Project: Zombie Escape

What is that?

It is a game with a goal to escape from zombie by finishing minigame each stage. There are 4 stage in this game. First, there are three moving circles around with alphabets and has a pointer to indicate alphabet, so you need to sequentially stop by press spacebar, to pass this stage three circles must be indicate follow the sequential alphabet above game panel.

Second, this stage is so important because it's only one stage that affect to your coin(score), you need to get coins as you can by press spacebar when ball running in gaps and you need to save the time for run away from zombie.

Third, there are four wire that need to cut the way to cut it so easy just enter spacebar when running ball moving on four gaps.

Finally, there are four sequential alphabets above and there are twenty alphabet boxes, to finish this stage you need to left click on alphabet box that have data same with above by sequential.

So, when you finish all stage the coin will be record. Note that if game is over the coin will not be record and when you do something mistake all zombies velocity will be increase and opportunity to born new zombie with the highest velocity(compare with older zombies).

Player:

- There is only one player.
- Player will gain coin in stage 1.
- Player will be die if zombie is closely.

Zombie:

- Zombie will be born when game start and it has opportunity to born when player do something mistake.
- Zombie will be increase it's velocity when player do something mistake.

Minigame:

- Play the game that appear and finish it to end the game.

Interface render.IRenderable

This interface use to describe base of something that can render

Method:

-public void draw(Graphic2D g2d); use to draw the IRenderable object.

-public void setDestroying(boolean destroyed); use to set that this object is destroyed or not.

-public void update(); use to update this object.

-public void updateAnimation(); update next frame photo;

-public boolean isVisible(); return true if this object is visible and false if it's not.

-public boolean isDestroyed(); return value's destroyed fields.

-public int getX(); get x position.

-public int getY(); get y position.

-public int getZ(); get z position.

Interface LogicGame.Logic

This interface use to describe base of logic class

Method:

-public void logicUpdate(); use to update data of class;

Class entity.AlphabetBox

This class implements from IRenderable use to create a box that contains with alphabet

Field:

#static int NOTHING, CORRECT, INCORRECT; describe object's stage

#int primaryKey; the key that identify each box if it's same primaryKey two boxes will be linked.

#int x, y; the position that you want the box be.

#int width, height; this is describe size of box.

#int selected; it's correct if user click on the box that linked, default value is nothing that mean this is not selected.

#boolean destroyed; it's true if this box was destroyed, default value is false;

-BufferedImage checkMark; keep image that will be happen depend on stage.

-int boxReactionCounter; reaction box counter.

Constructor:

initialize the value of each field.

Method:

-generated getter settle of fields.

-private void drawCorrectBox(Graphics2D g2d); use to draw image that it's correct.

-private void drawCorrectingBox(Graphics2D g2d); use to draw image that it's correcting.

-private void drawIncorrectBox(Graphics2D g2d); use to draw image that it's incorrect.

-public void drawDefaultBox(Graphics2D g2d); use to draw default image.

Class entity.Coin

This class implements from IRenderable use to create a coin that appear in stage 1.

Field:

#int x, y; the position that you want the coin be.

#int radius; radius of this coin.

#int disappearCounter; it's describe how long this coin can appear.

#boolean destroyed; it's true if coin was destroyed, default value is false.

+int seed; just like primary key that linked a coin with a SpacebarGap.

+boolean destroying; it's true when disappearCounter = 0.

-int deadCounter; to count when object close to die.

-BufferedImage coinImage; keep coin image.

Constructor:

initialize the value of each field.

Method:

-generated getter settle of fields.

-public void update(); decrease disappear counter if disappearCounter <= 0 set destroyed is true, then destroyed is true remove on screen object.

Class entity.Gateway

This class implements from IRenderable use to create a gateway.

Field:

#int x, y; the position that you want the gateway be.

#boolean destroyed; it's true if gateway was destroyed, default value is false.

#boolean gateClose; it's false the door is opening;

Constructor:

initialize the value of each field.

