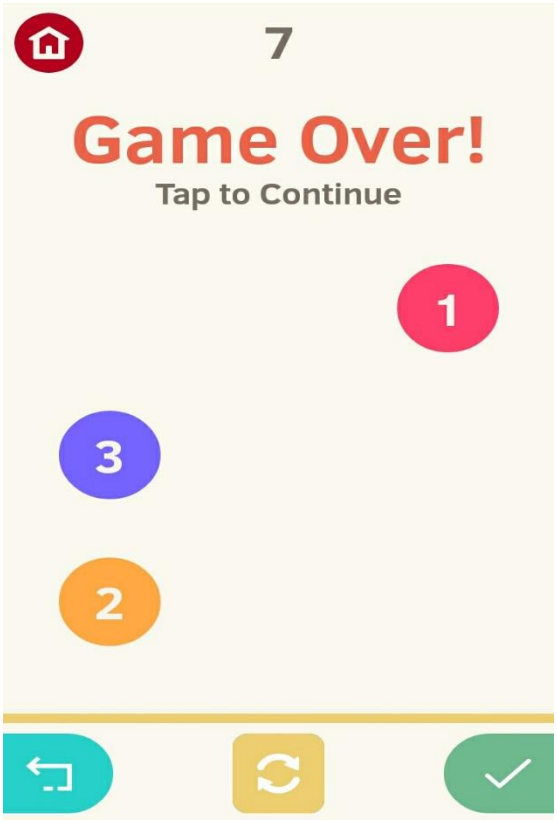
Hình 9: Ready



Hình 10:Normal



Hình 11: Win



Hình 12:Lost

3.1.8. Animation Cell

Animation Cell là một class chứa thông tin về Animation của một ô.

- int[] extras: lưu thông tin phụ đi kèm. Ví dụ một ô cần biết được điểm đến, biến extras sẽ lưu điểm đến.
- long animationTime: thời gian chạy của Animation.
- long delayTime: thời gian trì hoãn Animation.
- long timeElapsed: Thời gian đã trôi qua của Animation. Thông số này cần để tính toán được Animation đã hoàn thành bao nhiêu phần trăm so với animationTime, với mỗi tỷ lệ, ta sẽ có một vòng tròn đại diện cho số liệu đó

3.1.9. Grid

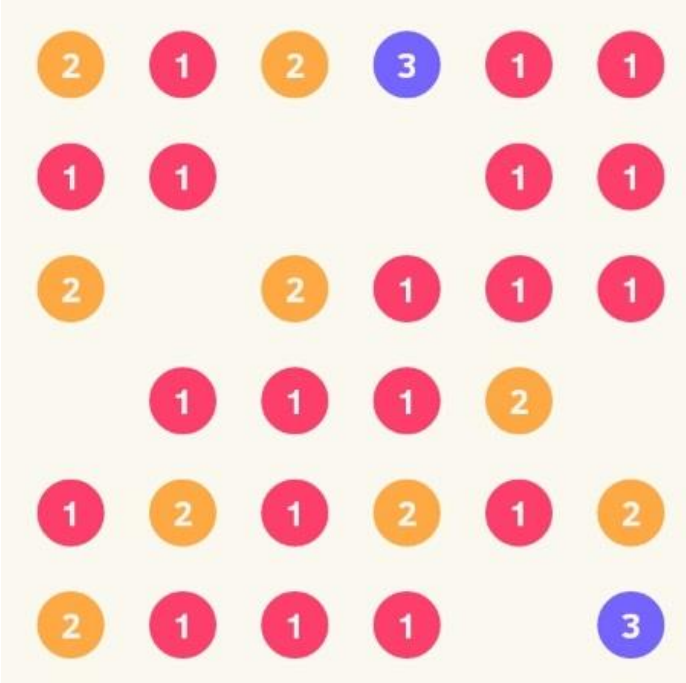
Một lưới (grid) tức là một ma trận các ô (cell). Nếu cell đại diện cho một ô với các thông số cơ bản thì grid cũng đại diện cho một lưới với một biến chính:

- Tile[][] field: Một tập hợp các ô sắp thành ma trận.

