Fancy Game

Brought to you by ~~General~~ tovarischka Chen, General Lerdorf, and Comrade Wong

P8 Java TEAM 1: "The One and Only Spooky Fancy 100% Balanced,100% Legit, Not a Scam, Free Trade, Quality Certified Guarantee Meme Team led by Three "*Smart 🅱ois*""

Lol we are team number one

About Our Project...

Solutions to world problems:

- Fancy Game
  - Solves boredom
  - Inspiration for game design
  - Teaches graphic design by showing how it's NOT done

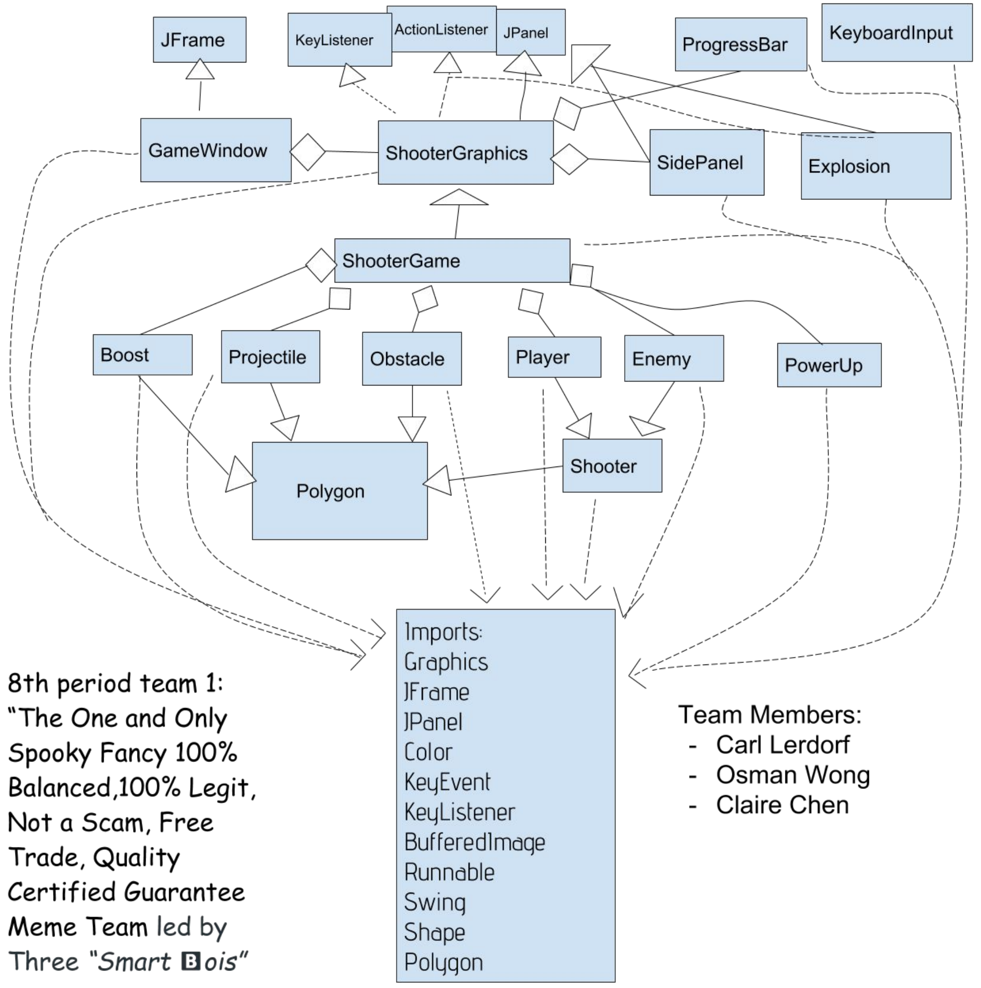
About the game:

- Objectives
  - Destroy enemies
  - Survive
  - Get points for doing so!

SEIZURE WARNING

COMIC SANS WARNING

SEIZURE WARNING



8th period team 1:
"The One and Only Spooky Fancy 100% Balanced,100% Legit, Not a Scam, Free Trade, Quality Certified Guarantee Meme Team led by Three "*Smart 🅱ois*"

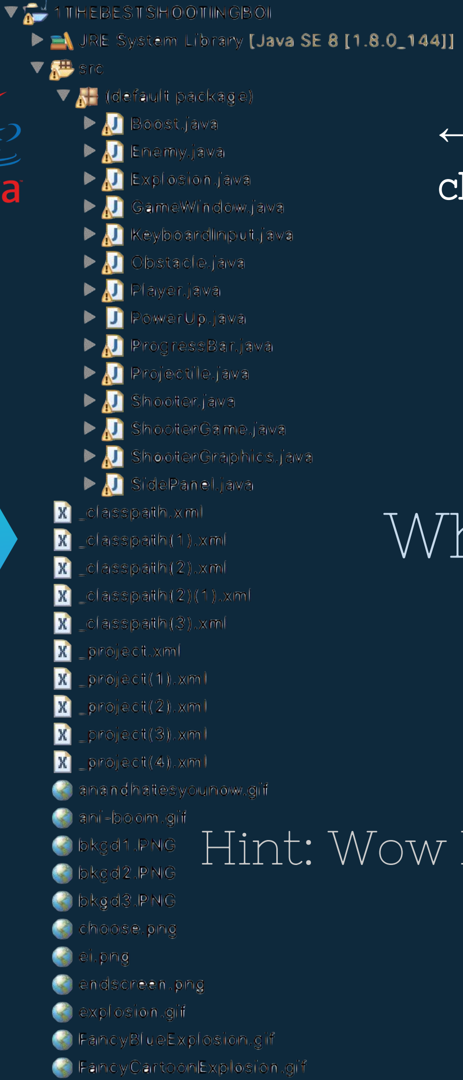
Team Members:
- Carl Lerdorf
- Osman Wong
- Claire Chen

