UML diagram of the project

main.panel

JPanel
...
+ JPanel()
+ JPanel(l: LayoutManager)
+ setPreferredSize(d: Dimension): void
+ setBorder(border: Border): void
...

InstructionPanel
- instructionText: String
- scoringText: String
- instructionLabel: JLabel
- scoringLabel: Label
+ InstructionPanel()

GamePanel
# game: Game
GamePanel()
GamePanel(layout: LayoutManager)
+ addGame(game: Game): void

GameInfoPanel
- HPPanelObj: HPPanel
- scoreLabel: JLabel
+ GameInfoPanel()
+ update(): void

HPPanel
- xOffset: int
- HPBarHeight: int
- edgeDistance: int
+ HPPanel()
# paintComponent(g: Graphics): void

GameMenuPanel
+ FPS30BT: JRadioButton
+ FPS60BT: JRadioButton
+ restartBT1: JButton
+ restartBT2: JButton
+ pauseBT: JButton
+ resumeBT: JButton
+ sensitivitySlider: JSlider
+ pausePanel: JPanel
+ playPanel: JPanel
+ endPanel: JPanel
+ controllerModeSelectionBox: JComboBox
+ GameMenuPanel()
+ changeMenu(status: Status): void
+ addGame(game: Game): void

StatusPanel
- textArea: JTextArea
+ StatusPanel()
+ addText(playerInfo: String): void

DisplayPanel
- bgColor: Color
+ DisplayPanel()
+ addController(controller: AbstractController): void
# paintComponent(g: Graphics): void
- renderTimer(g: Graphics): void

main

«interface» **ActionListener**
+ actionPerformed(e: ActionEvent): void

«interface» **ItemListener**
+ itemStateChanged(e: ItemEvent): void

«interface» **ChangeListener**
+ stateChanged(e: ChangeEvent): void

Game
+ MAX_TIME_SEC
- controller: AbstractController
- player: Player
- score: long
- highestScore: long
- MAX_TOTAL_ASTEROIDS: int
- REGEN_ASTEROID_TIME: double
- regenAsteroidStatus: int
- isPaused: boolean
- isEndChecked: boolean
- updateTimer: Timer
- renderTimer: Timer
- timeSec: int
- gamePanel: DisplayPanel
- statusPanel: StatusPanel
- menuPanel: GameMenuPanel
- gameInfoPanel: GameInfoPanel
- UPS: int
- updateCount: int
- FPS: int
- border: Border
Game(gamePanel: GamePanel,
statusPanel: StatusPanel,
menuPanel: MenuPanel
gameInfoPanel: GameInfoPanel)
+ init(): void
+ update(): void
+ render(): void
+ getTimeLeft(): int
+ getUpdateLeft(): int
+ setPaused(b: boolean): void
+ getPaused(): boolean
+ getPlayer(): Player
+ getBorder(): Border
+ getUPS(): int
+ setUPS(value: int): void
+ setFPS(value: int): void
+ getScore(): long
+ getHighestScore(): long
+ getGamePanelSize(): Dimension
+ renderBullets(g: Graphics): void
+ renderAsteroids(g: Graphics): void
+ actionPerformed(e: ActionEvent)
+ itemStateChanged(e: ItemEvent)
+ stateChanged(e: ChangeEvent)
- restart(): void
- updateTime(): void
- updateBullets(): void
- updateAsteroids(): void
- checkCollision(asteroid: Asteroid): void

«enumeration» **Status**
PLAY
PAUSE
END

GameWindow
- gamePanel: DisplayPanel
- infoPanel: StatusPanel
- menuPanel: GameMenuPanel
- gameInfoPanel: GameInfoPanel
- instructionPanel: InstructionPanel
+ GameWindow(title: String)

AsteroidShooter
+ main(args: String[]): void

objects

Point
x: double
y: double
+ Point()
+ Point(x: double, y: double)
+ getX(): double
+ getY(): double

GameObject
# game: Game
# theta: double
GameObject()
GameObject(x: double, y: double)
GameObject(x: double, y: double,
theta: Double)
GameObject(game: Game)
+ update(): void
+ render(g: Graphics): void
# updatePos(): void

