```java
package com.blindtigergames.werescrewed.entity.mover;

import com.badlogic.gdx.math.Vector2;
import com.badlogic.gdx.physics.box2d.Body;
import com.blindtigergames.werescrewed.entity.Entity;
import com.blindtigergames.werescrewed.entity.platforms.Platform;
import com.blindtigergames.werescrewed.entity.screws.Screw;

public class DirectionFlipMover implements IMover {

    boolean moveLeft;
    Vector2 impulse;
    float prevXPosMeter;
    float accum, timeToFlipAfterNoMove, maxSpeed;

    //private variables to prevent re-allocating them each time move() is called
    private float pos, diff, len;

    /**
     * Attach this mover to a dynamic body. IT will roll it left and right, and
flip directions if stuck on a wall
     * @param moveLeft Starting direction
     * @param impulseStrength 0.001f is a good slow acceleration speed
     * @param entityToMove Must be dynamic
     * @param timeToFlipAfterNoMove seconds to flip after being stuck on a wall.
1.5 is a good time.
     * @param maxSpeed 0.03 is a good speed
     */
    public DirectionFlipMover(boolean moveLeft, float impulseStrength, Entity
entityToMove, float timeToFlipAfterNoMove, float maxSpeed){
        this.moveLeft=moveLeft;
        this.impulse=new Vector2(impulseStrength,0);
        if(moveLeft)impulse.x*=-1;
        this.prevXPosMeter = entityToMove.getPosition( ).x;
        this.accum = 0;
        this.timeToFlipAfterNoMove=timeToFlipAfterNoMove;
        this.maxSpeed=maxSpeed;
    }

    /**
     * Initialize this mover with default values
     * @param moveLeft
     * @param entityToMove
     */
    public DirectionFlipMover(boolean moveLeft, Entity entityToMove){
        this(moveLeft, 0.001f, entityToMove, 1.5f, .03f);
    }


    @Override
    public void move( float deltaTime, Body body ) {
        pos = body.getPosition( ).x;
        diff = pos- prevXPosMeter ;
        len = Math.abs( diff );
        if (len< 0.01f){ //0.01 means the enemy hasn't move much
            accum+=deltaTime;
        }
        prevXPosMeter=pos;
        if(accum>timeToFlipAfterNoMove){
            moveLeft = !moveLeft;
            accum = 0;
            impulse.x=impulse.x*-1;
        }
        if(len<maxSpeed){
            body.applyLinearImpulse( impulse, body.getWorldCenter( ) );
        }
    }

    @Override
    public void runPuzzleMovement( Screw screw, float screwVal, Platform p ) {
        // TODO Auto-generated method stub

    }

    @Override
    public PuzzleType getMoverType( ) {
        // TODO Auto-generated method stub
        return PuzzleType.OVERRIDE_ENTITY_MOVER;
    }

}
```