Key Responsibilities

Osman-game mechanics

Carl-explosions and listening for key events

Claire-graphics, UI and the menu

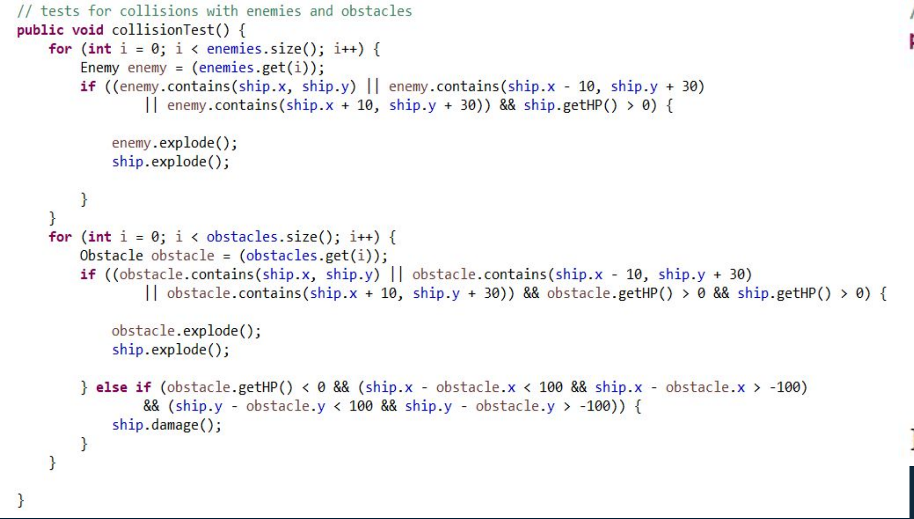


```
▼ 1THEBESTSHOOTINGBOI
  ▶ JRE System Library [Java SE 8 [1.8.0_144]]
  ▼ src
    ▼ (default package)
      ▶ Boost.java
      ▶ Enemy.java
      ▶ Explosion.java
      ▶ GameWindow.java
      ▶ KeyboardInput.java
      ▶ Obstacle.java
      ▶ Player.java
      ▶ PowerUp.java
      ▶ ProgressBar.java
      ▶ Projectile.java
      ▶ Shooter.java
      ▶ ShooterGame.java
      ▶ ShooterGraphics.java
      ▶ SidePanel.java
  _classpath.xml
  _classpath(1).xml
  _classpath(2).xml
  _classpath(2)(1).xml
  _classpath(3).xml
  _project.xml
  _project(1).xml
  _project(2).xml
  _project(3).xml
  _project(4).xml
  anandhatesyounow.gif
  ani-boom.gif
  bkgd1.PNG
  bkgd2.PNG
  bkgd3.PNG
  choose.png
  ei.png
  endscreen.png
  explosion.gif
  FancyBlueExplosion.gif
  FancyCartoonExplosion.gif
```

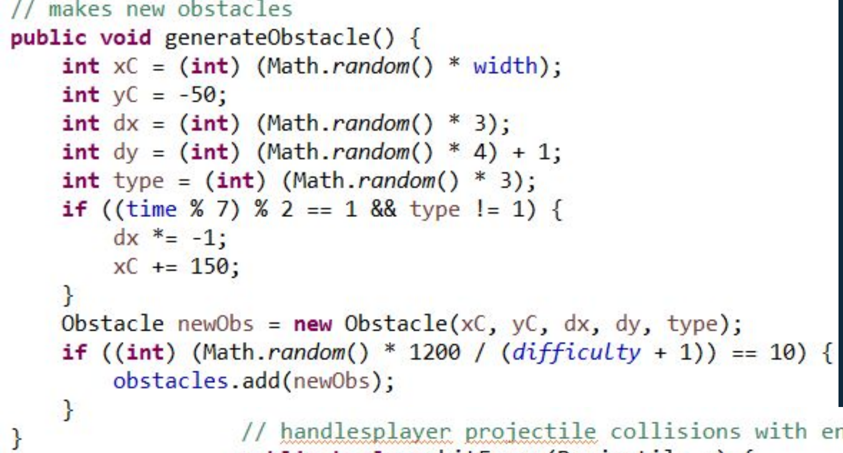
← it's not like we have that many classes anyways....(͡° ͜ʖ ͡°)

What do all these classes even do?

Hint: Wow look at this nice naming, no cancer guaranteed



```java
// tests for collisions with enemies and obstacles
public void collisionTest() {
    for (int i = 0; i < enemies.size(); i++) {
        Enemy enemy = (enemies.get(i));
        if ((enemy.contains(ship.x, ship.y) || enemy.contains(ship.x - 10, ship.y + 30)
                || enemy.contains(ship.x + 10, ship.y + 30)) && ship.getHP() > 0) {

            enemy.explode();
            ship.explode();

        }
    }
    for (int i = 0; i < obstacles.size(); i++) {
        Obstacle obstacle = (obstacles.get(i));
        if ((obstacle.contains(ship.x, ship.y) || obstacle.contains(ship.x - 10, ship.y + 30)
                || obstacle.contains(ship.x + 10, ship.y + 30)) && obstacle.getHP() > 0 && ship.getHP() > 0) {

            obstacle.explode();
            ship.explode();

        } else if (obstacle.getHP() < 0 && (ship.x - obstacle.x < 100 && ship.x - obstacle.x > -100)
                && (ship.y - obstacle.y < 100 && ship.y - obstacle.y > -100)) {
            ship.damage();
        }
    }

}
```

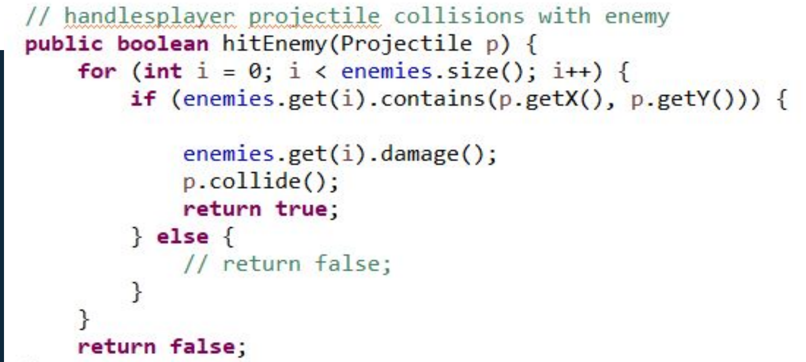


```java
// makes new obstacles
public void generateObstacle() {
    int xC = (int) (Math.random() * width);
    int yC = -50;
    int dx = (int) (Math.random() * 3);
    int dy = (int) (Math.random() * 4) + 1;
    int type = (int) (Math.random() * 3);
    if ((time % 7) % 2 == 1 && type != 1) {
        dx *= -1;
        xC += 150;
    }
    Obstacle newObs = new Obstacle(xC, yC, dx, dy, type);
    if ((int) (Math.random() * 1200 / (difficulty + 1)) == 10) {
        obstacles.add(newObs);
    }
}
```

