```
00001: package hevs.fragil.patapon.drawables;
00002:
00003:
00004: import com.badlogic.gdx.Gdx;
00005: import com.badlogic.gdx.graphics.Texture;
00006: import com.badlogic.gdx.graphics.g2d.Animation;
00007: import com.badlogic.gdx.graphics.g2d.Animation.PlayMode;
00008: import com.badlogic.gdx.graphics.g2d.Sprite;
00009: import com.badlogic.gdx.graphics.g2d.SpriteBatch;
00010: import com.badlogic.gdx.graphics.g2d.TextureRegion;
00011: import com.badlogic.gdx.math.Vector2;
00012:
00013: /**
00014:  * SpriteSheet management class to provide easy drawing and initialization
00015:  * @author LoÃ¯c Gillioz
00016:  */
00017: public class SpriteSheet {
00018:     Animation animation;
00019:     Texture sheet;
00020:     Sprite[] sprites;
00021:     SpriteBatch spriteBatch;
00022:     TextureRegion currentFrame;
00023:
00024:     float frameDuration;
00025:     float preAnimDelay = 0f;
00026:     private boolean flipped = false;
00027:
00028:     /**
00029:      * Initialize a new Spritesheet
00030:      * @param url : the image location
00031:      * @param cols : number of cols in your spritesheet image
00032:      * @param rows : number of rows in your spritesheet image
00033:      * @param frameDuration : duration of each frame of the animation
```

```
00034:      * @param flipped : true if flipped
00035:      * @param looping : true if looping animation
00036:      */
00037:     public SpriteSheet(String url, int cols, int rows, float frameDuration, boolean flipped, PlayMode playMode){
00038:         this.flipped = flipped;
00039:         this.frameDuration = frameDuration;
00040:
00041:         sheet = new Texture(Gdx.files.internal(url));
00042:         TextureRegion[][] tmp = TextureRegion.split(sheet, sheet.getWidth()/cols, sheet.getHeight()/rows);
00043:         sprites = new Sprite[cols * rows];
00044:         int index = 0;
00045:         for (int i = 0; i < rows; i++) {
00046:             for (int j = 0; j < cols; j++) {
00047:                 sprites[index++] = new Sprite(tmp[i][j]);
00048:             }
00049:         }
00050:
00051:         animation = new Animation(frameDuration, sprites);
00052:         spriteBatch = new SpriteBatch();
00053:
00054:         switch(playMode){
00055:         case LOOP             :    animation.setPlayMode(PlayMode.LOOP);
00056:                                    break;
00057:         case LOOP_PINGPONG    :    animation.setPlayMode(PlayMode.LOOP);
00058:                                    break;
00059:         case LOOP_RANDOM      :    animation.setPlayMode(PlayMode.LOOP);
00060:                                    break;
00061:         case LOOP_REVERSED    :    animation.setPlayMode(PlayMode.LOOP);
00062:                                    break;
00063:         case NORMAL           :    animation.setPlayMode(PlayMode.LOOP);
00064:                                    break;
00065:         case REVERSED         :    animation.setPlayMode(PlayMode.LOOP);
00066:                                    break;
```

```
00067:         }
00068:     }
00069:     /**
00070:      * Draws a frame at given position
00071:      * @param frameIndex : index of the frame in the spritesheet (left to right, up to down)
00072:      * @param posX : x location in pixels
00073:      * @param posY : y location in pixels
00074:      */
00075:     public void drawFrame(int frameIndex, int posX, int posY){
00076:         spriteBatch.begin();
00077:         Sprite tmp = sprites[frameIndex];
00078:         if(flipped){
00079:             tmp.setOrigin(96+32, 58);
00080:             tmp.setPosition(posX-32, posY);
00081: 
00082:         }
00083:         else {
00084:             tmp.setOrigin(96, 58);
00085:             tmp.setPosition(posX, posY);
00086:         }
00087:         tmp.setRotation(0);
00088:         if(flipped && tmp.isFlipX() == false)
00089:             tmp.flip(true, false);
00090:         tmp.draw(spriteBatch);
00091:         spriteBatch.end();
00092:     }
00093:     /**
00094:      * Draws a special walk animation for unit bodies, simulates a walk effect by applying rotations
00095:      * @param walkIndex : index of the walk animation from the legs
00096:      * @param spriteNumber : frame to draw (in the body spritesheet)
00097:      * @param posX : draw location in x
00098:      * @param posY : draw location in y
00099:      * @param originY : y rotation center
```

```
00100:      */
00101:     public void drawWalkAnimation(int walkIndex, int spriteNumber, float posX, float posY){
00102:         spriteBatch.begin();
00103:         Sprite tmp = sprites[spriteNumber];
00104:         float angle = 0f;
00105:         switch(walkIndex){
00106:             case 0 :     angle = -3;
00107:                         posY -= 3;
00108:                         break;
00109:             case 1 :     angle = 0;
00110:                         break;
00111:             case 2 :     angle = 3;
00112:                         posY -= 3;
00113:                         break;
00114:             case 3 :     angle = 0;
00115:                         break;
00116:         }
00117:         if(flipped){
00118:             tmp.setOrigin(96+32, 58);
00119:             tmp.setPosition(posX-32-96, posY);
00120:
00121:         }
00122:         else {
00123:             tmp.setOrigin(96, 58);
00124:             tmp.setPosition(posX-96, posY);
00125:         }
00126:         tmp.setRotation(angle);
00127: //        if(flipped && tmp.isFlipX() == false)
00128: //            tmp.flip(true, false);
00129:         tmp.draw(spriteBatch);
00130:         spriteBatch.end();
