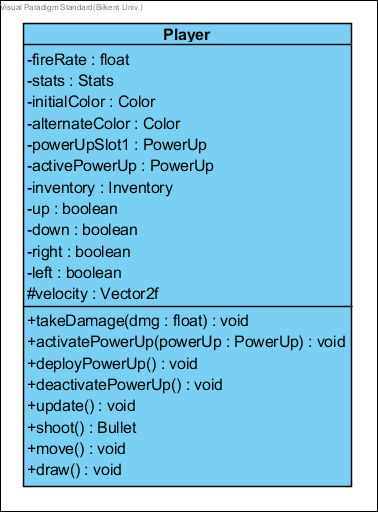
**Player Class**

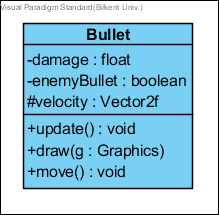
******Player controlled the game object which can move and shoot. ***Attributes:*  
private float fireRate:** To control the density of bullets. **private Stats stats:** players stats as a Stats object. **private ColorinitialColor:** color variable supplied for adrawn method to indicate player objects state changes. **private ColoralternateColor:** alternating color variable from the initial color **private PowerUp powerUpSlot1:** holds the primary and only(unless there is an item that specifies an extra slot) power up. **private PowerUpactivePowerUp:** holds a reference to an active power-up since there might be another one supplied by an item. **private Inventory inventory:** player’s inventory that holds items and num of chests keys credits…  
**private boolean up:** The variable to indicate the direction of upwards movement that is supplied by user input(via GameMaster). **private boolean down:** The variable to indicate the direction of downwards movement that is supplied by user input(via GameMaster). **private boolean right:** The variable to indicate the direction of rightwards movement that is supplied by user input(via GameMaster). **private boolean left:** The variable to indicate the direction of leftwards movement that is supplied by user input(via GameMaster). **protected Vector2f velocity:** the amount of velocity that player has**.**

***Methods:*  
public void takeDamage(float dmg):** the amount of Damage that will be applied to Player if player is killed it will be marked for removal.(calls stats.takeDamage(dmg)) **public void activatePowerUp(PowerUppowerUp):** activates the powerup specified in the parameter. **public void deployPowerUp():** deploys the activated powerUp.(calls activePowerUp.deploy()). **public void deactivatePowerUp():** deactivates the active power up.

**public void update():**It updates the position and the state of the object in every game loop. This method does not return anything. **public Bullet shoot():**It shoots bullets with heavy damage. **public void draw():**It draws the shape of the super class gameObject using dimension and position vectors with Graphics. This method does not return anything.

**public void move():**It moves towards the target position which is a DynamicGameObject(usually the player) until it crashes and collides the target.

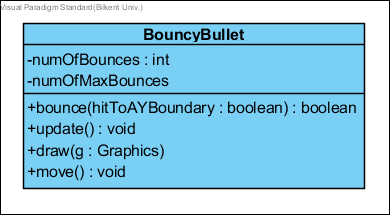
**Bullet Class**



This class constructs the Bulletobject. ***Attributes:*private float damage:** the amount of damage to be applied to the collided enemy.  
**private boolean enemyBullet:** he boolean variable that If it is shot by enemy it is true, : If it is shot by player it if false. **protected Vector2f velocity:** the amount of velocity that bullet has**.**

***Methods:*public void update():**It updates the position and the state of the object in every game loop. This method does not return anything. **public void draw(Graphics g):** It draws the shape of the super class gameObject using dimension and position vectors with Graphics. This method does not return anything. **public void move():**It moves towards the target position which is a DynamicGameObject(usually the player) until it crashes and collides the target.

**BouncyBullet Class**

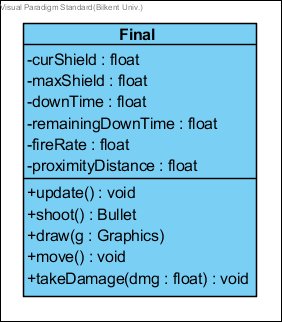
******

This class constructs the BouncyBulletobject.

***Attributes:*private intnumOfBounces:** counter for collisions with game Arena borders.**privateintnumOfMaxBounces:** max number of bounces for a bullet.