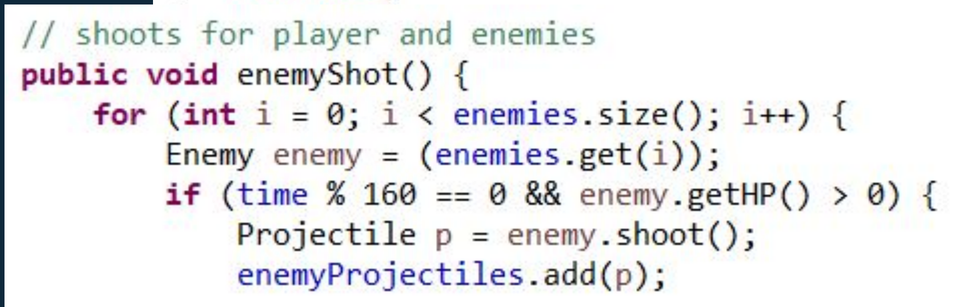
ShooterGame.java

What is it: Contains game logic and mechanics

- Handles events such as:
    - Collisions
    - Firing
    - enemy/obstacle movement
    - MUCH, MUCH MORE



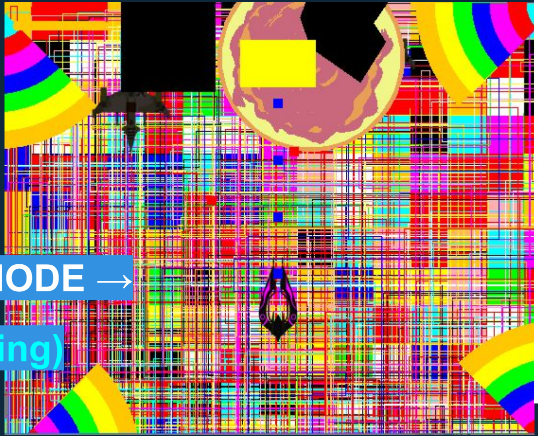
```java
// handlesplayer projectile collisions with enemy
public boolean hitEnemy(Projectile p) {
    for (int i = 0; i < enemies.size(); i++) {
        if (enemies.get(i).contains(p.getX(), p.getY())) {

            enemies.get(i).damage();
            p.collide();
            return true;
        } else {
            // return false;
        }
    }
    return false;
```

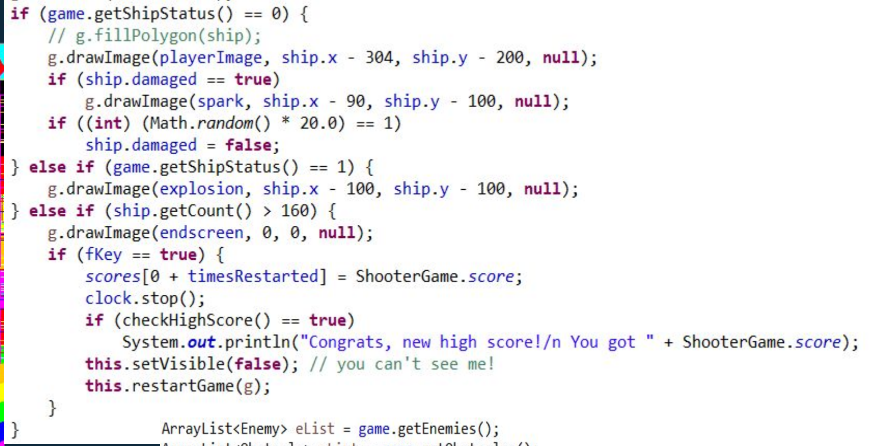


```java
// shoots for player and enemies
public void enemyShot() {
    for (int i = 0; i < enemies.size(); i++) {
        Enemy enemy = (enemies.get(i));
        if (time % 160 == 0 && enemy.getHP() > 0) {
            Projectile p = enemy.shoot();
            enemyProjectiles.add(p);
```

