Part 1 Game Introduction

In November 2078, the alien specie call “Metaqaur” come to planet Earth. They land on area section 25, where is outside of the city magnetic shield. You are the general, who is the only hope to prevent the Metaqaur from teleport gate, the way to enter the city. So make decistion and command the tower to protect the city, protect the human kind.

Part 2 User manual

Metaqaur is a tower defense game type. In this game type you are building tower to defense enemy from their destination. If you let the enemy reach the destination you will lose your health, if your health equal 0 or less than you will lose the game. You can earn money by kill the enemy and use those money to upgrade or build more tower.

When program start, first scene is loading scene, after all resources have been loading the game will go to menu page, which can connect to 2 page that is help page and game state. In help page there is a button that link to menu.

Once you are click play button in menu, program will go to game state. In game state there are 2 button, first one is pause button and another one is home button. If you click home button, program will go to menu and reset all game.

If you play the game until the end, the program will show end screen for a moment and then go to menu. But if you lose, the program will show text “game over” and delay for a moment before go to menu page also.

In game state, there are 3 canvas. First one at top is a bar, where show a current time of each wave, your current health and money, current stage and wave and include pause button and home button. Second canvas is a information window where locate on the right of program. Information window will show data of tower that you click and show data of creep in case that you right click or you are not click any tower. Last canvas is game screen, where all the creep and tower locate. In this canvas if you click an empty tile it will show build option, which has 3 type of tower to build, but if you click on tower, it will show upgrade option to upgrade your tower, upgrade option feature will depend on type of tower and level of your tower.

Part 3 Implementation Detail

1. Package application
    1.1. Class Main
        1.1.1. Field

- GameLogic gameLogic	The GameLogic component.
- Bar bar	The Bar component.
- Loading loading	The Loading component.
- GameScreen gameScreen	The GameScreen component.
- Help help	The Help component.
- InfoWindow infoWindow	The InfoWindow component.
- Menu menu	The Menu component.
- End end	The End component.
- HBox body	The HBox pane.
- VBox game	The VBox pane.
- StackPane loadingScreen	The StackPane use for loading screen.

- StackPane mainMenu	The StackPane use for menu page.
- StackPane helpScreen	The StackPane use for help page.
- StackPane endScreen	The StackPane use for end screen.
- double delayTime	Delay time.
- double gap	Use for calculate time.
- long currentTime	Use for calculate time.
- Thread sound	Thread for play background music.
- boolean fromPause	True if before state is pause.

1.1.2. Method

+ void tick()	Update method for Bar.
+ void paintComponent()	Draw method Bar.
+ void paintWave(int wave)	Paint current wave.
+ boolean onPauseButton()	Return true if mouse on pause button.
+ boolean onHomeButton()	Return true if mouse on home button.
+ boolean isBackHome()	Getter of backHome.
+ void addListerner()	Add listerner for sent value to InputUtility.

2. Package drawing

2.1. Class Bar

2.1.1. Field

- GameLogic gameLogic	The GameLogic component.
- boolean backHome	True if click home button.

2.1.2. Constructor

+ Bar(double width, double height, GameLogic gameLogic)	- Initialize fields from parameters. - Add listerner

	- Set ***visible to true.*** - Set ***backHome*** to false.

2.1.3. Method

+ void tick()	Update method for Bar.
+ void paintComponent()	Draw method Bar.
+ void paintWave(int wave)	Paint current wave.
+ boolean onPauseButton()	Return true if mouse on pause button.
+ boolean onHomeButton()	Return true if mouse on home button.
+ boolean isBackHome()	Getter of backHome.
+ void addListerner()	Add listerner for sent value to InputUtility.

2.2. Class End

2.2.1. Field

- GameLogic gameLogic	The GameLogic component.
- double delay	Use for delay before go to main menu.

2.2.2. Constructor

Void End(double width, double height,GameLogic gameLogic)	- Initialize fields from parameters. - Set ***visible to true.*** - Set ***delay*** equal -1.

2.2.3. Method

+ void tick()	Update method for End.
+ void paintComponent()	Draw method End.

2.3. Class GameScreen

2.3.1. Field

- GameLogic gameLogic	The GameLogic component.