00131:     }
00132:     /**
```

```
00133:      * Draws a rotated frame with a given angle
00134:      * @param spriteNumber : index of the frame in the spritesheet table
00135:      * @param angle : rotation angle applied in radians
00136:      * @param posX : draw location in x
00137:      * @param posY : draw location in y
00138:      */
00139:     public void drawRotatedFrame(int spriteNumber, float angle, float posX, float posY){
00140:         spriteBatch.begin();
00141:         Sprite tmp = sprites[spriteNumber];
00142:         Vector2 offset = new Vector2(0, (float)Math.sin(angle)*30);
00143:         if(flipped){
00144:             tmp.setOrigin(96+32, 58);
00145:             tmp.setPosition(posX-32-96 + offset.x, posY + offset.y);
00146:
00147:         }
00148:         else {
00149:             tmp.setOrigin(96, 58);
00150:             tmp.setPosition(posX-96 + offset.x, posY + offset.y);
00151:         }
00152:         tmp.setRotation((float)Math.toDegrees(angle));
00153:         tmp.draw(spriteBatch);
00154:         spriteBatch.end();
00155:     }
00156:     public int drawAllFrames(float time, float posX, float posY){
00157:         currentFrame = animation.getKeyFrame(time, true);
00158:         TextureRegion[] a = animation.getKeyFrames();
00159:         int index = java.util.Arrays.asList(a).indexOf(currentFrame);
00160:         spriteBatch.begin();
00161:         Sprite tmp = sprites[index];
00162:         if(flipped){
00163:             tmp.setOrigin(96+32, 58);
00164:             tmp.setPosition(posX-32-96, posY);
00165:
```

```
00166:         }
00167:         else {
00168:             tmp.setOrigin(96, 58);
00169:             tmp.setPosition(posX-96, posY);
00170:         }
00171:         if(flipped && tmp.isFlipX() == false)
00172:             tmp.flip(true, false);
00173:         tmp.draw(spriteBatch);
00174:         spriteBatch.end();
00175:         return index;
00176:     }
00177:     public int drawAllFrames(float time, Vector2 pos){
00178:         return drawAllFrames(time, pos.x, pos.y);
00179:     }
00180:     public int drawFrames(float time, int startIndex, int nbFrames, float posX, float posY){
00181:         currentFrame = animation.getKeyFrame(time, true);
00182:         TextureRegion[] a = animation.getKeyFrames();
00183:         int index = java.util.Arrays.asList(a).indexOf(currentFrame);
00184:         index = index % nbFrames;
00185:         index += startIndex;
00186:         spriteBatch.begin();
00187:         Sprite tmp = sprites[index];
00188:         if(flipped){
00189:             tmp.setOrigin(96+32, 58);
00190:             tmp.setPosition(posX-32-96, posY);
00191:
00192:         }
00193:         else {
00194:             tmp.setOrigin(96, 58);
00195:             tmp.setPosition(posX-96, posY);
00196:         }
00197:         if(flipped && tmp.isFlipX() == false)
00198:             tmp.flip(true, false);
```

```
00199:         tmp.draw(spriteBatch);
00200:         spriteBatch.end();
00201:         return index;
00202:     }
00203:     public int drawFrames(float stateTime, int startIndex, int nbFrames, Vector2 pos) {
00204:         return drawFrames(stateTime, startIndex, nbFrames, pos.x, pos.y);
00205:     }
00206:     public void drawFrameAlpha(int frameIndex, int posX, int posY, float alpha) {
00207:         spriteBatch.begin();
00208:         Sprite tmp = sprites[frameIndex];
00209:         if(flipped){
00210:             tmp.setOrigin(96+32, 58);
00211:             tmp.setPosition(posX-32-96, posY);
00212:
00213:         }
00214:         else {
00215:             tmp.setOrigin(96, 58);
00216:             tmp.setPosition(posX-96, posY);
00217:         }
00218:         if(flipped && tmp.isFlipX() == false)
00219:             tmp.flip(true, false);
00220:         tmp.draw(spriteBatch, alpha);
00221:         spriteBatch.end();
00222:     }
00223:     public void drawRotatedFrameAlpha(int spriteNumber, float angle, float posX, float posY, float alpha) {
00224:         spriteBatch.begin();
00225:         Sprite tmp = sprites[spriteNumber];
00226:         if(flipped){
00227:             tmp.setOrigin(96+32, 58);
00228:             tmp.setPosition(posX-32-96, posY);
00229:
00230:         }
00231:         else {
```

```
00232:             tmp.setOrigin(96, 58);
00233:             tmp.setPosition(posX-96, posY);
00234:         }
00235:         //rotate the sprite around the left down corner
00236:         tmp.setRotation((float)Math.toDegrees(angle));
00237:         if(flipped && tmp.isFlipX() == false)
00238:             tmp.flip(true, false);
00239: //        //location of the left down corner depends of offsets :
00240: //        float offsetX = 32 - tmp.getWidth()/2;
00241: //        float offsetY = tmp.getHeight()/2 - 38;
00242: //        Vector2 offset = new Vector2(offsetX, offsetY);
00243: //        Vector2 pos = new Vector2(posX, posY);
00244: //        offset.rotateRad(angle);
00245: //        pos.add(offset);
00246:         tmp.draw(spriteBatch, alpha);
00247:         spriteBatch.end();
00248:     }
00249:     public float getFrameDuration() {
00250:         return frameDuration;
00251:     }
00252:     public float getPreAnimDelay() {
00253:         return preAnimDelay;
00254:     }
00255:     public boolean finished(float stateTime) {
00256:         return animation.isAnimationFinished(stateTime);
00257:     }
00258:     public void drawRotatedFrameAlpha(int spriteNumber, float angle, Vector2 pos, float alpha){
00259:         drawRotatedFrameAlpha(spriteNumber, angle, pos.x, pos.y, alpha);
00260:     }
00261: }
```