## Header:

-Written by Eric Decasper, Sam Lestiko, Kevin Rossel
-Written from 4/29-5/24

## Introduction:

- Our program is a pvp, free for all, battle royale style game. Our program allows player to play against each other in a pvp game.
- People who want to play this game are people who want to have fun, playing a casual pvp game with friends
- One primary features is the main menu where username can be inputted and network key can also be inputted. The game can also be started from this menu. Another primary feature is the actual game interface where characters play against each other in a cool game of laser tag.

## Instructions:

When you launch the game you are first prompted by an "epic" loading screen. Next, the main menu will appear where you can press start, type an ip to a server, then join. Once in the game, you might want to know the controls, which are listed below.

**Controls:**

W: Move up
A: Move left
S: Move down
D: Move right
Left-Click: Fire a laser
Right-Click: Place a wall
ESC: Leave game

## Class list:

**Launcher** - This is the entry point of the program. It creates an instance of the Game and then runs it.

Methods:
main(String[] args)

**Window** - Is an instance of the window. Creates a JFrame, and adds a Canvas to it. (also adds a broken key listener to the JFrame)

Fields:
JFrame frame
Canvas canvas
String title
int width
int height
boolean[] keyPressed
boolean[] keyTyped
boolean[] virtualKeyPressed
boolean[] virtualKeyTyped

Constructor: Window(String, int, int)

Methods:
void init()
Canvas getCanvas()
boolean isKeyPressed()

**Game** - Instance of the game. It is the “game loop” where it looks something like this:

```
while(game is running) {
     if(in game & no overlay menu open)
     Go to every entity to call the update and draw function
     Else
     Draw current menu
}
```

Fields:
int width
int height
String title
Thread thread
boolean running
Thread thread
boolean inGame
boolean connected
BufferStrategy bs
Graphics2D g

NetworkManager manager

Constructor: Game(String,int,int)

Methods:
void init()
void tick()
void render()
void run()
void start()
void stop()

**Entity** - A super class to be a skeleton of an entity
Has features like:
tick() - move based on key presses or just do any moving it needs to do
draw() - draw the entity (by default just a rectangle)

Fields:
int x
int y
int width
int height
Color color

Constructor: Entity(int x, int y, int width, int height, Color color)

Methods:
void tick()
void draw(Graphics2D g)
int getX()
int getY()
void setX()
void setY()

**Player** - This is the users class. Extends entity, right now all that is different is that it has a field for the player's name.

Method:
tick()

Field:
String name

Constructor:

public Player(int x, int y, int width, int height, Color color, String name)

**MenuManager** - Has a single field called “current” which stores the current screen that is being displayed and then calls the draw function for that menu screen.
Methods:
Public void draw(Graphics2D g){}
Public void setCurrentScreen(Screen screen){}

Fields:
Screen current

Constructor:
Public MenuManager() {}
**Screen** - An interface for a screen. Literally just an abstract function called init() to instantiate buttons, and draw(Graphics2D g) to draw its buttons and stuff.

Methods:
public abstract void init()
public abstract void draw(Graphics2D g)

Fields: None

Constructor: none

**Loading** - This will be the loading screen for the game
Methods:

Public void init(){}
Public void draw(Graphics2D g){}
Fields:
None
Constructors:
None

**MainMenu** - This just has an array list of button objects that it adds to when the init function is called. Then draws the buttons. (right now its only a hello world button)

Void int()
Void draw(Graphics2D g)
Void mousePressed(MouseEvent)
Void mousePressed(MouseEvent)
Void mouseExited(MouseEvent)
Void mouseReleased(MouseEvent)
Void mouseClicked(MouseEvent)

**Button** - This is a class that represents a button. Stores x, y, width, and height. Also the text and color. Each button needs a button task which is…

public void draw(Graphics2D g)
public void clicked()
public int getX()
public int getY()
public int getWidth()
public int getHeight()

Fields:
private int x, y, width, height;
private String text;
private Color color;
private ButtonTask task;

Constructor:
public Button(int x, int y, int width, int height, String text, Color color, ButtonTask task)

**ButtonTask** - An interface that only has and abstract void called run(). To make a button task you just implement this and whatever you put in run() should be run when the button is clicked.

Methods:
Public abstract void run();
Field:
None
Constructors:
None
**NetworkManager** - This class is responsible for managing the connection to the server.

Fields: (none)

Constructor: NetworkManager()

Methods:
void newConnection(int, int)
void closeConnection()

**Connection** - Handles the data stream between the client and the server.

Fields: (none)

Constructor:
Connection(int,int)
Entity[] getEntities()

## Responsibility List:

Kevin - Game framework, Networking
Sam - Multiple menu screens, graphics, mechanic design.
Eric - Multiple menu screens, graphics, mechanic design.