***Methods:*public boolean bounce (booleanhitToAYBoundary):** First checks if numOfBounces<=macNumOfBounces then it inverts the bouncy bullet’s velocity’s x value if theparameter is false else it inverts y value.   
**public void update():**It updates the position and the state of the object in every game loop. This method does not return anything. **public void draw(Graphics g):** It draws the shape of the super class gameObject using dimension and position vectors with Graphics. This method does not return anything. **public void move():**It moves towards the target position which is a DynamicGameObject(usually the player) until it crashes and collides the target.

**Final Class**

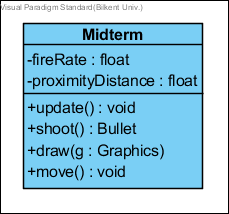
******

This class constructs the Finalobject.

***Attributes:*private float curShield:** holds the current value of the shield.**private float maxShield:**holds the maximum value of the shield.**private float downTime:**holds the initial value to be counted down from before the shield is back up.**private float remainingDownTime:**holds the countdown time for shields to be back up.**private float fireRate:** To control the density of bullets.  
**private float proximityDistance:** The minimum distance between final and the player since this enemy shoots. It does not attack the player by crashing. This value indicates how close can the final get to the player.

***Methods:***  
**public void update():**It updates the position and the state of the object in every game loop. This method does not return anything. **public Bullet shoot():**It shoots bullets with more heavy damage. **public void draw(Graphics g):** It draws the shape of the super class gameObject using dimension and position vectors with Graphics. This method does not return anything. **public void move():**It moves towards the target position which is a DynamicGameObject(usually the player) until it crashes and collides the target. **public void takeDamage(float dmg):** Calls the enemy’s stat objects take damage method which substracts given float value(the parameter) from the current health.

**Midterm Class**

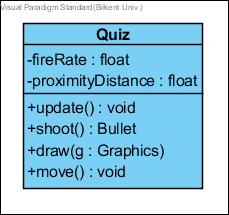


This class constructs the Midtermobject.

***Attributes:*private float fireRate:** To control the density of bullets.  
**private float proximityDistance:** The minimum distance between midterm and the player since this enemy shoots. It does not attack the player by crashing. This value indicates how close can the midterm get to the player.

***Methods:***  
**public void update():**It updates the position and the state of the object in every game loop. This method does not return anything. **public Bullet shoot():**It shoots bullets with heavy damage. **public void draw(Graphics g):** It draws the shape of the super class gameObject using dimension and position vectors with Graphics. This method does not return anything. **public void move():**It moves towards the target position which is a DynamicGameObject(usually the player) until it crashes and collides the target.

**Quiz Class**

******

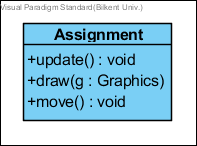
This class constructs the Quizobject.

***Attributes:*private float fireRate:** To control the density of bullets.  
**private float proximityDistance:** The minimum distance between quiz and the player since this enemy shoots. It does not attack the player by crashing. This value indicates how close can the quiz get to the player.

***Methods:***  
**public void update():**It updates the position and the state of the object in every game loop. This method does not return anything. **public Bullet shoot():** It shoots bullets with average damage. **public void draw(Graphics g):** It draws the shape of the super class gameObject using dimension and position vectors with Graphics. This method does not return anything. **public void move():**It moves towards the target position which is a DynamicGameObject(usually the player) until it crashes and collides the target.

**/////////////////////////////////**

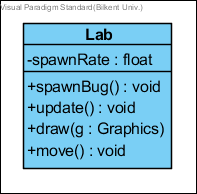
**Assignment Class**

******

This class constructs the Assignmentobject.

***Methods:***  
**public void update():**It updates the position and the state of the object in every game loop. This method does not return anything. **public void draw(Graphics g):** It draws the shape of the super class gameObject using dimension and position vectors with Graphics. This method does not return anything. **public void move():**It moves towards the target position which is a DynamicGameObject(usually the player) until it crashes and collides the target.

**Lab Class**

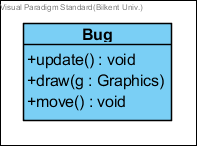
******

This class constructs the Lab object.

***Attributes:*private float spawnRate:** To control the density of “bug” objects spawned by thelab.

***Methods:*public spawnBug():** spawns bugs to attack player  
**public void update():**It updates the position and the state of the object in every game loop. This method does not return anything. **public void draw(Graphics g):** It draws the shape of the super class gameObject using dimension and position vectors with Graphics. This method does not return anything. **public void move():**It moves towards the target position which is a DynamicGameObject(usually the player) until it crashes and collides the target.