2.3.2. Constructor

Void GameScreen(double width, double height,GameLogic gameLogic)	- Initialize fields from parameters. - Add listerner - Set ***visible to true.***

2.3.3. Method

+ void paintComponent()	Draw method IceTower.
Void addListerner()	Add listerner for sent value to InputUtility.

2.4. Class Help

2.4.1. Field

- GameLogic gameLogic	The GameLogic component.
- boolean soundBackButton	True if click back button.

2.4.2. Constructor

+ Help(double width, double height,GameLogic gameLogic)	- Initialize fields from parameters. - Add listerner - Set ***visible to true.*** - Set ***soundBackButton*** to false.

2.4.3. Method

+ void tick()	Update method for IceTower.

+ void paintComponent()	Draw method IceTower.
+ boolean onBackButton()	Return true if mouse on back button.
+ void addListerner()	Add listerner for sent value to InputUtility.

2.5. Class InfoWindow

2.5.1. Field

- GameLogic gameLogic	The GameLogic component.
- Image image	Image of creep or tower.

2.5.2. Constructor

+ InfoWindow(int width, int height, GameLogic gameLogic)	- Initialize fields from parameters.

2.5.3.

2.5.4. Method

+ void paintComponent()	Update method for IceTower.
+ void tick()	Draw method IceTower.
+ void addListerner()	Add listerner for sent value to InputUtility.

2.6. Class Loading

2.6.1. Field

- GameLogic gameLogic	The GameLogic component.
- int deg1	Degree value for rotate loading picture.
- int deg2	Degree value for rotate loading picture.
- int deg3	Degree value for rotate loading picture.
- int count	Use for select draw text.
- long textTime	Use for delay text show duration.

- long loadingTime	Use for delay loading time, which from game to menu.

2.6.2. Constructor

+ Loading(double width, double height,GameLogic gameLogic)	- Initialize fields from parameters. - Set deg1, deg2, deg3 equal 5, 4, 3. - Set textTime to -1. - Set loadingTime to -1. - Set count to 1.

2.6.3. Method

+ void tick()	Update method for IceTower.
+ void paintComponent()	Draw method IceTower.
+ void paintLoadingScreen()	Paint loading picture.
+ void paintText(int textCase)	Paint text “loading” or “stage”.
+ void addListerner()	Add listerner for sent value to InputUtility.

2.7. Class Bar

2.7.1. Field

- GameLogic gameLogic	The GameLogic component.
- boolean soundPlayButton	True if click play button.
- boolean soundHelpButton	True if click help button.

2.7.2. Constructor

+ Menu(double width, double height,GameLogic gameLogic)	- Initialize fields from parameters. - Add listerner - Set ***visible to true.*** - Set ***soundplayButton*** to false.

	- Set ***soundHelpButton*** to false.

2.7.3. Method

+ void tick()	Update method for Menu.
+ void paintComponent()	Draw method Menu.
+ boolean onPlayButton()	Return true if mouse on play button.
+ boolean onHelpButton()	Return true if mouse on help button.
+ void addListerner()	Add listerner for sent value to InputUtility.

3. Package input

3.1. Class AudioUtility

3.1.1. Field

3.1.2. Method

+ void playSound(String sound)	Play sound clip.
+ void stopSound(String sound)	Stop sound clip.

3.2. Class OverrideableEntity

3.2.1. Field

# double speed	Speed value.
# double cX	x value at center.
# double cY	y value at center.

3.2.2. Method

# abstract Shape getBound()	Return shape.
# boolean overrideWith(OverrideableEntity other)	Return true if override with object ***other***.
+ void setSpeed(double speed)	Setter of value ***speed***.

3.3. Method Class InputUtility

3.3.1. Field

+ double mouseX	Location of mouse in x axis.
+ double mouseY	Location of mouse in y axis.
+ boolean mouseOnGameScreen	True if mouse enter GameScreen.
+ boolean mouseOnBar	True if mouse enter Bar.
+ boolean mouseOnMenu	True if mouse enter Menu.
+ boolean mouseOnHelp	True if mouse enter Help.
+ boolean mouseOnInfoWindow	True if mouse enter InfoWindow.
- boolean isLeftDown = false	True if left click.
- boolean isLeftClickedLastTick	True if left click.
- boolean isLeftRelease	True if release left click.
- boolean isRightDown	True if right click.
- boolean isRightClickedLastTick	True if right click.
- boolean isRightRelease	True if release right click.