Player
+ isReloadDone: boolean
+ isBoosted: boolean
+ isHit: boolean
- MAX_VELOCITY: int
- RELOAD_TIME: double
- REGEN_BOOST_TIME: double
- MAX_DEATH_FRAME: int
- ACCELERATION: double
- MAX_RELOAD_STATUS: long
- MAX_REGEN_BOOST_STATUS: long
- MAX_BULLETS_SHOT: int
- reloadStatus: long
- regenBoostStatus: long
- wasHit: boolean
- velocityX: double
- velocityY: double
- thetaTrue: double
- thetaUpdate: double
- HP: double
- triggerShoot: boolean
- triggerBoost: boolean
- accelerateDirection: Direction
- max_velocity: int
- fireColor: Color
- playerColorPrimary: Color
- playerColorSecondary: Color
- deathFrame: int
- deathPoints: GameObject[]
+ Player()
+ Player(game: Game)
+ Player(x: int, y: int)
+ render(g: Graphics): void
+ update(): void
+ setThetaUpdate(degree: double): void
+ setThetaUpdateZero(): void
+ setAccelerateDirection(d: Direction): void
+ accelerate(): void
+ setTriggerShoot(b: boolean): void
+ setTriggerBoost(b: boolean): void
+ isHitBorder(border: Border): int
+ getInfo(): String
+ boostSpeed(b: boolean): void
+ getIsDead(): boolean
+ getHP(): double
+ getVelocity(): double
+ bounceAsteroid(asteroid: Asteroid): void
# updatePos(): void
- updateShoot(): void
- updateBoost(): void
- updateTheta(): void
- deathAnimation(g: Graphics): void

Asteroid
- MAX_VELOCITY: int
- MIN_VELOCITY: int
- MAX_ANGULAR_SPEED: double
- MIN_ANGULAR_SPEED: double
- SIZES: String[]
- lifetime: int
- current_lifetime: int
- velocity: int
- theta: double
- rotation: double
- rotationSpeed: double
- HP: double
- size: String
- radius: int
- XP: int
+ Asteroid(game: Game)
+ render(g: Graphics): void
+ update(): void
+ isDead(): boolean
+ isDestroyed(): boolean
+ getRadius(): int
+ deductHP(o: GameObject): void
+ getVelocity(): double
+ getXP(): int
+ getTheta(): double
# updatePos(): void
- pickCoordinate(side: int): Point
- calculateLifetime(startPoint: Point, endPoint:
Point): void

Bullet
- MAX_VELOCITY: double
- velocityX: double
- velocityY: double
- length: double
- bulletColor: Color
Bullet(x: double, y: double,
theta: double, game: Game)
Bullet(x: double, y: double,
theta: double)
+ update(): void
+ render(g: Graphics): void
+ outsideBorder(): boolean
# updatePos(): void

«enumeration» **Direction**
FORWARD
BACKWARD
DEFAULT

Border
- x: int
- y: int
- w: int
- h: int
- strokeCol: Color
+ Border(x: int, y: int, w: int, h: int)
+ Border(upperLeft: Point,
w: int,
h: int)
+ fromCenter(center: Point,
w: int,
h: int): Border
+ fromCenter(Dimension d,
w: int,
h: int): Border
+ render(g: Graphics): void
+ getX(): int
+ getY(): int
+ getWidth(): int
+ getHeight(): int

controller

«interface» **KeyListener**
+ keyPressed(e: KeyEvent): void
+ keyTyped(e: KeyEvent): void
+ keyReleased(e: KeyEvent): void

«interface» **MouseInputListener**
+ mouseReleased(e: MouseEvent): void
+ mouseDragged(e: MouseEvent): void
+ mousePressed(e: MouseEvent): void
+ mouseClicked(e: MouseEvent): void
+ mouseMoved(e: MouseEvent): void
+ mouseEntered(e: MouseEvent): void
+ mouseExited(e: MouseEvent): void

AbstractController
# game: Game
AbstractController()
+ addGame(game: Game)
+ mouseEntered(e: MouseEvent): void
+ mouseExited(e: MouseEvent): void

KeyController
- upBT: int
- downBT: int
- leftBT: int
- rightBT: int
- leftBTOnce: boolean
- rightBTOnce: boolean
- upBTHeld: boolean
- rotateSpeed: double
+ KeyController(mode: String)
+ switchMode(mode: String): void
+ setRotateSpeed(value: int): void
+ mouseReleased(e: MouseEvent): void
+ mouseDragged(e: MouseEvent): void
+ mousePressed(e: MouseEvent): void
+ mouseClicked(e: MouseEvent): void
+ mouseMoved(e: MouseEvent): void
+ keyPressed(e: KeyEvent): void
+ keyTyped(e: KeyEvent): void
+ keyReleased(e: KeyEvent): void

utils

MathUtils
+ getDistance(startPoint: Point, endPoint: Point): double
+ getAngle(centerPoint: Point, endPoint: Point): double

asteroidshooter