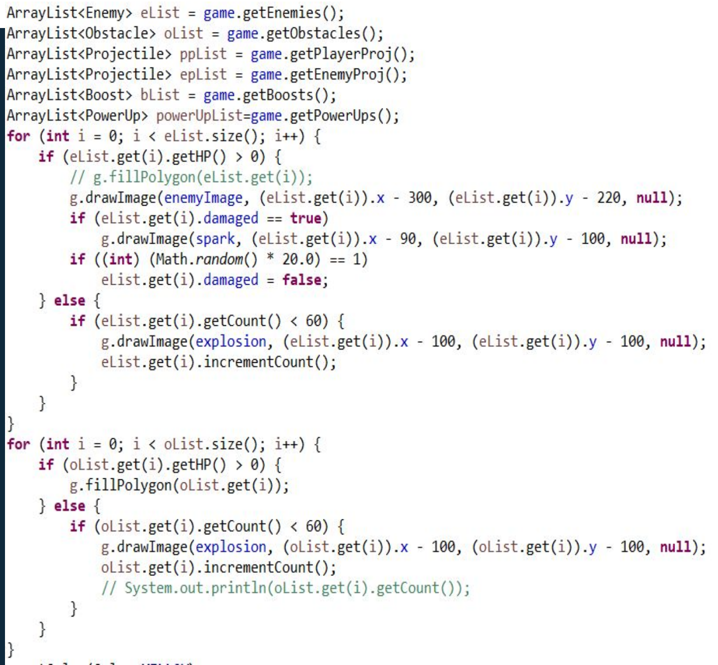
GOD MODE →

(seizure warning)





```java
if (game.getShipStatus() == 0) {
    // g.fillPolygon(ship);
    g.drawImage(playerImage, ship.x - 304, ship.y - 200, null);
    if (ship.damaged == true)
        g.drawImage(spark, ship.x - 90, ship.y - 100, null);
    if ((int) (Math.random() * 20.0) == 1)
        ship.damaged = false;
} else if (game.getShipStatus() == 1) {
    g.drawImage(explosion, ship.x - 100, ship.y - 100, null);
} else if (ship.getCount() > 160) {
    g.drawImage(endscreen, 0, 0, null);
    if (fKey == true) {
        scores[0 + timesRestarted] = ShooterGame.score;
        clock.stop();
        if (checkHighScore() == true)
            System.out.println("Congrats, new high score!/n You got " + ShooterGame.score);
        this.setVisible(false); // you can't see me!
        this.restartGame(g);
    }
}
```



```java
ArrayList<Enemy> eList = game.getEnemies();
ArrayList<Obstacle> oList = game.getObstacles();
ArrayList<Projectile> ppList = game.getPlayerProj();
ArrayList<Projectile> epList = game.getEnemyProj();
ArrayList<Boost> bList = game.getBoosts();
ArrayList<PowerUp> powerUpList=game.getPowerUps();
for (int i = 0; i < eList.size(); i++) {
    if (eList.get(i).getHP() > 0) {
        // g.fillPolygon(eList.get(i));
        g.drawImage(enemyImage, (eList.get(i)).x - 300, (eList.get(i)).y - 220, null);
        if (eList.get(i).damaged == true)
            g.drawImage(spark, (eList.get(i)).x - 90, (eList.get(i)).y - 100, null);
        if ((int) (Math.random() * 20.0) == 1)
            eList.get(i).damaged = false;
    } else {
        if (eList.get(i).getCount() < 60) {
            g.drawImage(explosion, (eList.get(i)).x - 100, (eList.get(i)).y - 100, null);
            eList.get(i).incrementCount();
        }
    }
}
for (int i = 0; i < oList.size(); i++) {
    if (oList.get(i).getHP() > 0) {
        g.fillPolygon(oList.get(i));
    } else {
        if (oList.get(i).getCount() < 60) {
            g.drawImage(explosion, (oList.get(i)).x - 100, (oList.get(i)).y - 100, null);
            oList.get(i).incrementCount();
            // System.out.println(oList.get(i).getCount());
        }
    }
}
```

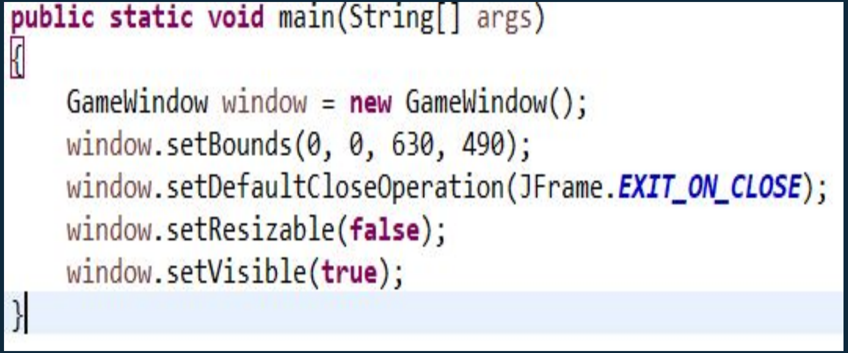
ShooterGraphics.java

What is it: Contains most graphics

-Draws basically everything, also helps with listening for key events and has the timer

-Handles lots of user interactions with the graphical interface, and allows us to set difficulty, restart game, etc…usability (or lack thereof)

-Anything that involves drawing and displaying goes here



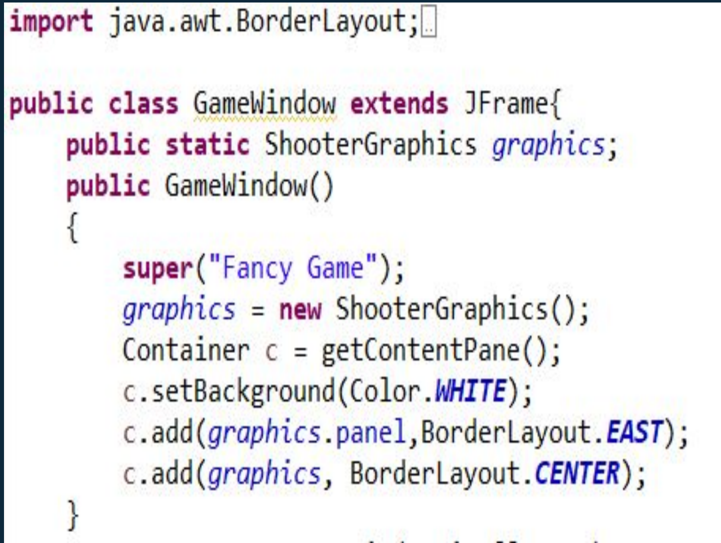
```
public static void main(String[] args)
{
    GameWindow window = new GameWindow();
    window.setBounds(0, 0, 630, 490);
    window.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    window.setResizable(false);
    window.setVisible(true);
}
```