Method:

-generated getter settle of fields.

-public void update(); if gateClose is false y will be decrease for open gateway.

Class entity.Player

This class implements from IRenderable use to create a player

Field:

#int x, y; the position that you want the player be.

#int charWidth, charHeight; describe player's size.

#boolean destroyed; it's true if player was died, default value is false.

#boolean destroying; it's true when player close to die.

-int deadCounter; to count when object close to die.

-boolean walking; if it's true player's walking.

-boolean visible; true if it can visible.

-int threadCounter; count the time's thread run.

+AnimationManager animation; define player character.

+AnimationManager animationWalking; define player walking character.

+AnimationManager animationStanding; define player standing character.

+static Object playerLocker; it's synchronize key.

Constructor:

-public Player(); initialize the value of each field, call the method createThread().

Method:

-generated getter settle of fields.

-public void setWalking(boolean walking); if it's walking set it's stand vice versa if it's standing set it's walking.

-public void update(); walking to the position that define each stage if it can't walking set it's standing.

-public boolean collideWith(Zombie zombie); return true if player collide with zombie.

-public void zombieIsComing(); to use sound and flip player by thread.

-public void threadCounterIncrement(); increase threadCounter.

-public void threadCounterDecrement(); decrease threadCounter.

-public void threadCounterReset(); reset thread counter to zero.

-public void createThread(); create player's thread.

Class entity.PlayerStatus

This class implements from IRenderable use to create a player status.

Field:

-int combo; it's counter when you continuously get coin.

Constructor:

initialize the value of each field.

Method:

-generated getter settle of fields.

-public void addScore(int coin); use to add coin.

-public void addCombo(int combo); use to add combo.

-public void comboInterrupted(); reset combo to one.

-public void subtractionScore(); use to subtraction score and note that when score less than zero set it to zero.

Class entity.RunningBall

This class implements from IRenderable use to create a running ball.

Field:

#int x, y; the position that you want the running ball be.

#int xTab, yTab; the position of tab.

#int diameter; diameter of ball.

#int tabDistance; tab distance.

#int direction; define backward or forward direction.

#boolean destroyed; it's true if ball was destroyed, default value is false.

-int shuffleDirectionDelay; delay of redirection.

Constructor:

initialize the value of each field.

Method:

-generated getter settle of fields.

-public void update(); moving the ball by 2 and redirection when it collide with edge of tab. If it's stage 2 shuffle redirection.

Class entity.SpacebarGap

This class implements from IRenderable use to create a spacebar gap.

Field:

#int x, y; the position that you want the gap be.

#int width; width of gap.

#int disappearCounter; define time to disappear.

#int primaryKey; the key that identify each gap if it's same primaryKey it's will be linked.

#boolean destroyed; it's true if gap was destroyed, default value is false.

#Color color; define gap's color.

+int seed; just like primary key that linked a coin with a SpacebarGap.

Constructor:

public SpacebarGap(int primaryKey, Color color, int disappearCounter, int x, int y); initialize the value of each field. if it's stage 0 make it stable on x axis.

Method:

-generated getter settle of fields.

-public void update(); decrease disappearCounter if it less than zero remove it.

Class entity.Wire

This class implements from IRenderable use to create a wire.

Field:

#int x, y; the position that you want the wire be.

#int width, height; wire's size.

#int primaryKey; the key that identify each wire if it's same primaryKey it's will be linked.

#boolean destroyed; it's true if wire was destroyed, default value is false.

#Color color; define wire's color.

Constructor:

initialize the value of each field.

Method:

-generated getter settle of fields.

Class entity.Zombie

This class implements from IRenderable use to create a zombie.

Field:

#int x, y, z; the position that you want the zombie be.

#int charWidth,charHeight; zombie's size.

#boolean destroyed; it's true if zombie was destroyed, default value is false.

#boolean destroying; it's true when zombie close to die.

#int disappearCounter; define time to disappear.