Ngoài ra còn undoField và bufferedField cũng là một tập hợp các ô, phục vụ cho việc người chơi muốn đi lùi (Undo).

3.1.10.AnimationGrid

Giống với Grid là một tập hợp các Cell thành một ma trận, AnimationGrid là một tập hợp các AnimationCell thành ma trận.



3.1.11.MenuView:

MenuListener lắng nghe các sự kiện và gọi các phương thức từ MenuActivity. Có các hàm xử lý:

- onTouch: Trả về kết quả người dùng có chạm vào màn hình hay không, các trường hợp xử lý là người dùng ấn (ACTION_DOWN), thả (ACTION_UP) và di chuyển (ACTION_MOVE, quá trình giữa ACTION_DOWN và ACTION_UP).
- rectanglePressed: trả về kết quả một hình chữ nhật (button play, button back, button home) có được chọn hay không.
- iconPressed: trả về kết quả một icon có được chọn hay không

3.1.12.GameActivity

GameActivity có nhiệm vụ tiếp nhận độ khó từ MenuActivity và khởi tạo Game tương ứng. Các hàm chính:

- onCreate: nhận độ khó và khởi tạo game.

3.1.13.MainGame

MainGame là phần xử lý logic của Game. MainGame chứa các thông tin chi tiết của Game đang chạy. MainGame bao gồm những dữ liệu sau:

- numCellX: số ô theo chiều ngang.
- numCellY: số ô theo chiều dọc.
- gameState: trạng thái hiện tại của trò chơi.
- timer: thời gian đã qua của game hiện tại.
- maxTime: thời gian giới hạn của màn chơi.
- grid: bảng số hiện tại của trò chơi.
- winGrid: bảng số chiến thắng của trò chơi.

Các hàm chính:

- newGame(): tạo lập màn chơi mới. Trong hàm này bảng số ngẫu nhiên sẽ được tạo và bảng số đích có được bằng cách di chuyển ngẫu nhiên tất cả các ô cùng một lúc bảng số ngẫu nhiên vừa tạo ra. Sau đó bảng số đích sẽ được hiển thị.
- gameStart(): bắt đầu trò chơi. Hàm sẽ đặt về mặc địch các giá trị điểm số, thời gian. Sau đó hiển thị bảng số bắt đầu và bắt đầu tính thời gian
- moveAndCheck(int x, int y, int direction): di chuyển ô vị trí (x,y) theo hướng cụ thể.Sau khi di chuyển hàm sẽ gọi lần lượt các hàm kiểm tra xem người chơi có đang chiến thắng hay thua cuộc hay không.
- checkWin(): kiểm tra người chơi có chiến thắng hay không.
- checkLost(): kiểm tra người chơi có thua cuộc hay không.

3.1.14.GameView

GameView có nhiệm vụ lấy thông tin từ MainGame và vẽ lên màn hình. GameView sử dụng canvas để vẽ hình. GameView lưu trữ vị trí của các đối tượng sẽ vẽ lên trên màn hình.

GameView có một số hàm chính:

- makeLayout(): tính toán các vị trí đối tượng khi kích thước ứng dụng thay đổi.
- onSizeChange(): tính toán lại vị trí các tối tượng và tạo lại các tài nguyên cần thiết.
- onDraw(): vẽ các đối tượng lên màn hình.

3.1.15.GameListener

GameListener có nhiệm vụ lấy thông tin người dụng nhập vào, cụ thể là thao tác chạm màn hình. GameListener có hàm chính là onTouch(). Cụ thể hoạt động của hàm onTouch() như sau:

Nếu người dùng bắt đầu chạm vào màn hình: lưu vị trí hiện tại và kiểm tra xem người dùng có chạm vào bất cứ biểu tượng nào hay không.

Nếu người dùng vẫn đang di chuyển: nếu quãng đường di chuyển vượt qua ngưỡng cho trước thì tính hướng di chuyển và nếu lúc bắt đầu chạm người dùng chọn vào ô số thì di chuyển ô đó.

Nếu người dùng thả ra: nếu lúc chạm vào màn hình người dùng chạm vào biểu tượng và lúc thả ra vẫn thả ở vị trí biểu tượng thì gọi chức năng tương ứng của biểu tượng.