GameWindow.java

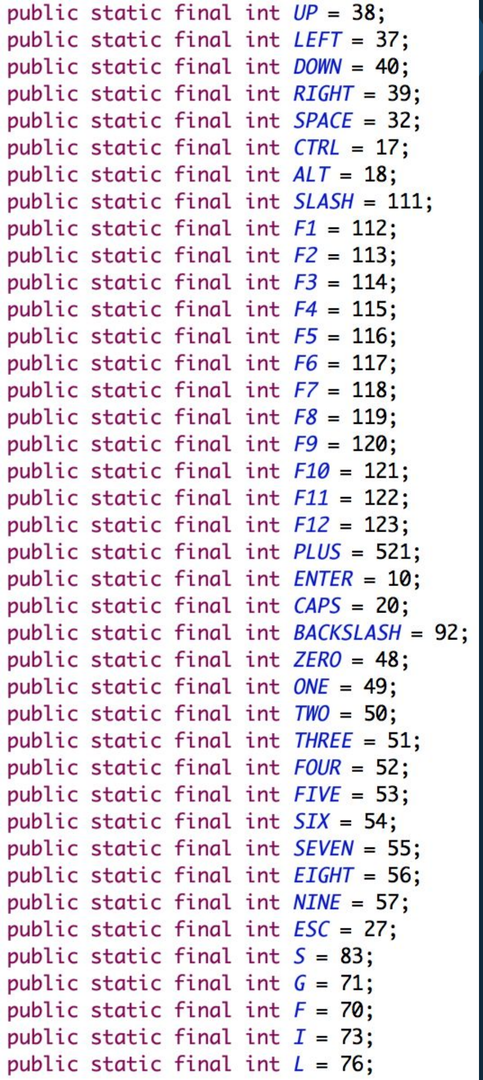
Contains: Main Method

-Creates JFrame, adds ShooterGraphics and its SidePanel

-This is what you would run to open the game



```
import java.awt.BorderLayout;

public class GameWindow extends JFrame{
    public static ShooterGraphics graphics;
    public GameWindow()
    {
        super("Fancy Game");
        graphics = new ShooterGraphics();
        Container c = getContentPane();
        c.setBackground(Color.WHITE);
        c.add(graphics.panel,BorderLayout.EAST);
        c.add(graphics, BorderLayout.CENTER);
    }
```



```java
public static final int UP = 38;
public static final int LEFT = 37;
public static final int DOWN = 40;
public static final int RIGHT = 39;
public static final int SPACE = 32;
public static final int CTRL = 17;
public static final int ALT = 18;
public static final int SLASH = 111;
public static final int F1 = 112;
public static final int F2 = 113;
public static final int F3 = 114;
public static final int F4 = 115;
public static final int F5 = 116;
public static final int F6 = 117;
public static final int F7 = 118;
public static final int F8 = 119;
public static final int F9 = 120;
public static final int F10 = 121;
public static final int F11 = 122;
public static final int F12 = 123;
public static final int PLUS = 521;
public static final int ENTER = 10;
public static final int CAPS = 20;
public static final int BACKSLASH = 92;
public static final int ZERO = 48;
public static final int ONE = 49;
public static final int TWO = 50;
public static final int THREE = 51;
public static final int FOUR = 52;
public static final int FIVE = 53;
public static final int SIX = 54;
public static final int SEVEN = 55;
public static final int EIGHT = 56;
public static final int NINE = 57;
public static final int ESC = 27;
public static final int S = 83;
public static final int G = 71;
public static final int F = 70;
public static final int I = 73;
public static final int L = 76;
```

KeyboardInput.java

Contains: utilities helpful for interpreting and listening for key events

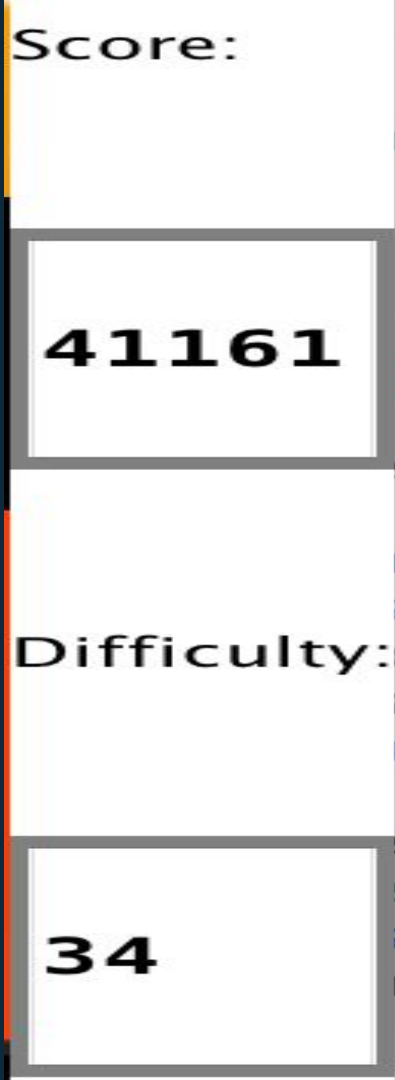
← +100 organization

SidePanel.java

Is the panel on the right side

-Displays score and difficulty

-Updates constantly


Score:

41161

Difficulty:

34

ProgressBar.java

Is a progress bar(duh?)

