Saral Jalan

1969689257

Tejada

**Assignment 5 Design Doc.**

*Purpose/Overview/Rules*

* To create a game using the QT library.
* Get familiar with breakout code and manipulate it to make your own game.

**Rules of the Game**

*Rules of the Game:*

* The purpose of this game is to take the cursor and destroy all the bricks on the screen. The cursor is the bigger white block. The bricks are the smaller white rectangular objects that don’t move.
* You move with the arrow keys. Up, Down, Left and Right.
* **Press SpaceBar to start the game.**
* The objective is to move the cursor and make contact with a brick while staying away from the moving balls and obstacles and shooting objects.
* The evil smiley faces are obstacles.
* If you hit one of the balls you will lose a life.
* You have 10 lives as displayed on the top of the screen.
* If you destroy all 30 blocks by making contact with the cursor and the brick then you can move to the next level.
* The next level is much more tough. There are more obstacles. There are three ships that try to shoot you each on the edge of the screen. Stay away from the bullets or lives will be lost. Each bullet goes faster and each one does different damage.
* The third level has spears that constantly move. They are pink and move in small rows. These are obstacles that also must be avoided to gain lives.
* There is also a random wolf that hops around the screen. If you hit that you will lose a life.
* Good luck!
* Commands in the game:
  + Space-reset,start
  + Arrow Keys to move cursor
  + h-go to harder level/Second Level go harder(if want to play harder level again press h)
  + f- final level/Third world(if want to play final level again press f)
  + e-adds in three extra lives. Can only use this three times and lives do not get carried over from levels.
  + esc-quit
  + p-pause

*Classes*

* 13 classes-catcher.cpp, obstacles.cpp, main.cpp, brick.cpp, game.cpp, superShooter.cpp, tank.cpp, fire.cpp, bullet.cpp, bomb.cpp, bonus.cpp, downObstacles.cpp, sideObstacles.cpp
* 12 Objects
* The bigger rectangular cursor is from catcher.cpp.
* The 15 bricks are all of type brick.cpp.
* The obstacle.cpp object is used to create the balls that fly around the screen.
* game.cpp is really where all the game play coding happens. The movements and the key button movements etc.
* downObstacles inherits from sideObstacles. Polymorphism is used to between these two classes.
* The breakout code was used as a template to write all classes.
* PS: When I was putting in images for ships and stuff it was becoming really slow, so I just made solid block colors.

*Global Data/Functions*

* Nothing is global.

*User Interface*

* Used the QT GUI to create the game.
* Use the arrow keys to move the catcher object.

***Document Functions of game.cpp***

1. *void paintEvent(QPaintEvent \*event);*

* No return type.
* Everything gets printed in this function.
* The QPaintEvent manages everything that is printed onto the window.
* Bricks, Obstacles, Catcher,etc.
* The results of the game are also displayed.
* Also has function repaint() which basically refreshed whatever is on the screen. Everything gets painted again.

1. *void timerEvent(QTimerEvent \*event);*

* *This void functions always is running basically like an infinite loop.*
* *The movement of all the objects happens in this function.*
* *In order to speed up objects, I have printed some of the obstacles two times in this function so for every iteration the ball will move two times.*

1. *void keyPressEvent(QKeyEvent \*event);*

* *Everything the gamer times will be handled by this function.*
* *You can quit,pause,restart etc, based on this function.*
* *Also you control the movement of the cursor through this function.*

1. *void startGame()*

* *restarts the game*

1. *void pauseGame();*

* *pauses the game*

1. *void stopGame();*

* *stops the game*

1. *void victory();*

* *Displays the victory message.*
* *Occurs when you beat a level.*

1. *void checkCollision();*

* *Checks for all the collisions in the game.*
* *Examples-Catcher with the obstacles. Catcher with the bricks.*
* *Calculates the number of blocks and lives left.*

1. *Game(QWidget \*parent = 0);*

* *Constructor.*
* *Defines the pointers created in the private section.*
* *Sets position of all the objects.*

1. *~Game();*

* *Destructor*
* *Deletes all the objects when game is quit to avoid memory leak.*

***Document Functions of class catcher.cpp***

1. *void resetState()*

* *moves object to the middle location*
* *used at the beginning of the game*

1. *void moveLeft(int);*

* *Moves the object to the left.*

1. *void moveRight(int);*

* *Moves the object to the right.*

1. *void moveTop(int);*

* *Moves the object one space up.*

1. *void moveBottom(int)*

* *Moves the object one space down.*

1. *int getWidth();*

* *Returns the width of the object.*
* *Return type: int*

1. *int getHeight();*

* *Returns the height of the object.*
* *Return type: int*

1. *QRect getRect();*

* *Returns the rectangle of the object.*

1. *QImage & getImage();*

* *Returns the image of the rectangle.*

1. *Catcher();*

* *Constructor.*
* *Sets the image to catcher.*

1. *~Catcher();*

* *Prints out that catcher has been deleted. Destructor.*

***Documented functions of the brick.cpp.***

1. void resetState();

* resets and initializes the positions of the bricks.
* Needed so bricks will not go out of boundary.