#AnimationManager animation; define zombie character.

-int deadCounter; to count when object close to die.

+boolean moving; use to define that zombie moving or not.

Constructor:

public Zombie(int speed); initialize the value of each field. and random zombie to born.

Method:

-generated getter settle of fields.

-public void update(); moving the zombie.

Class LogicGame.NorthScreenLogic

This class implements from Logic use to calculate events on north screen.

Field:

#int spawnZombieCounter; use to count if it has suitable value zombie can be born.

#boolean firstZombie; use to define this is first zombie.

#Player player; use to define a game's player.

#PlayerStatus playerStatus; use to define game's player status.

#Gateway gateway1, gateway2; use to define gateway.

#ArrayList<Zombie> zombies; use to keep zombies.

-int movingDelayCounter; use to count if it has suitable value zombie will be moving.

-SouthScreenLogic southScreenLogic; use to linked with south screen logic.

+static boolean spawnZombie; if it's true zombie can be born and increase their speed.

-List<IRenderable> list; keep the list of IRenderable object on north screen.

-int nextZombieSpeedUp; keep the next zombie's speed.

Constructor:

public NorthScreenLogic(); initialize the value of each field. and add player, player status, gateway1, gateway2 to RenderableHolder.

Method:

-generated getter settle of fields.

-public void update(); update entities. born zombie if spawnZombie is true but it has time delay for 10 second and move zombie every 0.5 second. if it's stage 4 destroyed all entities except player and player status.

Class LogicGame.SouthScreenLogic

This class implements from Logic use to calculate events on south screen.

Field:

#OpenGatewayZero openGatewayZero; define minigame gatewayzero.

#GetCoin getCoin; define minigame getcoin.

#Passcode passcode; define minigame passcode.

#WireCut wireCut; define minigame wirecut.

-NorthScreenLogic northScreenLogic; linked with north screen logic.

-boolean startStage; if it's true it's start stage.

-List<IRenderable> list; list that keep IRenderable object on south screen.

Constructor:

public SouthScreenLogic(); set startStafe true and add to list.

Method:

-generated getter settle of fields.

-public void update(); if press escape key game will be pause. when startStage is true it will be new object for each stage and add to RenderableHolder. if it's change stage all entities older stage will be remove. and update all minigame.

Class minigame.GetCoin

This class implements from IRenderable use to create a minigame.

Field:

#int xTab, yTab; define tab's position.

#int tabDistance; define tab's distance.

#int direction; define running ball's direction.

#int comboCounter; use to count continuous get coin.

#int disappearCounter; the time that coin is gone and can't get it.

#boolean destroyed; if it's true getCoin with be destroy.

#boolean answer; if you press spacebar in gap it will be true.

#PlayerStatus playerStatus; define player status linked with north screen logic

#RunningBall runningBall; define running ball

#ArrayList<SpacebarGap> gaps; keep all gaps

#ArrayList<Coin> coins; keep all coin

-int spawnDelayCounter; coin born's counter.

Constructor:

public GetCoin(); initialize the value of each field.

Method:

-generated getter settle of fields.

-public boolean enterInGap(SpacebarGap gap); return true if ball is in spacebarGap.

-public void zombieAppear(); set spawnZombie on north screen logic to be true.

-public void update(); update runningBall, gaps, coins. spawn coin and spacebar gap with random position every 1 second. when you press spacebar it's play sound effect if runningBall in the gap you will get coin otherwise zombie will be appear or their speed increase. remove coin and gap if it's enter or destroyed. if you press enter this stage will be pass.

Class minigame.OpenGatewayZero

This class implements from IRenderable use to create a minigame.

Field:

#int xTab, yTab; define tab's position.

#int tabDistance; define tab's distance.

#int direction; define running ball's direction.

#int comboCounter; use to count continuous get coin.

#int answerCounter; increase by 1 if answer is true

#boolean destroyed; if it's true getCoin with be destroy.

#boolean answer; if you press spacebar in gap it will be true.