-In this case, used to show player HP

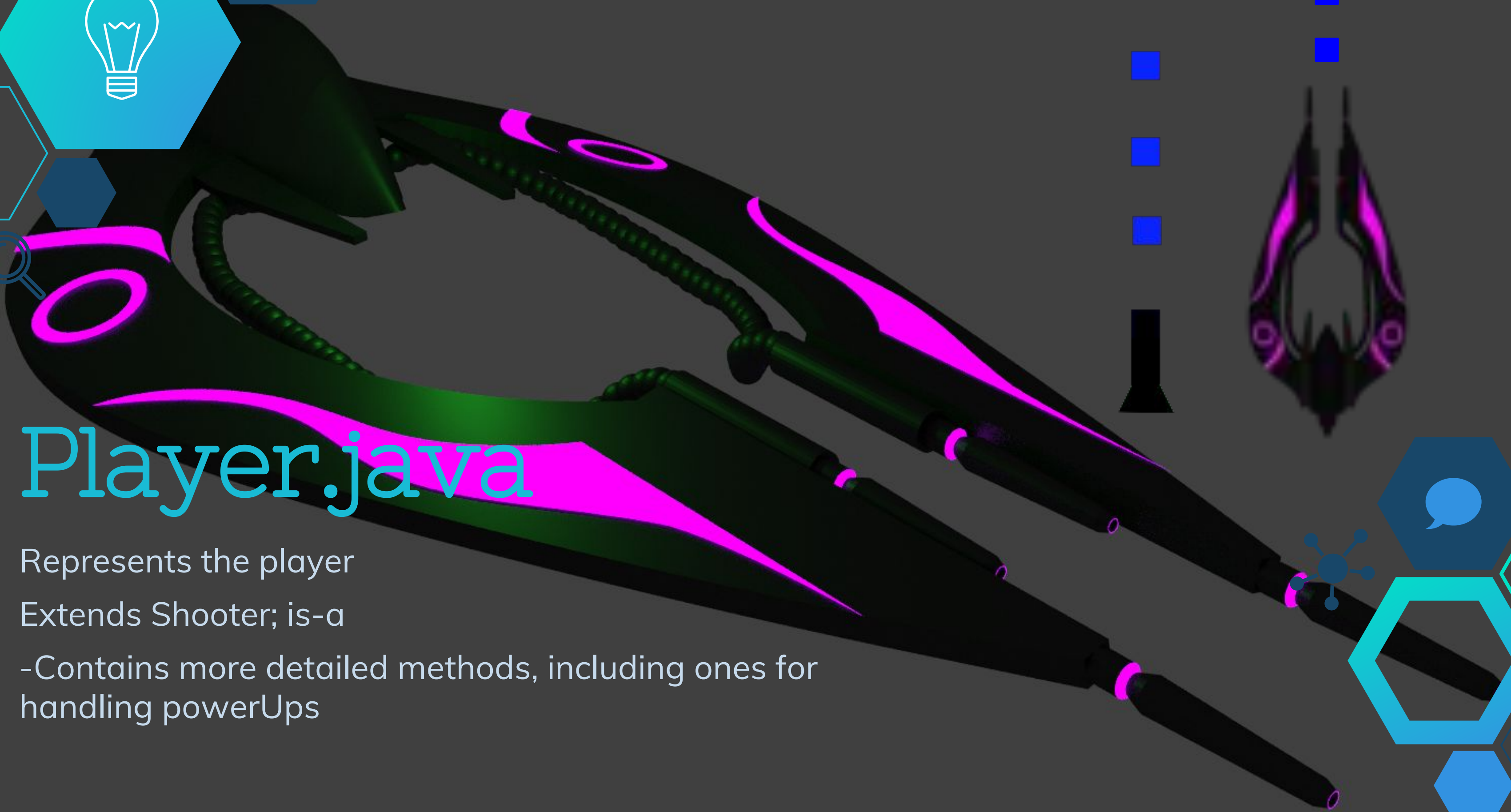
-Can lengthen due to boosts

Shooter.java

Represents a “shooter” that can move and shoot projectiles

Has-a relationship with projectiles

-contains some basic movement and action methods

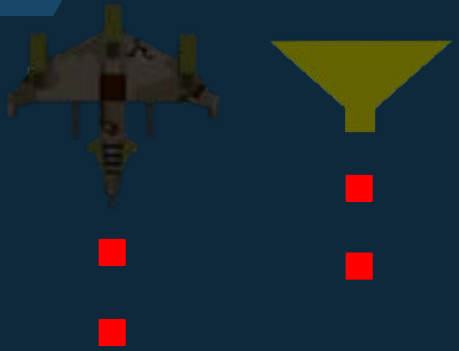


Player.java

Represents the player

Extends Shooter; is-a

-Contains more detailed methods, including ones for handling powerUps



Enemy.java

Represents an enemy

Extends Shooter

-Contains methods for enemy movement and behavior

Projectile.java

Represents a projectile(or “bullet”) that is shot by the player and enemies and deals damage upon contact

Obstacle.java

Represents an obstacle that kills the player upon collision

-Contains methods for movement, damage, etc.

-3 different types of obstacles: asteroid, nuke, block. They only differ in appearance; gameplay-wise they all behave the same

PowerUp.java

Represents a power up that may fall after killing an enemy or blowing up a projectile, and gives a special effect upon collection by player

-Types: Double and triple shot, rapid shot, point boosts, shield, HP recovery up to max, faster movement

Boost.java

Represents “boosts", which give the player extra health points when shot down, even giving extra health beyond the usual maximum of 10

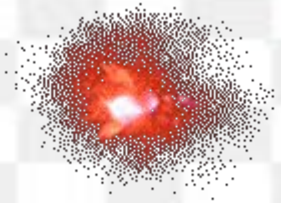
They are sketchy yellow rectangles which blink in and out of existence, so good luck hitting them

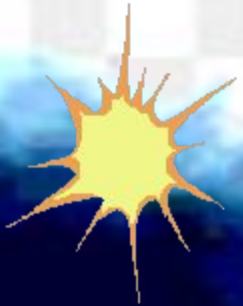
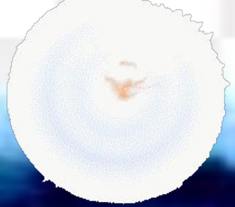
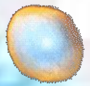
Used to display a variety of explosions when an enemy/obstacle/you is blown up

Random new explosion gifs!