3.3.2. Method

+ void mouseLeftDown()	Set isLeftDown and isLeftClickedLastTick to true.
+ void mouseLeftRelease()	Set isLeftDown to false and isLeftRelease to true.
+ boolean isLeftClickTriggered()	Return isLeftClickedLastTick.
+ boolean isLeftClickRelease()	Return isLeftRelease.
+ void mouseRightDown()	Set isRightDown and isRightClickedLastTick to true.
+ void mouseRightRelease()	Set isRightDown to false and isRightRelease to true.

+ boolean isRightClickTriggered()	Return isLRightClickedLastTick.
+ boolean isRightClickRelease()	Return isRightRelease.
+ void updateInputState()	Set isLeftClickedLastTick, isLeftRelease, isRightClickedLastTick and isRightRelease to false.

4. Package logic

4.1. Class GameLogic

4.1.1. Field

- List<Entity> gameObjectContainer	List of object in game for update logic.
- Spawn spawn	The Spawn component.
- FieldOption fieldOption	The FieldOption component.
- long now	Current time.
- int nextStage	Value of next stage.
- boolean reset	True if game is going to reset.
- boolean isPause	True if game is push.
- int time	Current Time of each wave.
- int wave	Current wave.
- int money	Current money.
- int stage	Current stage.
- int health	Current health.
- boolean gameOver	True if health less than 0.
+ enum STATE	State of game.
- STATE gameState = STATE.Loading	Current state.

4.1.2. Method

+ GameLogic()	- Initialize ***gameObjectContainer*** as new Arraylist. - Initialize ***field***. - Initialize ***spawn.*** - Initialize ***fieldOption.*** - set ***stage*** equal 0. - set ***reset*** to false. - set ***isPause*** to false. - set ***health*** equal 20. - set ***wave*** equal 1.

4.1.3. Method

+ void logicUpdate()	Do if game state equal game - tick() every object in ***gameObjectContaine.*** - if isDestroy() of object is true, remove object.
# void addNewObject(Entity entity)	Add object entity to ***gameObjectContaine*** and add to ***entities*** in ***RenderableHolder.***
+ List<Entity> getGameObjectContainer()	Getter of value ***gameObjectContaine.***
+ Field getField()	Getter of value ***Field.***
+ FieldOption getFieldOption()	Getter of value ***FieldOption.***
+ long getNow()	Getter of value ***now.***
+ void setNow(long now)	Setter of value ***now.***
+ Spawn getSpawn()	Getter of value ***spawn.***
+ int getNextStage()	Getter of value ***nextStage.***
+ boolean isReset()	Getter of value ***reset.***
+ void setReset(boolean reset)	Setter of value ***reset.***
+ void setLoadStage(int nextStage)	Set game state to loading and set nextStage.

+ boolean isPause()	Getter of value ***pause.***
+ void setPause(boolean isPause)	Setter of value ***pause.***
+ int getTime()	Getter of value ***time.***
+ void setTime(int time)	Setter of value ***time.***
+ int getWave()	Getter of value ***wave.***
+ void setWave(int wave)	Setter of value ***wave.***
+ int getMoney()	Getter of value ***money.***
+ void setMoney(int money)	Setter of value ***money.***
+ int getStage()	Getter of value ***stage.***
+ void setStage(int stage)	Setter of value ***stage.***
+ int getHealth()	Getter of value ***health.***
+ void setHealth(int health)	Setter of value ***health.***
+ boolean isGameOver()	Getter of value ***gameOver.***
+ void setGameOver(boolean gameOver)	Setter of value ***gameOver.***
+ STATE getGameState()	Getter of value ***GameState.***
+ void setGameState(STATE gameState)	Setter of value ***GameState.***

4.2. Class Entity

4.2.1. Field

# double x	Location x of component.
# double y	Location y of component.
# int z	Value ***z*** use for sort entities collection.
# boolean visible	True if visible.
# boolean destroyed	True if destroy.

4.2.2. Constructor

# Entity()	Set ***visible*** to true and set ***destroy*** to false.

4.2.3. Method

+ boolean isDestroyed()	Getter of ***destroy***.
+ boolean isVisible()	Getter of ***visible***.
+ int getZ()	Getter of ***z***.

