```
00001: package hevs.fragil.patapon.physics;
00002:
00003: import com.badlogic.gdx.math.Vector2;
00004:
00005: import ch.hevs.gdx2d.components.physics.primitives.PhysicsStaticBox;
00006: import ch.hevs.gdx2d.lib.GdxGraphics;
00007: import ch.hevs.gdx2d.lib.interfaces.DrawableObject;
00008: import hevs.fragil.patapon.drawables.SpriteSheet;
00009: import hevs.fragil.patapon.mechanics.Param;
00010:
00011: public abstract class Tower extends PhysicsStaticBox implements DrawableObject, CollidedObject{
00012:     private int x;
00013:     private int h;
00014:     private float life = 10;
00015:     protected SpriteSheet basis1;
00016:     protected SpriteSheet basis2;
00017:     protected SpriteSheet head;
00018:
00019:     public Tower(int x, int h){
00020:         super("tower", new Vector2(x, Param.FLOOR_DEPTH + h/2*20), 100, h*20);
00021:         this.h = h;
00022:         this.x = x;
00023:         this.setCollisionGroup(-4);
00024:         loadFiles();
00025:     }
00026:
00027:     @Override
00028:     public void draw(GdxGraphics g) {
00029:         for(int i = 0 ; i < h ; i++){
00030:             if(i%2 == 0)
00031:                 basis1.drawFrame(0, (int)(x - g.getCamera().position.x + Param.CAM_WIDTH / 2)-50, Param.FLOOR_DEPTH + i*20);
00032:             else
00033:                 basis2.drawFrame(0, (int)(x - g.getCamera().position.x + Param.CAM_WIDTH / 2)-50, Param.FLOOR_DEPTH + i*20);
```

```
00034:         }
00035:         head.drawFrame(0, (int)(x - g.getCamera().position.x + Param.CAM_WIDTH / 2)-50, Param.FLOOR_DEPTH + h*20);
00036:
00037:     }
00038:     @Override
00039:     public int getCollisionGroup() {
00040:         return 0;
00041:     }
00042:
00043:     @Override
00044:     public boolean applyDamage(float damage) {
00045:         if(life <= 0){
00046:             return true;
00047:         }
00048:         else{
00049:             life -= damage;
00050:         }
00051:         return false;
00052:     }
00053:     public boolean isExploded(){
00054:         if(life<=0){
00055:             return true;
00056:         }
00057:         else return false;
00058:     }
00059:     public int getPos(){
00060:         return x;
00061:     }
00062:     public int getHeight(){
00063:         return h;
00064:     }
00065:     public abstract void loadFiles();
00066:
```

```
00067:     public boolean isOccuped(int posToTry) {
00068:         if(posToTry < x+50 && posToTry > x-50)
00069:             return true;
00070:         return false;
00071:     }
00072:     public int getLeftLimit(){
00073:         return x-50;
00074:     }
00075: }
```