#PlayerStatus playerStatus; define player status linked with north screen logic

#RunningBall runningBall; define running ball

#SpacebarGap gap; keep all gaps

#Coin coins; keep all coin

Constructor:

initialize the value of each field.

-generated getter settle of fields.

-public void enterSpacebar(); increase answerCounter ans if it's more than or equal three change stage.

-public boolean enterInGap(SpacebarGap gap); return true if ball is in spacebarGap.

-public void zombieAppear(); set spawnZombie on north screen logic to be true.

-public void update(); create new runningBall if it's null. when you press spacebar it's play sound effect if runningBall in the gap it's destroyed otherwise zombie will be appear or their speed increase. update runningBall.

Class minigame.Passcode

This class implements from IRenderable use to create a minigame.

Field:

#ArrayList<AlphabetBox> keyBoxs; keep twenty alphabet boxes

#ArrayList<AlphabetBox> passwords; keep four alphabet boxes

#ArrayList<Integer> randPrimaryKey; keep twenty primary key

#int width,height; define width and height of keyBoxs

#int passwordCounter; count the correct password

Constructor:

public Passcode();initialize the value of each field. generate keyBoxs and passwords with random primary key

Method:

-generated getter settle of fields.

-public boolean isInPressAreaX(); return true if x is in area of all keyBoxs's position x.

-public boolean isInPressAreaY(); return true if y is in area of all keyBoxs's position y.

-public boolean isInPressBoxAreaX(); return true if x is in keyBoxs's area x with same primary key.

-public boolean isInPressBoxAreaY(); return true if y is in keyBoxs's area y with same primary key.

-public void zombieAppear(); set spawnZombie on north screen logic to be true.

-public void update(); if password counter more than 3 change stage. if click mouse check it's on area or not, yes increase passwordCounter, not run method zombieAppear().

Class minigame.WireCut

This class implements from IRenderable use to create a minigame.

Field:

#int x, y; wireframe's position

#int width, height; wireframe's size

#int xTab, yTab; define tab's position.

#int tabDistance; define tab's distance.

#int direction; define running ball's direction.

#int randomX; random x position on spacebarTab.

#ArrayList<Wire> wires; keep all wire.

#RunningBall runningBall; define running ball .

#ArrayList<SpacebarGap> gaps; keep all gaps.

-BufferedImage wireFrameImage; keep wireframe's image.

-AnimationManager wireframeAnimation; keep wireframe's animation.

Constructor:

public WireCut();initialize the value of each field. generate gap and wire with dual primary key.

Method:

-generated getter settle of fields.

-public int randGapX(); random x position on spacebarGap.

-public void zombieAppear(); set spawnZombie on north screen logic to be true.

-public void update(); update runningBall, gaps. when you press spacebar it's play sound effect if runningBall in the gap you will cut wire otherwise zombie will be appear or their speed increase. remove wire and gap if it's enter or destroyed. if all wires are cut change stage.

-public boolean enterInGap(SpacebarGap gap); return true if ball is in spacebarGap.

Class utility.ConfigurableOption

This class use to define variable whole game and make it easy to manage.

Class utility.Debugger

This class use for debug game it'll print class and hashcode.

Field:

-static count; keep sequential print.

Method:

-public static void printTest(Object object); print class name and hash code.

Class utility.InputUtility

This class use to help manage input.

Field:

-static int mouseX, mouseY; keep position when press mouse.

-static boolean mouseLeftDown, mouseRightDown, mouseOnScreen; it's true when you press mouse.

-static boolean mouseLeftTriggered, mouseRightTriggered; it's true when you press mouse for a clock then it's false.

-static boolean prevMouseLeftTriggered, prevMouseRightTriggered; use to keep previuse triggered.

-static boolean[] keyPress; keep if there is press on key board.

-static boolean[] keyTriggered; it's true when you press mouse for a clock then it's false.

Method:

-generated getter settle of fields.