4.3. Class Creep

4.3.1. Field

# double hp	Hitpoint of creep.
# double speed	Speed of creep.
# double maxSpeed	Max speed of creep.
# double maxHp	Max hitpoint of creep.
# double armor	Armor of creep. Creep will not get hit if damage less than or equal armor (except damage is from laser tower).
# int price	Value of money that player will earn after this creep dead.
# Spawn spawn	The Spawn component
# GameLogic gameLogic	The GameLogic component
# double rotateSpeed	Speed of rotating of creep.
# double distance	Distance of 1 tile of field.
# String name	Name of creep.

4.3.2. Constructor

+ Creep(String name, double x,double y, double speed, double rotateSpeed, double hp, double armor, int price,Spawn spawn,GameLogic gameLogic)	- Initialize fields from parameters. - Set distance equal field.SIZE. - Set z equal 3.

4.3.3. Method

+ void checkHp(double val, double max,double min)	Check hitpoint of creep is lessthan min set it equal min, if more than max set it equal max.
+ abstract List creepData()	Abstract method that return list of data of creep.
+ double getHp()	Getter of value ***hp***
+ void setHp(double hp)	Setter of value ***hp***
+ void setDestroy()	Setter of value ***destroy***
+ void moveUp()	If ***distance*** more than 0, decrease ***y*** and distance by ***speed***.
+ void moveRight()	If ***distance*** more than 0, increase ***x*** and distance by ***speed***.
+ void moveDown()	If ***distance*** more than 0, increase ***y*** and distance by ***speed***.
+ void moveLeft()	If ***distance*** more than 0, decrease ***x*** and distance by ***speed***.

4.4. Class NormalCreep

4.4.1. Field

- List<OrbitEntity> overrideWith	List of OrbitEntity that override with this NormalCreep.
- int path	Path of creep. Path = 1 is mean move up. Path = 2 is mean move right. Path = 3 is mean move down. Path = 4 is mean move left.

- int dimension	Dimention of creep. Use for calculate if this creep override with OrbitEntity or not.
- int direction	Direction of creep in degree (0 is +x axis).
- int rotate	Direction that creep need to be rotate.
- boolean isRotate	Boolean value use for control that this NormalCreep need to rotate or not.
- boolean isSlow	Boolean value use for control that this NormalCreep is get slow or not.
- long lastHitSlow	The value that store time when this NormalCreep get slow last time.
- double slowBy	Percent of that this NormalCreep get slow.
- Image creep	Image of NormalCreep.

4.4.2. Constructor

+ NormalCreep(String name, double x,double y, double speed, double rotateSpeed, double hp, double armor, int price,Spawn spawn,GameLogic gameLogic, String image)	- Initialize fields from parameters. - Initialize ***overideWith*** as new Arraylist. - Set ***path*** equal 0. - Set ***direction*** equal 0. - Set ***dimension*** equal 30. - Set ***rotate*** equal ***direction***. - Set ***isRotate*** to false. - Set ***isSlow*** to false. - Set ***lastHitSlow*** equal 0. - Set ***slowBy*** equal 1. - Set ***z*** equal 10. - Set ***image*** using setImage(String) method.

4.4.3. Method

+ void tick()	Update method for NormalCreep.
+ void draw(GraphicsContext gc)	Draw method for NomalCreep.
+ void setImage(String image)	Set Image of NormalCreep.
# Shape getBound()	Return override shape of NormalCreep.
+ void getHit(OrbitEntity o)	Check that ***overrideWith*** contain OrbitEntity ***o*** or not, if true increase ***hp*** by (***armor***-***damage*** of ***o***).
+ List creepData()	Return list of string name, hp, speed, armor and rotate speed.

4.5. Class Tower

4.5.1. Field

# GameLogic gameLogic	The GameLogic component.
# int amount	Amount of orbitEntity of Tower.
# int attackLevel	Attack level of Tower.
# double speed	Speed of Tower.
+ final int MAX_AMOUNT	Maximum value of amount.
+ final int MAX_ATTACK_LEVEL	Maximum value of attack level.

4.5.2. Constructor

+ Tower(int x, int y,GameLogic gameLogic, double speed, int amount, int attackLevel)	- Initialize fields from parameters. - set ***z*** equal 5.

4.5.3. Method

+ abstract List towerData()	Abstract method that return list of data of creep.