3.1.16.MediaPlayerManager

MediaPlayerManager được thiết kế theo mẫu Singleton. MediaPlayerManager có nhiệm vụ đơn giản là chạy nhạc nền có thời lượng dài, tại một thời điểm chỉ chạy một nhạc duy nhất, nếu có nhạc khác yêu cầu chạy thì ngừng nhạc đang phát và chạy nhạc mới.

MediaPlayerManager load nhạc từ bộ nhớ máy nên có độ trễ nhất định, tuy nhiên độ trễ này rất nhỏ và khó nhận biết.

MediaPlayerManager có hàm chính là play(), hàm này bật nhạc có resid cụ thể.

3.1.17.SoundPoolManager

SoundPoolManager được thiết kế theo mẫu Singleton. SoundPoolManager có nhiệm vụ là chạy đoạn nhạc ngắn có thể chồng lên nhau. Điều này làm cho SoundPoolManager rất phù hợp để chạy những đoạn nhạc hiệu ứng ngắn như di chuyển, nhấn nút.

Khác với MediaPlayerManager, SoundPoolManager phải tải nhạc lên RAM trước, nhưng nhờ vậy độ trễ gần như bằng 0. Cần lưu ý bộ nhớ nhạc của SoundPoolManager không được quá 16MB.

Khi tải nhạc lên RAM, cữ mỗi nhạc khi được load lên sẽ trả về một mã số, khi muốn chạy nhạc nào thì phải truyền vào mã đó. Điều này gây phiền hà khi không được dùng resid nhạc dạng R.raw.abc. Chính vì vậy SoundPoolManager sử dụng kiểu dữ liệu SparseIntArray của Java để tạo mối quan hệ giữa resid nhạc và mã số trả về. Khi cần chạy nhạc thì chỉ cần truyền vào resid nhạc.

SoundPoolManager có hàm chính là play(), hàm này bật nhạc có resid cụ thể.

3.1.18.Kết quả thực nghiệm đạt được

Sau khi cài đặt vào máy ảo lẫn trên thiết bị di động cũng như chạy trực tiếp trên desktop, game đã hoạt động tốt, chương trình không bị giật. Hình ảnh và âm thanh nền hợp lý, không có sai sót.

Bố cục của chương trình hợp lí và tự động thay đổi theo kích cỡ màn hình.

CHƯƠNG 4. KẾT LUẬN VÀ HƯỚNG MỞ RỘNG

4.1.1. Kết luận

Đồ án đã giúp cho sinh viên có kiến thức khá tương đối để xây dựng một phần mềm có giao diện bằng ngôn ngữ lập trình Java trên máy tính cũng như thiết bị di động chạy trên hệ điều hành Android.

Nhóm đã biết kết hợp sử dụng nhiều công cụ để phục vụ các công việc khác nhau (lập trình trên IntelliJ IDEA, Android Studio, tạo máy ảo bằng Genymotion/Nox, quản lý code bằng GitHub, build chương trình bằng thiết bị ảo và thật, phân công viết báo cáo bằng Microsoft Word online, sử dụng TeamViewer để hỗ trợ debug) nhằm hỗ trợ hoàn thành đồ án tốt hơn.

Trong quá trình làm việc, các thành viên trong nhóm luôn hỗ trợ lẫn nhau, giúp đỡ những lúc khó khăn.

4.1.2. Hướng mở rộng của chương trình

Thêm phần giới thiệu vào đầu chương trình với các hình ảnh và âm thanh thu hút người chơi.

Độ khó được nâng lên dần dần, người chơi không được phép lựa chọn độ khó của trò chơi.

Ghi nhận số điểm để xếp hạng những số điểm cao nhất trong một lần chơi game.

Phát triển phần gợi ý bước đi ngắn nhất cho người chơi, nhưng đổi lại người chơi bị mất một khoảng thời gian.

4.2. TÀI LIỆU THAM KHẢO

[1]. https://github.com/ovolve/2048-AI

[2]. https://www.becomeasuperlearner.com/courses/become-a-superlearner-masterclass