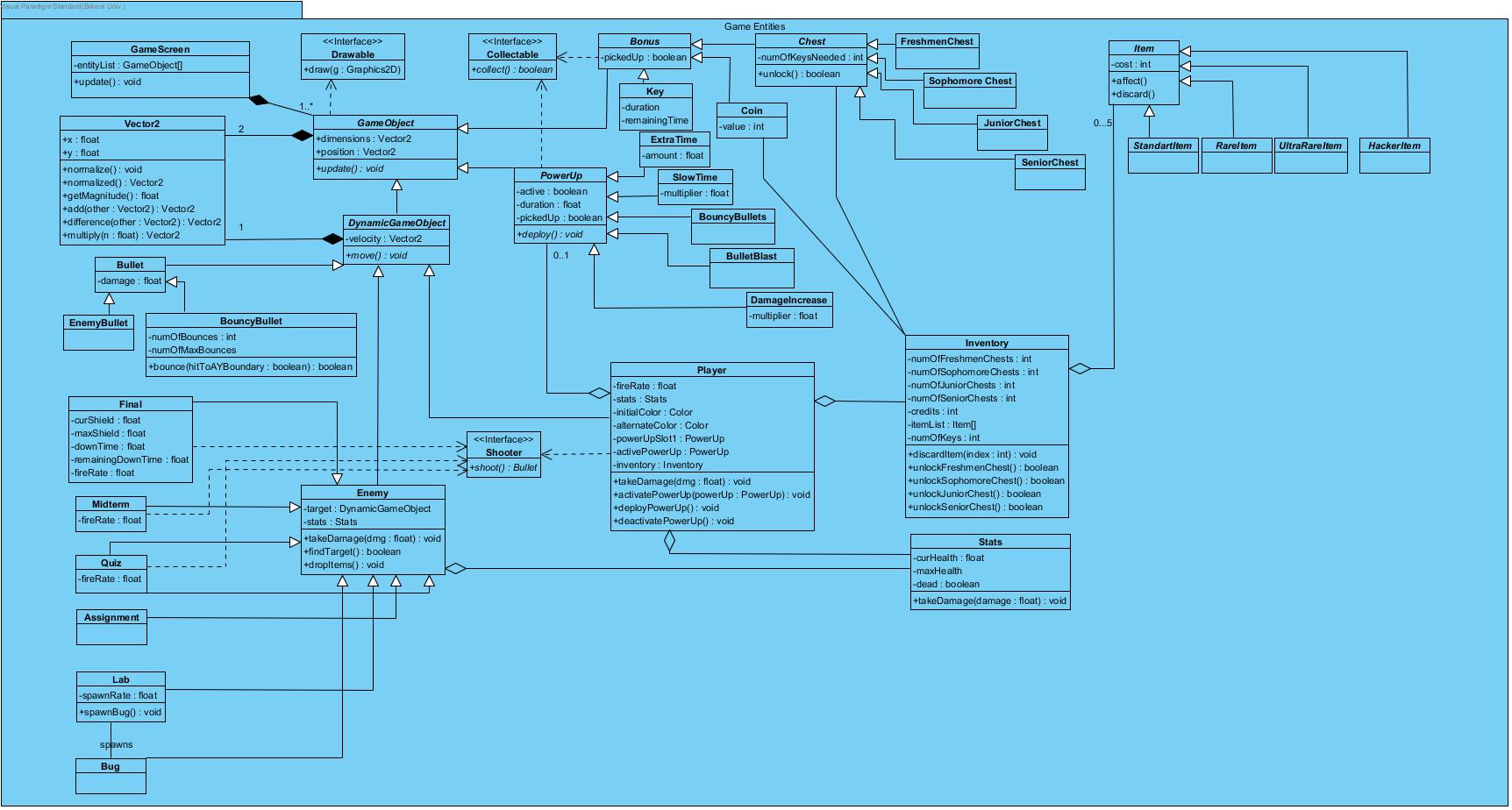
# Game Entities Package



# GameScreen Class



#### Attributes:

**private ArrayList<GameObject> entityList:** Holds the entity(GameObjects) objects of the game.

#### Methods:

**public void update():** calls the update method of every entity object.

# Vector2



#### Attributes:

**public float x:** holds the x value to a 2D space.

**public float y:** holds the y value to a 2D space.

#### Methods:

**public void normalize():** transforms this vector2 to a “unit” vector2.

**public Vector2 normalized():** returns a unit vector2 with same direction of this.

**public float getMagnitude():** returns the magnitude ((x^2+y^2)^½) of this.

**public Vetor2 add(Vector2 other):** adds this Vector2 to other and returns a new Vector2.

**public Vetor2 difference(Vector2 other):** subtracts this vector2 from other and returns a new Vector2.

**public Vector2 multiply(float n):** Multiplies x and y with n and returns a new Vector2.

# Drawable (interface)



#### Methods:

**public void draw():** overrided by gameObjects a simple draw method to visualize objects.

# GameObject Class (abstract)



#### Attributes:

**public Vector2 dimensions:** rectangle sizes of a GameObject.

**public Vector2 position:** center position of a GameObject.

#### Methods:

**public void update():** updates the information accordingly to the internal commands.

# DynamicGameObject (Abstract)



#### Attributes:

**private Vector2 velocity:** holds 2D velocity of a moving object.

#### Methods:

**private void move():** updates the location by velocity.

# Stats Class



#### Attributes:

**private float curHealth:**

**private float maxHealth:**

These are all values for health stats.

**private boolean dead():** flag value for removal from the gameScreen.

#### Methods:

**public void takeDamage(float damage):** decrements the curHealth by damage. Checks dead flag true if curHealt<0.

# Enemy Class



#### Attributes:

**private DynamicGameObject target:** the target of the enemy which will be attacked and possibly damaged.

**private Stats stats:** stats objects for the enemy.

#### Methods:

**public void takeDamage(float dmg):** the amount of Damage that will be applied to Enemy if enemy is killed it will be marked for removal.(calls stats.takeDamage(dmg))

**public boolean findTarget():** Enemy seeks target.

**public void dropItems():** Enemy drops items before getting removed because of death.

# Inventory Class

# Entity Subsystem Class Diagram.jpg

#### Attributes:

**private int numOfFreshmenChests:**

**private int numOfSophomoreChests:**

**private int numOfJuniorChests:**

**private int numOfSeniorChests:**

these specify the number of chests from each type which are collected and not opened.

