Jogo Básico LibGDX Parte II (baseado em A simple game - https://libgdx.com/dev/simple-game/)

Movimentando o balde (mouse ou toque de tela)

Se o jogador tocar a tela (ou clicar o mouse), queremos que o balde seja deslocado horizontalmente e fique centralizado nessa posição.

```
// processa a entrada do usuário
if(Gdx.input.isTouched()) { // se a tela foi tocada…
   Vector3 posicaoDeToque = new Vector3();
   posicaoDeToque.set(Gdx.input.getX(), Gdx.input.getY(), 0);
   camera.unproject(posicaoDeToque);
   balde.x = posicaoDeToque.x – 64 / 2;
```

A classe Vector3 encapsula um vetor 3D (a coordenada z foi zerada). Gdx.input.getX(), Gdx.input.getY() retornam a posição do mouse/toque. É necessário transformar o sistema de coordenadas para o sistema usado no mundo do jogo (camera.unproject(posicaoDeToque)) e logo usá-lo para posicionar a coordenada x do balde.

Movimentando o balde (teclado)

Queremos movimentar o balde sem aceleração a 200 pixels/unidades por segundo para a direita ou para a esquerda. Para implementar esse movimento baseado no tempo é preciso saber o tempo que se passou entre o último frame renderizado e o atual:

```
if(Gdx.input.isKeyPressed(Keys.LEFT)) balde.x -= 200 * Gdx.graphics.getDeltaTime();
if(Gdx.input.isKeyPressed(Keys.RIGHT)) balde.x += 200 * Gdx.graphics.getDeltaTime();

// garante que o balde ficará dentro das fronteiras da tela
if(balde.x < 0) balde.x = 0;
if(balde.x > 800 - 64) balde.x = 800 – 64;
```

Adicionando pingos de chuva

Criamos uma lista de instâncias de retângulos:

```
pingosDeChuva = new Array<Rectangle>();
```

A classe Array é um utilitário do LibGDX que pode ser usado no lugar da coleção de Java padronizada ArrayList. A vantagem do seu uso é que ela minimiza a produção de lixo tanto quanto for possível.

Também precisamos acompanhar a última vez que um pingo foi gerado (em nanossegundos por isso o tipo long):

```
private long instanteUltimaGota;
```

Para criar os pingos criamos um método separado gerarGota():

```
   private void gerarGota() {
       Rectangle Gota = new Rectangle();
       Gota.x = MathUtils.random(0, 800-64);
       Gota.y = 480;
       Gota.width = 64;
       Gota.height = 64;
       pingosDeChuva.add(Gota);
       instanteUltimaGota = TimeUtils.nanoTime();
   }
```

Esse método instancia um novo objeto Rectangle Gota, lhe atribui uma posição aleatória no topo da tela o adiciona ao array pingosDeChuva. TimeUtils é uma classe do libGDX que oferece métodos estáticos básicos de tempo. Neste caso retorna o tempo em nanosegundos que é guardado para que possamos decidir se será gerado ou não um novo pingo (gota).

No método create() o array é instanciado:

```
       // criar um array de pingosDeChuva e gerar a primeira Gota
       pingosDeChuva = new Array<Rectangle>();
       gerarGota();
```

No método render() verificamos se, pelo tempo que passou, já é hora de gerar uma nova gota e, se for o caso, geramos uma nova gota:

```
       // verificamos se se precisamos gerar uma nova Gota
       if(TimeUtils.nanoTime() - instanteUltimaGota > 1000000000) gerarGota();
```

Queremos fazer que as gotas se movam para baixo a 200 pixels/unidades por segundo e que sejam removidas do array assim que chegarem à borda inferior da tela ou que toquem no balde e neste último caso que também seja tocado um efeito sonoro (o método Rectangle.overlaps() se o retângulo se sobrepõe a outro triângulo):

```
     // mover os pingosDeChuva, remover os que chegarem à borda inferior da tela
     // ou que toquem no balde. Neste último caso que também
     // seja tocado um efeito sonoro
       for (Iterator<Rectangle> iter = pingosDeChuva.iterator(); iter.hasNext();) {
         Rectangle pingo = iter.next();
         pingo.y -= 200 * Gdx.graphics.getDeltaTime();
         if(pingo.y + 64 < 0) iter.remove();
          if(pingo.overlaps(balde)) {
             gotaSom.play();
             iter.remove();
          }
       }
```

As gotas precisam ser também renderizadas. Incluí-las no processo do

“SpriteBatch”:

```
      // inicie um batch (lote) e desenhe o balde
      // e todas as gotas
       batch.begin();
       batch.draw(baldeImagem, balde.x, balde.y);
       for(Rectangle pingo: pingosDeChuva) {
         batch.draw(gotaImagem, pingo.x, pingo.y);
       }
       batch.end();
```

Limpeza

```
  @Override
  public void dispose() {
       // descarte de todos recursos nativos
       gotaImagem.dispose();
       baldeImagem.dispose();
       gotaSom.dispose();
       chuvaMusica.dispose();
       batch.dispose();
   }
```

Atividade ILJ003-Framework03 - LibGDX Jogo Básico Parte 2

Comente os recursos disponíveis no LibGDX (procure no Google ou no Duckdukgo os métodos que foram usados) e as técnicas de programação utilizadas para se obter as ações que foram abordados na segunda parte do jogo de exemplo “Gota”.