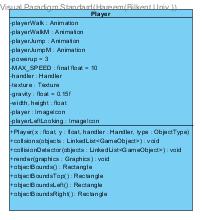
#### Player class



The Player class is responsible for creating the player object, which represents our protagonist. This class is responsible for providing the control of player, display information and the rule checking of the interaction of player with other objects. The player class provides this data to the game engine to be displayed.

*Constructor*

**public Player( float x, float y, Handler handler, ObjectType type ) :** thisconstructor creates the player object, by initializing the player with default values for the first time and initializing the object instances of Animation for walking, running and shooting, the Camera object for positioning the screen and the weapon objects.

*Attributes*

**private int powerUp :** This attribute holds the amount of the power up player has.

**private final float MAX\_SPEED :** This attribute holds the value of the maximum speed at which the player moves, the speed is determined by the input given by the user.

**private Handler handler:**

**Texture texture :**

**ImageIcon player** : This atribute holds the player image.

**ImageIcon playerLeftLooking :**

**private float gravity:** This attribute holds the speed with which the player falls down.

**private float width ,height :**

**private int gravity:** this attribute holds the speed with which the player falls down.

**private Animation playerWalk:** This attribute holds the instance of the Animation object for the walking animation of the player to the right.

**private Animation playerJump:** This attribute holds the instance of the Animation object for the jumping animation of the player to the right.

**private Animation playerWalkM:** This attribute holds the instance of the Animation object for the walking animation of the player to the left.

**private Animation playerJumpM:** This attribute holds the instance of the Animation object for the jumping animation of the player to the left.

*Methods*

**public void collisions( LinkedList objects, LinkedList enemies ):** This method is responsible for checking if the player has collided with an enemy or an object(i.e, powerUp or obstacle). It creates the actions according to variable that is collided (Suc as increasing the powerup amount when player has a collision with powerup).

**public void walkAnimation(Animation anim):** this method is responsible for running the animation of the player walking upon given input.

**public void runAnimation(Animation anim):** This method is responsible for running the animation of the player running upon given input.

**public void shootAnimation(Animation anim):** This method is responsible for running the animation of the player shooting upon given input.

**public float getGravity():** This methos returns the float gravity.

**public void setGravity(int gravity) :** This method sets the players gravity to the gravity given as parameter.

**public Handler getHandler():** This methos returns the handler.

**public Rectangle objectBoundsLeft():**

**public Rectangle objectBoundsTop():**

**public Rectangle objectBoundsRight():**

**public void collisionDetector(LinkedList<GameObject> objects):** This method overrides the collisionDetector method in GameObject class.

**public void render(Graphics graphics):** This method overrides the render method in GameObject class.

**public Rectangle objectBounds():** This method overrides the objectBounds method in GameObject class. The method creates and returns a Rectangle to determine the object Bounds.