1. bool isDestroyed();

* Returns if the brick is destroyed or not.
* If destroyed then true.
* If not destroyed then false.

1. void setDestroyed(bool);

* Initializes all the bricks to not destroyed.

1. QRect getRect();

* Returns the rectangle form of the bricks.

1. void setRect(QRect);

* Sets the rectangle type.
* Needed to set the image as a rectangle.

1. QImage & getImage();

* Returns the image of the rectangle and brick object.

1. int getWidth();

* Returns the width of the brick.

1. int getHeight();

* Returns the height of the brick.

1. void movePositioning(int, int);

* Sets the position of the block to the (X,Y) location in the window.

1. Brick(int, int);

* Creates and sets the image of the brick.
* Takes in the initial position to set the brick on the window.
* Constructor.

1. ~Brick();
   * Destructor.
   * Deletes the brick objects.

***Documented functions of the obstacle.cpp***

1. Obstacle();

* Constructor
* Creates the obstacles and sets the image to them.

1. ~Obstacle();

* Destructor
* Deletes the obstacle objects.

1. void resetState();

* moves the obstacles to a specific location.

1. void moveBottom(int);
2. void moveTop(int);
3. void moveLeft(int);
4. void moveRight(int);
5. void autoMove();

* moves the obstacles automatically

1. QRect getRect();

* Used for image setting.

1. QImage & getImage();

* Gets the image. Png file.

1. void movePositioning(int, int);

* moves the brick to a certain position.

**Documented functions of Bomb.cpp**

QRect getRect();

QImage & getImage();

void movePositioning(int, int);

void autoMoveRight();

void setXDirection(int);

**Documented Functions of Bonus.cpp**

int getWidth();

int getHeight();

QRect getRect();

QImage & getImage();

void autoMove(int, int);

**Documented Functions of Brick.cpp**

void resetState();

bool isDestroyed();

void setDestroyed(bool);

QRect getRect();

void setRect(QRect);

QImage & getImage();

int getWidth();

int getHeight();

void movePositioning(int, int);

**Documented Functions of Bullet.cpp**

void getRidofBullets();

QRect getRect();

QImage & getImage();

void movePositioning(int, int);

void setBulletStatus(bool);

bool getBulletStatus();

void autoMoveRight();

void setXDirection(int);

**Documented Functions of downObstacles.cpp**

downObstacles();

~downObstacles();

void autoMoveDown();

(Inherits everything from sideObstacles) Polymorphism also used.

**Document Functions from Fire.cpp**

QRect getRect();

QImage & getImage();

void movePositioning(int, int);

void autoMoveUp();

void setYDirection(int);

**Documented Functions from Rocket.cpp**

Rocket();

~Rocket();

int getWidth();

int getHeight();

QRect getRect();

QImage & getImage();

void movePositioning(int, int);

void autoMove();

int getXCoordinate() const;

int getYCoordinate() const;

void setXDirection(int);

**Documented Functions from sideObstacles.cpp**

sideObstacles();

~sideObstacles();

void moveLeft(int);

void moveRight(int);

void moveTop(int);

void moveBottom(int);

int getWidth();

int getHeight();

QRect getRect();

QImage & getImage();

void movePositioning(int, int);

void autoMoveRight();

void autoMoveDown();

void randomMove();

void randomMovement();

**Documented Functions from superShooter.cpp**

superShooter();

~superShooter();

void autoMovement();

void setYDirection(int);

QRect getRect();

QImage & getImage();

void movePositioning(int, int);

**Documented Functions from Tank.cpp**

int getWidth();

int getHeight();

QRect getRect();

QImage & getImage();

void movePositioning(int, int);

void autoMove();

int getXCoordinate() const;

int getYCoordinate() const;

void setYDirection(int);

***Main.cpp Documentation***

1. **void center(QWidget widget)**

* **sets up the screen.**
* **Sets up dimensions of window.**

1. **Main**

* **Creates instance of game.**
* **Puts it onto the window.**
* **Game play starts.**