+ abstract void destroyTower()	Abstract method of destroyTower.

4.6. Class NormalTower

4.6.1. Field

+ final int price	Constant value price of NormalTower.
- List<OrbitEntity> orbitEntities	List of OrbitEntity of NormalTower.

4.6.2. Method

+ NormalTower(int x, int y,GameLogic gameLogic, double speed, int amount, int attackLevel)	- Initialize fields from parameters. - Initialize ***orbitEntities*** as new Arraylist. - add OrbitEntity use addOrbitEntities() method. - set attack level by using setAttackLevel(int attackLevel) method.

4.6.3. Method

+ void tick()	Update method for NormalTower.
+ void draw(GraphicsContext gc)	Draw method for NomalTower.
+ void addOrbitEntities()	Add OrbitEntity by 1 if ***amount*** < ***MAX_AMOUNT***.
+ void destroyTower()	- Set destroy to true. - Set destroy of all orbitEntity of this tower to true.
+ int getAmount()	Getter of value ***amount.***
+ void setAmount(int amount)	Setter of value ***amount.***
+ int getAttackLevel()	Getter of value ***attackLevel.***

+ void setAttackLevel(int attackLevel)	Setter of value ***attackLevel.***
+ int getUpgradeAmountPrice(int upgradelevel)	Return upgrade price depend on current ***amount.***
+ int getUpgradeAttackPrice(int upgradelevel)	Return upgrade price depend on current ***attackLevel.***
+ List towerData()	Return list of data of this NormalTower.

4.6.4.

4.7. Class IceTower

4.7.1. Field

+ final int price	Constant value price of IceTower.
+ final int MAX_SLOW_LEVEL	Maximum value of slow level.
- List<SlowOrbitEntity> orbitEntities	List of SlowOrbitEntity of IceTower.
- int slowLevel	Slow level of IceTower.

4.7.2. Constructor

+ IceTower(int x, int y,GameLogic gameLogic, double speed, int amount, int attackLevel,int slowLevel)	- Initialize fields from parameters. - Initialize ***orbitEntities*** as new Arraylist. - add OrbitEntity use addOrbitEntities() method. - set attack level by using setAttackLevel(int attackLevel) method.

4.7.3. Method

+ void tick()	Update method for IceTower.
+ void draw(GraphicsContext gc)	Draw method IceTower.
+ void addOrbitEntities()	Add OrbitEntity by 1 if ***amount*** < ***MAX_AMOUNT***.
+ void destroyTower()	- Set destroy to true.

	- Set destroy of all SlowOrbitEntity of this tower to true.
+ int getAmount()	Getter of value ***amount.***
+ void setAmount(int amount)	Setter of value ***amount.***
+ int getAttackLevel()	Getter of value ***attackLevel.***
+ void setAttackLevel(int attackLevel)	Setter of value ***attackLevel.***
+ int getSlowLevel()	Getter of value ***slowLevel.***
+ void setSlowLevel(int slowLevel)	Setter of value ***slowLevel.***
+ int getUpgradeSlowPrice(int upgradelevel)	Return upgrade price depend on current ***slowLevel.***
+ int getUpgradeAttackPrice(int upgradelevel)	Return upgrade price depend on current ***attackLevel.***
+ List towerData()	Return list of data of this IceTower.

4.8. Class LaserTower

4.8.1. Field

+ final int price	Constant value price of IceTower.
- String direction	Direction of laser tower (right or left).
- StraightEntity o	The StraightEntity component

4.8.2. Method

+ LaserTower(int x, int y,GameLogic gameLogic, double speed, int amount, int attackLevel, String direction)	- Initialize fields from parameters. - Initialize StraightEntity.

4.8.3. Method

+ void tick()	Update method for LaserTower.
+ void draw(GraphicsContext gc)	Draw method for LaserTower.

+ void destroyTower()	- Set destroy of StraightEntity to true. - Set destroy to true.
+ int getAttackLevel()	Getter of value ***attackLevel.***
+ void setAttackLevel(int attackLevel)	Setter of value ***attackLevel.***
+ int getUpgradeAttackPrice(int upgradelevel)	Return upgrade price depend on current ***attackLevel.***
+ List towerData()	Return list of data of this LaserTower.