Carl worked very hard to format and use these explosion gifs. Respect +100

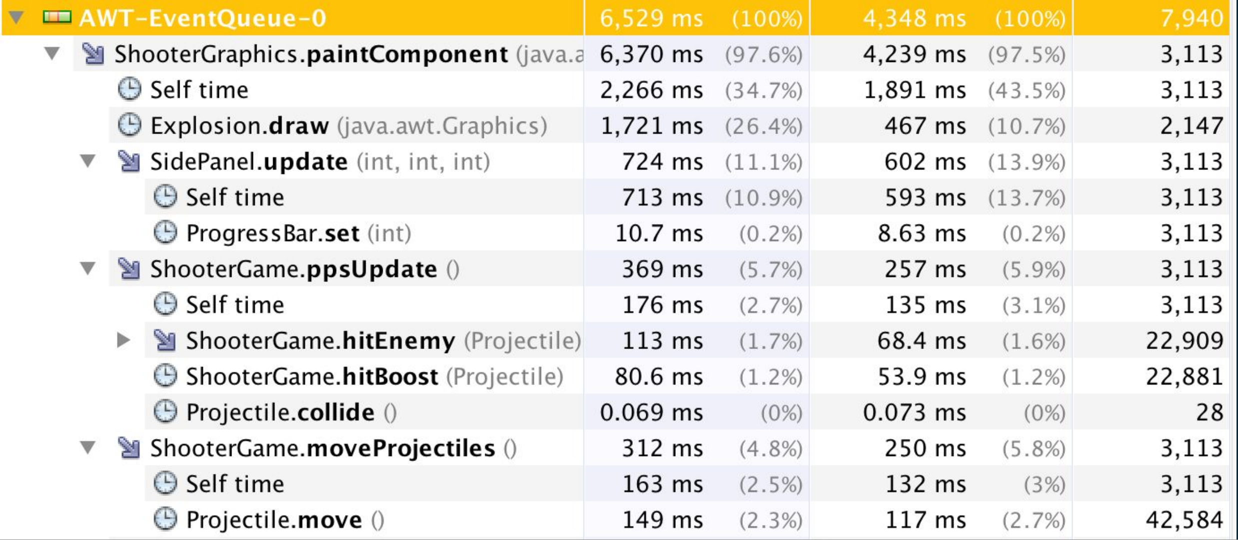
Explosion.java



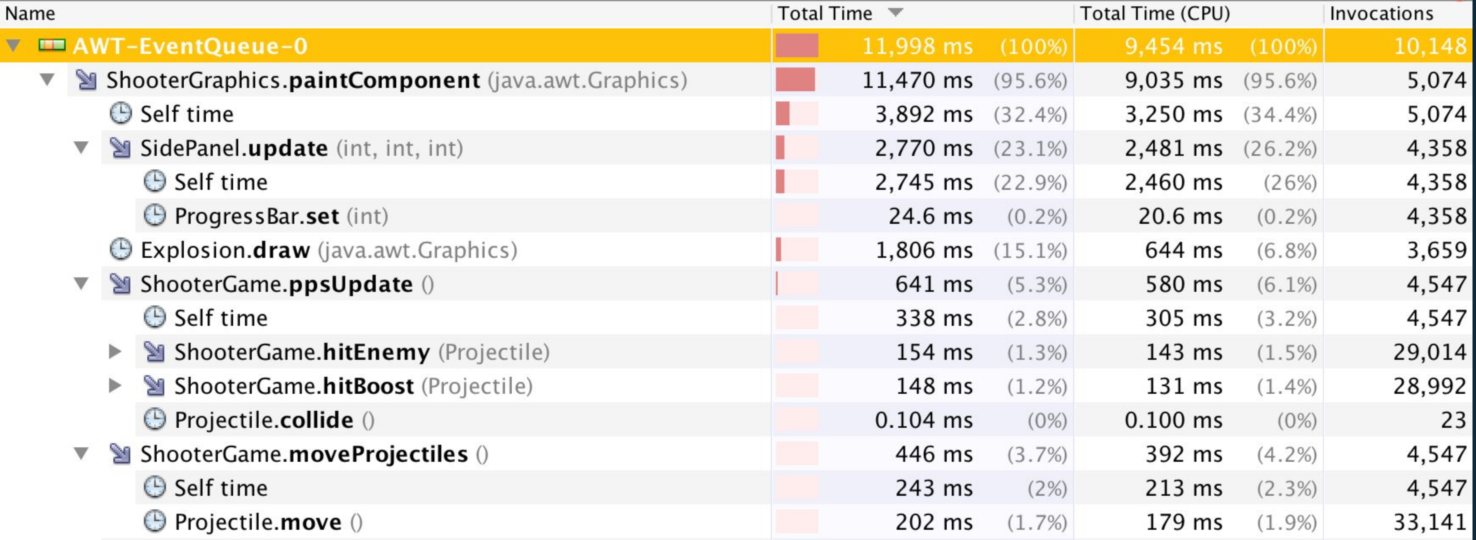




Lag Problem

Visual VM



AWT-EventQueue-0	6,529 ms	(100%)	4,348 ms	(100%)	7,940
ShooterGraphics.**paintComponent** (java.a	6,370 ms	(97.6%)	4,239 ms	(97.5%)	3,113
Self time	2,266 ms	(34.7%)	1,891 ms	(43.5%)	3,113
Explosion.**draw** (java.awt.Graphics)	1,721 ms	(26.4%)	467 ms	(10.7%)	2,147
SidePanel.**update** (int, int, int)	724 ms	(11.1%)	602 ms	(13.9%)	3,113
Self time	713 ms	(10.9%)	593 ms	(13.7%)	3,113
ProgressBar.**set** (int)	10.7 ms	(0.2%)	8.63 ms	(0.2%)	3,113
ShooterGame.**ppsUpdate** ()	369 ms	(5.7%)	257 ms	(5.9%)	3,113
Self time	176 ms	(2.7%)	135 ms	(3.1%)	3,113
ShooterGame.**hitEnemy** (Projectile)	113 ms	(1.7%)	68.4 ms	(1.6%)	22,909
ShooterGame.**hitBoost** (Projectile)	80.6 ms	(1.2%)	53.9 ms	(1.2%)	22,881
Projectile.**collide** ()	0.069 ms	(0%)	0.073 ms	(0%)	28
ShooterGame.**moveProjectiles** ()	312 ms	(4.8%)	250 ms	(5.8%)	3,113
Self time	163 ms	(2.5%)	132 ms	(3%)	3,113
Projectile.**move** ()	149 ms	(2.3%)	117 ms	(2.7%)	42,584



Name	Total Time		Total Time (CPU)		Invocations
AWT-EventQueue-0	11,998 ms	(100%)	9,454 ms	(100%)	10,148
ShooterGraphics.**paintComponent** (java.awt.Graphics)	11,470 ms	(95.6%)	9,035 ms	(95.6%)	5,074
Self time	3,892 ms	(32.4%)	3,250 ms	(34.4%)	5,074
SidePanel.**update** (int, int, int)	2,770 ms	(23.1%)	2,481 ms	(26.2%)	4,358
Self time	2,745 ms	(22.9%)	2,460 ms	(26%)	4,358
ProgressBar.**set** (int)	24.6 ms	(0.2%)	20.6 ms	(0.2%)	4,358
Explosion.**draw** (java.awt.Graphics)	1,806 ms	(15.1%)	644 ms	(6.8%)	3,659
ShooterGame.**ppsUpdate** ()	641 ms	(5.3%)	580 ms	(6.1%)	4,547
Self time	338 ms	(2.8%)	305 ms	(3.2%)	4,547
ShooterGame.**hitEnemy** (Projectile)	154 ms	(1.3%)	143 ms	(1.5%)	29,014
ShooterGame.**hitBoost** (Projectile)	148 ms	(1.2%)	131 ms	(1.4%)	28,992
Projectile.**collide** ()	0.104 ms	(0%)	0.100 ms	(0%)	23
ShooterGame.**moveProjectiles** ()	446 ms	(3.7%)	392 ms	(4.2%)	4,547
Self time	243 ms	(2%)	213 ms	(2.3%)	4,547
Projectile.**move** ()	202 ms	(1.7%)	179 ms	(1.9%)	33,141