-public static void postUpdate(); set triggered false and keep the old value in previous.

Class utility.RandomUtility

This class use to random number in range

Method:

-public static int random(int s, int e); random number duration s to e.

Class utility.TimeToCounter

This class use to change seond to tick.

Field:

+static int ClockCycle; define time that use for one clock.

Method:

-public static int getCounter(int timeINms); use to change time to tick.

Class score.HighScoreUtility

This class use to record player's rank.

Field:

-String name; keep player's name.

-int score; keep player's score.

Constructor:

initialize the value of each field.

Method:

-generated getter settle of fields.

-private static String[] defaultRecord(); return array of default record.

-public int compareTo(HighScoreRecord o); compare object.

-public static void recordHighScore(int score); record high score if the score is higher than top 10.

-public static void displayTop10(); display top ten score.

-private static boolean loadHighScore(); load high score.

-private static boolean readAndParseScoreFile(File f); use to read and sort score.

-private static boolean createDefaultScoreFile(); create default score file.

-private static String getXORed(String in); use to encryption.

-public static void setReadFileName(String name); set read file name.

Class score.ScoreParsingException

This class extends from exception use to show score parsing exception.

Field:

-int errorType; keep error type.

Constructor:

initialize the value of each field.

Method:

-public String getMessage(); get error message.

Class render.RenderableHolder

This class use to keep all IRenderable class

Field:

-static final RenderableHolder instance; singleton pattern with eagger initialization.

-List northEntities, southEntities; use to keep IRenderable object each position.

Constructor:

initialize the value of each field.

Method:

-generated getter settle of fields.

-public synchronize clear(); remove all object.

-public synchronize void add northEntity(IRenderable entity); add object from north screen to list.

-public synchronize void add southEntity(IRenderable entity); add object from south screen to list.

Class Main.ScreenManager

This class use to manage whole screen.

Field:

+int INTROSCREEN; declare intro screen key

+int SELECTSCREEN; declare select screen key

+int GAMESCREEN; declare game screen key

+int ATTACKSCREEN; declare attack screen key

+int WINNINGSCREEN; declare winning screen key

+int PAUSESCREEN; declare pause screen key

-int FADING; declare fading in method key

-int FADEOUT; declare pause fading out method key

-boolean chagingScreen; key to tell that the screen is changing

- AudioClip bgm : declare background music

- NorthScreen northScreen : declare north screen

- SouthScreen southScreen : declare south screen

- NorthScreenLogic northScreenLogic : declare north screen logic

- SouthScreenLogic southScreenLogic : declare south screen logic

- IntroScreen introScreen : declare intro screen

- PauseScreen pauseScreen : declare pause screen

- GameOverScreen gameOverScreen : declare gameover screen

- JFrame MainFrame : declare Main frame

- JPanel panelInsideFrame : declare panel that will contain in main frame

currentScreen : ArrayList<JComponent>

currentLogic : ArrayList<Logic>

Constructor:

-public void ScreenManager(); setup MainFrame and panel inside MainFrame then put this panel into MainFrame ,loop the game process such as screen and logic method.

Method:

-generated getter settle of fields.

-public void resetScreen(); create new every screen object

-public void changeScreen(int); method that manage how to change between screen.

-public void fadeScreen(JPanel, boolean) method that mange how to fade between screen.

-public void addListener(JPanel) add mouse and keyboard listener to an argument panel.

Class render.AnimationManager

This class used to collect Animation and image data that used to play or show in this project.

Field:

+s int DONOTTHING : primarykey that used to told do not thing in every special method.

+s int FlipToUsual : primarykey that used to told flip method to flip animation to usual side.

+s int FlipToUnUsual : primarykey that used to told flip method to flip animation to the opposite side from usual.

+s int FLIP : primarykey that used to told flip method to flip animation to the other side from now.

+s int RotateRight : primarykey that used to told flip method to flip animation to rotate right.

+s int RotateLeft : primarykey that used to told flip method to flip animation to rotate left.