4.9. Class OrbitEntity

4.9.1. Field

# int orbitRange	Orbit range of OrbitEntity.
# int damage	damage of OrbitEntity.
# double angle	angle of OrbitEntity, use for calculate orbit position.
# double locationX	locationX of OrbitEntity, use for calculate orbit position.
# double locationY	locationY of OrbitEntity, use for calculate orbit position.

4.9.2. Method

+ void setDamage(int damage)	Setter of value ***damage.***

4.10. Class NormalOrbitEntity

4.10.1. Field

- int deg;	degree of NormalOrbitEntity, use for calculate orbit position.
- double x1	X value of NormalOrbitEntity, use for calculate orbit position.

- double y1	Y value of NormalOrbitEntity, use for calculate orbit position.
- int dimension	Dimention of NormalOrbitEntity. Use for calculate if this NormalOrbitEntity override with Creep or not.

4.10.2. Method

+ NormalOrbitEntity(int x, int y, int orbitRange, int posiotion)	- Initialize fields from parameters. - set some value for calculating.

4.10.3. Method

+ void tick()	Update method for NormalOrbitEntity.
+ void draw(GraphicsContext gc)	Draw method for NormalOrbitEntity.
# Shape getBound()	Return override shape of NormalOrbitEntity.
+ void orbit()	Calculate new position of x and y.

4.11. Class SlowOrbitEntity

4.11.1. Field

- int deg	degree of NormalOrbitEntity, use for calculate orbit position.
- double x1	X value of NormalOrbitEntity, use for calculate orbit position.
- double y1	Y value of NormalOrbitEntity, use for calculate orbit position.
- int dimension	Dimention of NormalOrbitEntity. Use for calculate if this NormalOrbitEntity override with Creep or not.
- double slow	Slow value of SlowOrbitEntity.

4.11.2. Constructor

+ SlowOrbitEntity(int x, int y, int orbitRange, int posiotion,double speed)	- Initialize fields from parameters. - set some value for calculating. - set ***slow*** value equal 35.

4.11.3. Method

+ void tick()	Update method for SlowOrbitEntity.
+ void draw(GraphicsContext gc)	Draw method for SlowOrbitEntity.
# Shape getBound()	Return override shape of SlowOrbitEntity.
+ void orbit()	Calculate new position of x and y.
+ double getSlow()	Getter of value ***slow.***
+ void setSlow(double slow)	Setter of value ***slow.***

4.12. Class StraightEntity

4.12.1. Field

- GameLogic gameLogic	The GameLogic component.
- String direction	Direction of StraightEntity (right or left).
- int width	Width of StraightEntity.
- double damage	Damage of StraightEntity.

4.12.2. Method

+ StraightEntity(int x, int y, String direction, GameLogic gameLogic)	- Initialize fields from parameters. - Set some value depend on ***direction***.

4.12.3. Method

+ void tick()	Update method for StraightEntity.

+ void draw(GraphicsContext gc)	Draw method for StraightEntity.
# Shape getBound()	Return override shape of SlowOrbitEntity.
+ void setDamage(double damage)	Setter of ***damage***.

4.13. Class Field

4.13.1. Field

- boolean destroyed	True if destroy.
- int FieldNumber	Field number or stage.
+ final int SIZE	Size of 1 tile equal 50.
- int[][][] FIELD	Map of game.
- double deg	Degree value use for rotate image.

4.13.2. Constructor

+ Field()	- set ***deg*** value equal 0. - set ***destroy*** to false. - set ***FIELD_NUMBER*** value equal 0.

4.13.3. Method

+ void draw(GraphicsContext gc)	Draw method for Field.
+ void tick()	Do nothing.
+ int getZ()	Return -9999
+ boolean isDestroyed()	Getter of value ***destroy.***
+ boolean isVisible()	Getter of value ***visible.***
+ int get FieldNumber ()	Getter of value ***FieldNumber.***
+ void set FieldNumber (int fIELD_NUMBER)	Setter of value ***FieldNumber.***
+ int[][][] getFieldMap()	Getter of value ***FieldMap.***

4.14. Class FieldOption

4.14.1. Field

- GameLogic gameLogic	The FieldOption component.
- Tower[][] tower	Store object Tower on field.
- int fieldOnX	Current field x that mouse on or mouse click.
- int fieldOnY	Current field y that mouse on or mouse click.
- boolean buildMode	True if in buildMode (left click on available tile for build or upgrade).
- String build	Store string of build type.
- String laserTower	Store string of direction of laser tower.
- int upgradX	Current field x in buildMode.
- int upgradY	Current field y in buildMode.
- int circleOptionDimension	Dimension of circle option.