**Bug Class**



This class constructs the Bug object.

***Methods:***  
**public void update():** It updates the position and the state of the object in every game loop. This method does not return anything. **public void draw(Graphics g):** It draws the shape of the super class gameObject using dimension and position vectors with Graphics. In case of Bug this simply draws a green square with the given dimension and position. **public void move():** It moves towards the target position which is a DynamicGameObject(usually the player) until it crashes and collides the target.

**Chest Class**

****

This class constructs the chests. Chest Class is a parent class and have four children classes are Freshmen Chest, Sophomore Chest, Junior Chest, and Senior Chest. ***Attributes:*  
private int numOfKeysNeed**: It holds a number of the keys needed.

***Methods:*public boolean unlock():**This method unlocks the chest and it returns whether the chest is unlocked or not.

**public void update() :** It updates the position and the state of the object in every game loop. This method does not return anything.

**Freshmen Chest Class**

****This class constructs the freshmen chest object.

***Methods:***

**public boolean collect() :** It collects the item. This method returns true if it collected the item otherwise returns false. It adds the power up to the player power up.

**public void update() :** It updates the position and the state of the object in every game loop. This method does not return anything.

**public void draw(Graphics g) :** It draws the shape of the super class gameObject using dimension and position vectors with Graphics. This method does not return anything.

**public Item unlock() :** It opens the chest object. It has 90% chance of giving a standard tier item, 7% chance of giving a rare tier item and 3% chance of giving an ultra-rare tier item.

**Sophomore Chest Class**

****This class constructs the sophomorechest object.

***Methods:***

**public boolean collect() :** It collects the item. This method returns true if it collected the item otherwise returns false. It adds the power up to the player power up.

**public void update() :** It updates the position and the state of the object in every game loop. This method does not return anything.

**public void draw(Graphics g) :** It draws the shape of the super class gameObject using dimension and position vectors with Graphics. This method does not return anything.

**public Item unlock() :** It opens the chest object. It has 50% chance of giving a standard tier item, 30% chance of giving a rare tier item and 20% chance of giving an ultra-rare tier item.

**Junior Chest Class**

****This class constructs the junior chest object.

***Methods:***

**public boolean collect() :** It collects the item. This method returns true if it collected the item otherwise returns false. It adds the power up to the player power up.

**public void update() :** It updates the position and the state of the object in every game loop. This method does not return anything.

**public void draw(Graphics g) :** It draws the shape of the super class gameObject using dimension and position vectors with Graphics. This method does not return anything.

**public Item unlock() :** It opens the chest object. It has 30% chance of giving a standard tier item, 35% chance of giving a rare tier item, 25% chance of giving an ultra-rare tier item, 10% chance of giving a

“hacker” tier item.

**Senior Chest Class**

****This class constructs the senior chest object.

***Methods:***

**public boolean collect() :** It collects the item. This method returns true if it collected the item otherwise returns false. It adds the power up to the player power up.

**public void update() :** It updates the position and the state of the object in every game loop. This method does not return anything.

**public void draw(Graphics g) :** It draws the shape of the super class gameObject using dimension and position vectors with Graphics. This method does not return anything.

**public Item unlock() :** It opens the chest object. It has 18% chance of giving a standard tier item, 25% chance of giving a rare tier item, 32% chance of giving an ultra-rare tier item, 25% chance of giving a “hacker” tier item.

**Coin Class**

****

This class constructs the coin object. ***Attributes:***

**Private int value:** It holds a number of coins.

***Methods:***

**public boolean collect() :** It collects the item. This method returns true if it collected the item otherwise returns false. It adds the power up to the player power up.

**public void update() :** It updates the position and the state of the object in every game loop. This method does not return anything.

**public void draw(Graphics g) :** It draws the shape of the super class gameObject using dimension and position vectors with Graphics. This method does not return anything.

**Item Class**

****

This class constructs the items. Item Class is a parent class and have three children classes are Standard Item, Ultra Rare Item, Hacker Item, and Rare Item.