+s int BufferOPTIMIZED : primarykey that used to told render.ImageReader.get(int) method to load image with special method for “optimized gif”.

-s Boolean isPlay : contain this animation play status that play or stop

-s Boolean isLoop : contain data for loop if true means looping

- boolean isFinish : contain data for told that is this animation is finish playing or not.

- int frame : told index of current frame animation .

- ImageData[]img : contain image data by ImageData object.

- int width : told this animation width.

- int height : told this animation height.

- int delayTime : told this animation delay time between each frame in millisecond

- int delayCounter : counter that's used to count when call update method .

- int charWidth : contain this character width.

- int charHeight : contain this character height.

- int setX : contain this character's foot in x-coordinate.

- int setY : contain this character's foot in y-coordinate

- int flipInfo : data to collect this animation flip status (false means animation was not flipped and vice versa).

Constructor:

public WireCut();initialize the value of each field. generate gap and wire with dual primary key.

Method:

-public void flip(int): flip animation with primary key.

-public void flip(): flip animation to the another way.

-public void flipToUnUsual(): flip animation to unusual side.

-public void flipToUsual(): flip animation to normal side.

-public void flipImage(): flip animation to the other side.

-public int getCharWidth(): get character width.

-public int getCharHeight(): get character height.

-public int getCharWidthByHeight(int): get character width by calculate form height's character argument to scale by original ratio.

-public int getCharHeightByWidth(int): get character height by calculate form width's character argument to scale by original ratio.

-public int getSetX(): get character foot's x-coordinate location .

-public int getSetY(): get character foot's y-coordinate location .

-public int getFrame(): get this number of all frames.

-public int getWidth(): get this animation's image width.

-public int getHeight(): get this animation 's image height.

-public int getWidthByHeight(int): get image width by calculate form height's image argument to scale by original ratio.

-public int getHeightByWidth(int): get image height by calculate form width's image argument to scale by original ratio.

-public void play(): told this animation to play.

-public void loop(): told this animation to loop.

-public void stop(): told this animation to stop playing.

-public boolean isFinish(): get boolean that told is this animation has been finish play.

-public void setFinish(boolean): set this animation to finish play or not.

-public void update(): call this method to update this animation frame.

-public ImageData[] getAllImage(): get All image data in ImageData array object.

-public ImagteData getCurrentImageData(): get current image data in ImageData object.

-public ImagteData getCurrentImageData(int): get image data by index argument in ImageData object.

-public BufferdImage getCurrentBufferedImage(): get current image data in BufferdImage object.

-public BufferdImage getCurrentBufferedImage(int): get image data by index argument in BufferdImage object.

-public Object clone(): throw CloneNotSupportedException to told that this object cannot be cloned.

Class render.RenderAnimationHelper

This class used to help developer to draw Animation easier by draw animation like **RenderHelper class** but this class draw with ratio of character width and height.

Field:

#int x, y; the position that you want the player be.

#int charWidth, charHeight; describe player's size.

Constructor:

- None.

Method:

-public static void draw (Graphics2D g2d,AnimationManager animation,int x,int y,int userCharWidth,int userCharHeight): used to help developer to draw animation on Graphics2D which coordinate x and y is position that point to character's foot position and it can scale the animation by userCharWidth or userCharHeight argument which you can leave one width or height to draw with width or height relate to its original character ratio.

Class render. RenderHelper

This class used to help developer to draw Animation easier by draw animation with special key word to told this class to draw image in which position relate to x and y argument.

Field:

+staic int LEFT, CENTER, RIGHT , TOP, MIDDLE , BOTTOM, REPEAT : special key word to told which position to draw image that relate with x, y coordinate.

Constructor:

- None.

Method:

- public static void draw(Graphics2D g, BufferedImage img, int x, int y, int width, int height, int position) : throw all arguments to next draw(...) method which send event as null.