4.14.2. Constructor

+ FieldOption(GameLogic gameLogic)	- Initialize fields from parameters. - Initialize ***tower*** as 2D Array.

4.14.3. Method

+ void tick()	Update method for FieldOption.
+ void draw(GraphicsContext gc)	Draw method for FieldOption.
+ boolean clickCircle(double Cx, double Cy)	Return true if click in circle with center equal ***Cx*** and ***Cy***, dimension equal ***circleOptionDimension***.
+ void showUpgrade(int x, int y)	Show upgrade icon circle with center equal ***x*** and ***y***, dimension equal ***circleOptionDimension***.

+ int getFieldType(double x, double y)	Return field value from field.
+ Tower getTowerClick()	Return object Tower that get click.
+ int getZ()	Return 20.
+ boolean isDestroyed()	Getter of ***destroy***.
+ boolean isVisible()	Getter of ***visible***.
+ void destroyField()	Destroy all object Tower in ***tower***.
+ void drawPrice(String price, GraphicsContext gc,int x, int y)	Draw upgrade price at ***x***, ***y*** with string ***price***.

4.15. Class Spawn

4.15.1. Field

- GameLogic gameLogic	The GameLogic component.
- Field field	The Field component.
- int x	x value where creep spawn.
- int y	y value where creep spawn.
- int[][] TIME	Time duration of each wave.
- double[][] FREQ	Frequency of creep spawn of each wave.
- final int[] MONEY	Start money of each stage.
- int waveTime	Current wave time.
- int count	Count use for drawing.
- double currentTime	Use in tick().
- double delayEnd	Use for delay end part.
- boolean isWaveStart	True if wave already start.
- boolean isStageClear	True if last wave is end.
- boolean draw	Use for check that need to draw spawn or not.
- boolean firstWave	True if it is first wave of each stage.
- Creep currentCreep	Creep of current wave.

4.15.2. Constructor

+ Spawn(GameLogic gameLogic,Field field)	- Initialize fields from parameters. - Set some value.

4.15.3. Method

+ void tick()	Update method for Spawn.
+ void draw(GraphicsContext gc)	Draw method for Spawn.
- Pair<Integer,Integer> findStart()	Return pair of position x, y where creep is spawn.
+ Pair<Integer,Integer> findEnd()	Return pair of position x, y where is destination of creep.
+ List<Integer> getPath(int fieldNumber,int x,int y,String from)	Return path for creep.
- int findPath(int x, int y)	Get next path to go use in getPath method.
+ Creep getCurrentCreep()	Getter of CurrentCreep.

4.16. Class Myexception

4.16.1. Method

+ MyException(STATE state)	If state equal GameOver, stop play back ground music.
+ String message()	Return “game over”

4.17. Class Myexception2

4.17.1. Field

- int wave	Value of wave.

4.17.2. Method

+ MyException2(int wave)	Set wave equal to parameter.
+ String message()	Return “maximum wave is” +(wave-1)

5. Package application

5.1. Interface IRenderable

5.1.1. Field

5.1.2. Method

+ void tick()	Update method.
+ void draw(GraphicsContext gc)	Draw method.
+ int getZ()	Getter of ***z***.
+ boolean isDestroyed()	Getter of ***destroy***.
+ boolean isVisible()	Getter of ***visible***.

5.2. Class RenderableHolder

5.2.1. Field

+ void tick()	Update method.
+ void draw(GraphicsContext gc)	Draw method.
+ int getZ()	Getter of ***z***.
+ boolean isDestroyed()	Getter of ***destroy***.
+ boolean isVisible()	Getter of ***visible***.

5.2.2. Method

+ void loadResource()	Getter of ***z***.
+ RenderableHolder()	Getter of ***destroy***.
+ void add(IRenderable entity)	Getter of ***visible***.

+ void update()	Remove object from ***entities*** if destroy is true.
+ RenderableHolder getInstance()	Return instance (singleton).
+ List<IRenderable> getEntities()	Getter of ***entities.***
+ void resetGame()	Set ***entities*** to new Arraylist.