Game rule overview

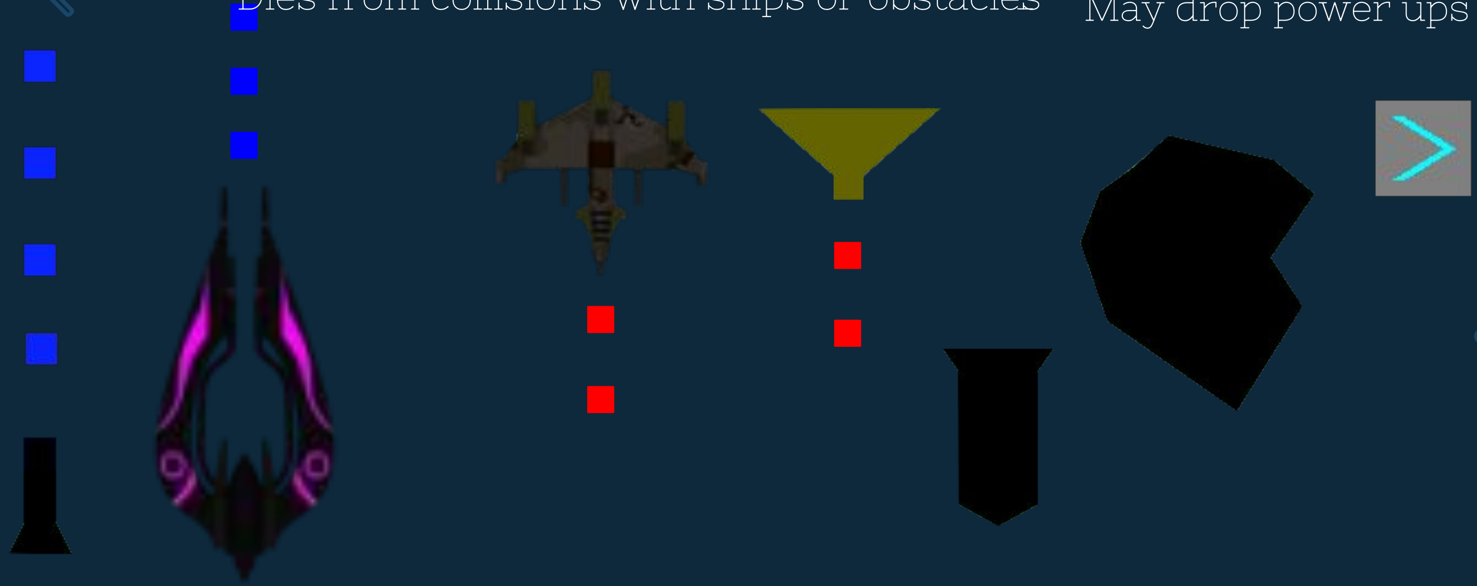
Player

- Controlled by arrow keys
- Takes damage from projectiles
- Dies from collisions with ships or obstacles

Enemies

- Move left and right
- Gets damaged by projectiles

May drop power ups on death



Score:

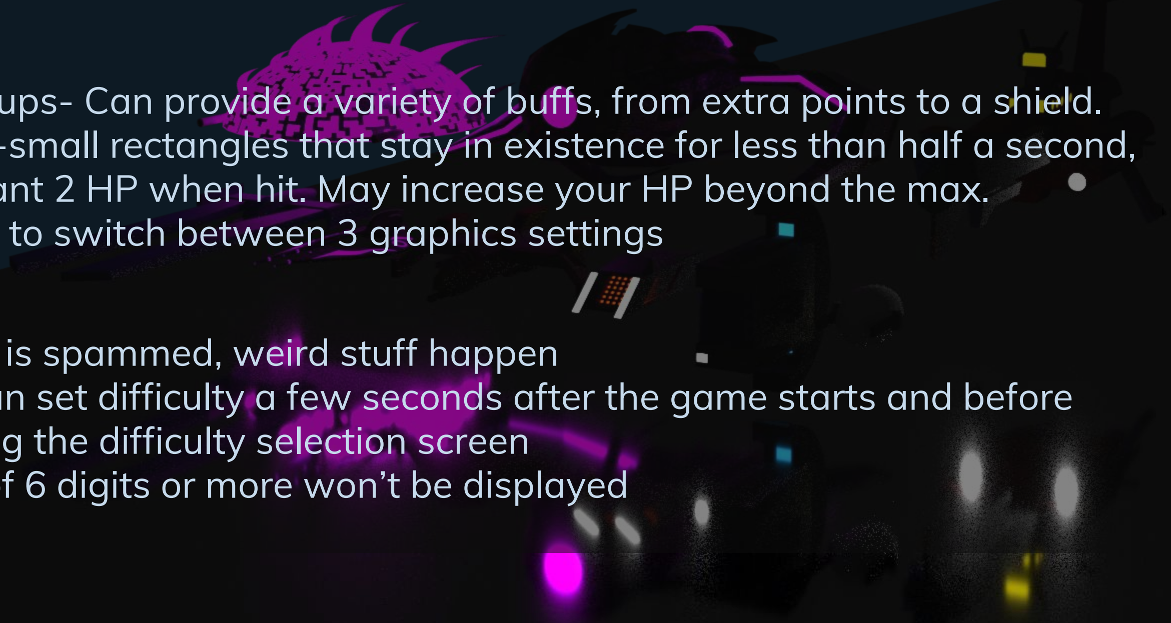
41161

Difficulty:

34

Current Features/Bugs

Current Features/Bugs



Features:

- Power ups- Can provide a variety of buffs, from extra points to a shield.
- Boosts-small rectangles that stay in existence for less than half a second, and grant 2 HP when hit. May increase your HP beyond the max.
- Press L to switch between 3 graphics settings

Bugs:

- If L key is spammed, weird stuff happen
- User can set difficulty a few seconds after the game starts and before reaching the difficulty selection screen
- Score of 6 digits or more won't be displayed

Next Steps

- More types of enemies
- More power ups
- Different player ships
- Multiplayer
- Storing high scores in text file locally or in database
- Using those stored scores to provide benefits/upgrades to player

Image sources: codepen.io, Google

Art made using Krita and MS Paint (when Krita broke)

Ships made using blender and fusion

Inspiration from Bullethead, Space Invaders, Asteroids

Thanks Mr. Taylor for random suggestions, tolerating our spam, and providing gifs

Credits & Acknowledgements