- public static void draw(Graphics2D g, BufferedImage img, int x, int y, int width, int height, int position,RenderHelperMouseEvent event) : draw BufferedImge on Graphics2D which draw by used special key word to told which position to draw image that relate to x, y coordinate argument and check that is event has object or not if yes then call check event method.

- private static void checkEvent(RenderHelperMouseEvent event, int x, int y, int width, int height) : check event and handle event by event in RenderHelperMouseEvent object.

- private static boolean isMouseEntered(int x,int y,int width,int height) : check that is user's mouse position has been in image area.

- public static void addAntiAlising(Graphics2D g2d) : add anti-alising in Graphics.

- public static void removeAntiAliasing(Graphics2D g2d) : remove anti-alising in Graphics.

Class render.RenderHelperMouseEvent

This is an abstract class that use to make mouse event in RenderHelper class.

Field:

- none.

Constructor:

- none.

Method:

- public abstract void mouseClicked(); call this event when mouse has been clicked on image area.

- public abstract void mousePressed(); call this event when mouse has been pressed on image area.

- public abstract void mouseReleased();call this event when mouse has been released on image area.

- public abstract void mouseEntered();call this event when mouse has been moved in image area.

- public abstract void mouseExited();call this event when mouse has been moved out image area.

Class resource. FethResourceException

This class extends from Exception use to throw Exception type with message by error type when get Animation from ArrayList by wrong key.

Field:

+ static final int ANIMATION ; primary key for animation error type.

+ static final int AUDIO ; primary key for audio error type.

- int errorType; used to collect error type.

- String url; used to collect url of error file.

Constructor:

- Public FethResourceException(int errorType,String url) : Get and set the value of each field.

Method:

- public String getMessage(); get error message.

Class resource. GetResourceException

This class extends from Exception use to throw Exception type with message by error type when class loader get resource and that file was not found.

Field:

+ static final int ANIMATION ; primary key for animation error type.

+ static final int AUDIO ; primary key for audio error type.

- int errorType; used to collect error type.

- String url; used to collect url of error file.

Constructor:

- Public FethResourceException(int errorType,String url) : Get and set the value of each field.

Method:

- public String getMessage(); get error message.

Class resource.Resource

This class declare and mange every resources that used in this project.

Field:

- static HashMap<String,AnimationManager> rs = new HashMap<>(); contain Animation resource.

- static HashMap<String,AudioClip> audio = new HashMap<>(); contain audio resource.

+ static final Font standardFont : contain font style.

+ static BufferedImage[] playButton :

+ BufferedImage[] rankButton : BufferedImage[]

+ BufferedImage[] continueButton : BufferedImage[]

+ BufferedImage[] exitButton : BufferedImage[]

+ BufferedImage[] restartButton : BufferedImage[]

+static Cursor CURSOR_DEFAULT : Cursor

+static Cursor CURSOR_HAND : Cursor

Constructor:

initialize the resource file location and put them to ArrayList.

Method:

- public AnimationManager read(String url,int setX,int setY,int setCharWidth,int setCharHeight,int mode) throws FethResourceException : get AnimationManager object by its url

location with set its x,y foot location , character width and height and special primary keyword which told ImageReader class to get gif file with OPTIMIZED mode or not.

- public AnimationManager read(String, int, int, int, int): return AnimationManager like the others method in previous mention.

- public AnimationManager read(String, int) : return AnimationManager like the others method in previous mention.

- public AnimationManager read(String) : return AnimationManager like the others method in previous mention.

- public AudioClip audioRead(String) : return AudioClip by string keyword.

- public static AnimationManager get(String) : return AnimationManager by string keyword.

- public static BufferedImage getImage(String) : return BufferedImage by string keyword.

- public static BufferedImage getImage(String, int) : return BufferedImage by string keyword and its index of frame.

- public static AudioClip getAudio(String) : return AudioClip by string keyword.

Class ui.GameOverScreen

This class extends from JComponent use to display Game over screen.

Field:

- AnimationManager BG; Background animation.