**private int credits :** number of coins collected.

**private Item[] itemList:** 5 items that are owned by player.

**Private int numOfKeys:**

#### Methods:

**public void discardItem(int index) :** removes item in the given index.

**public boolean FreshmenChest():**instantiates and opens a freshmenChest.

**public boolean SophomoreChest():**instantiates and opens a SophomoreChest.

**public boolean JuniorChest():**instantiates and opens a JuniorChest.

**public boolean SeniorChest():** instantiates and opens a SeniorChest.

# Shooter(interface)



#### Methods:

**public Bullet shoot():** An abstract shoot method for shooting player and enemies.

# Player Class



#### Attiributes:

**private float fireRate:** To control the density of bullets.

**private Stats stats:** players stats as a Stats object.

**private Color initialColor:**

**private Color alternateColor:**

Color variables are supplied for drawn method to indicate player objects state changes.

**private PowerUp powerUpSlot1:** holds the primary and only(unless there is a item that specifies an extra slot) power up.

**private PowerUp activePowerUp:** holds a reference to a the 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…

#### 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(PowerUp powerUp):** 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.

# Bullet Class



#### Attiributes:

**private float damage:** the amount of damage to be applied to the collided enemy.

# BouncyBullet Class



#### Attributes:

**private int numOfBounces:** counter for collisions with game Arena borders.

**private int numOfMaxBounces:** max number of bounces for a bullet.

#### Methods:

**public boolean bounce (boolean hitToAYBoundary):** First checks if numOfBounces<=macNumOfBounces then it inverts the bouncy bullet’s velocity’s x value if parameter is false else it inverts y value.

# Final Class



#### Attributes:

**private float curShield:** holds the current value for the shield.

**private float maxShield:** holds the maximum value for 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.

# Midterm Class



#### Attributes:

**private float fireRate:** To control the density of bullets.

# Quizz Class



#### Attributes:

**private float fireRate:** To control the density of bullets.

# Assignment Class



# Lab Class



#### Attributes:

**private float spawnRate:** To control the density of “bug” ojects spawned by lab.

#### Methods:

**public spawnBug():** spawns bugs to attack player

# Bug Class