***Attributes:***

**private int cost**:It holds the cost of the item.

**private String name :** It holds the name of the item.

**private String description :** It holds the description of the item.

***Methods:***

**Public void affect():**It sends the effect to the game master.

**Standard Item Class**

****This class constructs the standard items

**Ultra Rare Item Class**

****This class constructs the ultra-rare items

**Rare Item Class**

****This class constructs the rare items

**Hacker Item Class**

****This class constructs the hacker items.

**Collectable Interface**



Collectable is an interface for all collectible items in the game. These are Bonus and PowerUps.

***Methods:***

**public boolean collect():**is called when the player collides any Collectable items in the game.

**Bonus Class**



Bonus Class is a parent class and have three children classes are Chest, Coin, and Key.

***Attributes:***

**private Boolean pickedUp:**is determine whether the Bonus item is picked up by player or not.

***Methods:***

**public void update() :** It updates the position and the state of the object in every game loop. This method does not return anything.

**Key Class**



This class constructs the key.

***Attributes:***

**private float duration:**it is specified the duration of existence of a key in the game.

**private float remainingTime:**it is specified the remaining time from the duration of the key.

***Methods:***

**public boolean collect() :** It collects the item. This method returns true if it collected the item otherwise returns false. It adds the power up to the player power up.

**public void update() :** It updates the position and the state of the object in every game loop. This method does not return anything.

**public void draw(Graphics g) :** It draws the shape of the super class gameObject using dimension and position vectors with Graphics. This method does not return anything.

**PowerUp Class**



PoverUp Class is a parent class and has five children classes are ExtraTime, SlowTime, BouncyBullets, BulletBlast, and DamageIncrese.

***Attributes:***

**private boolean active:** it specifies whether the power-up is active or not.

**private float duration:** it specifies the duration of existence of power-up in the game.

**private float pickedUp:** it specifies whether the power-up is picked-up by player or not

***Methods:***

**public void deploy():** this method deploys the power-up when the player collides power-up to use in the game.

**public void update() :** It updates the position and the state of the object in every game loop. This method does not return anything.

**ExtraTime Class**



ExtraTime Class adds the extra time to player’s game-time

***Attributes:***

**private float amount:** it specifies the amount of time which is added to the player.

***Methods:***

**public boolean collect() :** It collects the item. This method returns true if it collected the item otherwise returns false. It adds the power up to the player power up.

**public void update() :** It updates the position and the state of the object in every game loop. This method does not return anything.

**public void draw(Graphics g) :** It draws the shape of the super class gameObject using dimension and position vectors with Graphics. This method does not return anything.

**SlowTime Class**



SlowTime Class slows the time for all game objects.

***Attributes:***

**private float multiplier:** it specifies the ratio of the slowing time.

***Methods:***

**public boolean collect() :** It collects the item. This method returns true if it collected the item otherwise returns false. It adds the power up to the player power up.

**public void update() :** It updates the position and the state of the object in every game loop. This method does not return anything.

**public void draw(Graphics g) :** It draws the shape of the super class gameObject using dimension and position vectors with Graphics. This method does not return anything.

**BouncyBullets Class**



BouncyBullets Class replaces the bullets with the bouncy bullets which bounce off (3 times per bullet) from the borders of the game are instead of simply going out.

***Methods:***

**public boolean collect() :** It collects the item. This method returns true if it collected the item otherwise returns false. It adds the power up to the player power up.

**public void update() :** It updates the position and the state of the object in every game loop. This method does not return anything.

**public void draw(Graphics g) :** It draws the shape of the super class gameObject using dimension and position vectors with Graphics. This method does not return anything.

**BulletBlast Class**



BulletBlast Class sends out a circular group of bullets originating from the player.

***Methods:***

**public boolean collect() :** It collects the item. This method returns true if it collected the item otherwise returns false. It adds the power up to the player power up.

**public void update() :** It updates the position and the state of the object in every game loop. This method does not return anything.

**public void draw(Graphics g) :** It draws the shape of the super class gameObject using dimension and position vectors with Graphics. This method does not return anything.

**DamageIncrease Class**



DamageIncrease Class makes Player’s bullets harder for 5 seconds.

***Attributes:***

**private float multiplier:** it specifies the ratio of the increasing damage.

***Methods:***

**public boolean collect() :** It collects the item. This method returns true if it collected the item otherwise returns false. It adds the power up to the player power up.

**public void update() :** It updates the position and the state of the object in every game loop. This method does not return anything.

**public void draw(Graphics g) :** It draws the shape of the super class gameObject using dimension and position vectors with Graphics. This method does not return anything.