- BufferedImage img; Background current BufferedImage.

- int width,height; Screen width and height.

Constructor:

initialize the value of each field and setup components.

Method:

- protected void paintComponent(Graphics g): draw components.

- private void drawStartBT(Graphics2D g, BufferedImage img, int x, int y, int width, int height, int position): method that help developer to draw button on Graphics2D and setup its mouseEvent.

Class ui.IntroScreen

This class extends from JComponent use to display intro screen.

Field:

- AnimationManager BG; Background animation.

- BufferedImage img; Background current BufferedImage.

- int width,height; Screen width and height.

Constructor:

initialize the value of each field and setup components.

Method:

- protected void paintComponent(Graphics g): draw components.

- private void drawStartBT(Graphics2D g, BufferedImage img, int x, int y, int width, int height, int position): method that help developer to draw button on Graphics2D and setup its mouseEvent.

- private void drawRankBT(Graphics2D g, BufferedImage img, int x, int y, int width, int height, int position): method that help developer to draw button on Graphics2D and setup its mouseEvent.

Class ui. NorthScreen

This class extends from JComponent use to display north screen of game.

Field:

- AnimationManager bgAnimation; Background animation.

- BufferedImage img; Background current BufferedImage.

- int width,height, statusHeight; Screen width , height and status height.

- ArrayList<IRenderable> entity; contain north renderable list.

Constructor:

initialize the value of each field and setup components.

Method:

- protected void paintComponent(Graphics g): draw components.

Class ui. PauseScreen

This class extends from JComponent use to display pause screen.

Field:

- AnimationManager BG; Background animation.

- BufferedImage img; Background current BufferedImage.

- int width,height; Screen width and height.

Constructor:

initialize the value of each field and setup components.

Method:

- protected void paintComponent(Graphics g): draw components.

- private void drawContinueBT (Graphics2D g, BufferedImage img, int x, int y, int width, int height, int position): method that help developer to draw button on Graphics2D and setup its mouseEvent.

- private void drawReStartBT (Graphics2D g, BufferedImage img, int x, int y, int width, int height, int position): method that help developer to draw button on Graphics2D and setup its mouseEvent.

- private void drawExitBT (Graphics2D g, BufferedImage img, int x, int y, int width, int height, int position): method that help developer to draw button on Graphics2D and setup its mouseEvent.

Class ui. SouthScreen

This class extends from JComponent use to display south screen of game.

Field:

- AnimationManager bgAnimation; Background animation.

- BufferedImage img; Background current BufferedImage.

- int width,height, statusHeight; Screen width , height and status height.

- ArrayList<IRenderable> entity; contain south renderable list.

Constructor:

initialize the value of each field and setup components.

Method:

- protected void paintComponent(Graphics g): draw components.

Class ui. WinningScreen

This class extends from JComponent use to display winning screen.

Field:

- AnimationManager BG; Background animation.

- BufferedImage img; Background current BufferedImage.

- int width,height; Screen width and height.

Constructor:

initialize the value of each field and setup components.

Method:

- protected void paintComponent(Graphics g): draw components.

- private void drawReStartBT (Graphics2D g, BufferedImage img, int x, int y, int width, int height, int position): method that help developer to draw button on Graphics2D and setup its mouseEvent.

Class ui.IntroScreen

This class extends from JComponent use to display intro screen.

Field:

- AnimationManager BG; Background animation.

- BufferedImage img; Background current BufferedImage.

- int width,height; Screen width and height.

Constructor:

initialize the value of each field and setup components.

Method:

- protected void paintComponent(Graphics g): draw components.

- private void drawStartBT(Graphics2D g, BufferedImage img, int x, int y, int width, int height, int position): method that help developer to draw button on Graphics2D and setup its mouseEvent.

- private void drawRankBT(Graphics2D g, BufferedImage img, int x, int y, int width, int height, int position): method that help developer to draw button on Graphics2D and setup its mouseEvent.