libGDX

https://github.com/libgdx/libgdx/wiki/

Rafael Vieira Coelho

rafael.coelho@farroupilha.ifrs.edu.br

Parte 6 - Jogo Completo

- Versão 1 - Construção do Projeto
- Versão 2 - Desenha icicle
- **Versão 3 - Desenha Jogador**
- Versão 4 - Controle de Teclado (setas)
- Versão 5 - Adiciona Icicles
- Versão 6 - Remove Icicles que somem da tela
- Versão 7 - Detecta Colisão
- Versão 8 - Adiciona o HUD
- Versão 9 - Adiciona níveis de dificuldade
- Versão 10 - Adiciona seleção de nível de dificuldade

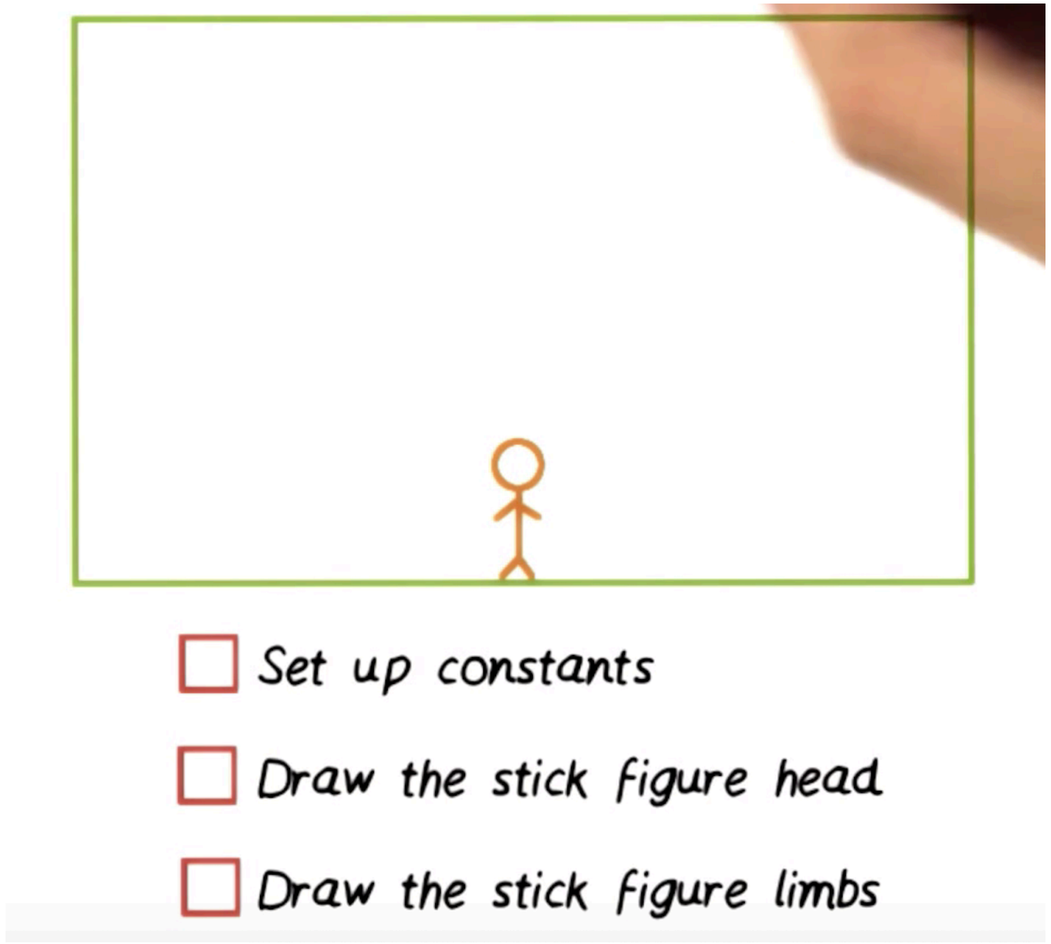
https://github.com/libgdx/libgdx/wiki/

Como Podemos Desenhar o Jogador?

1. Definimos as características na classe de constantes

2. Desenhamos a cabeça do jogador

3. Desenhados os membros do jogador (corpo)



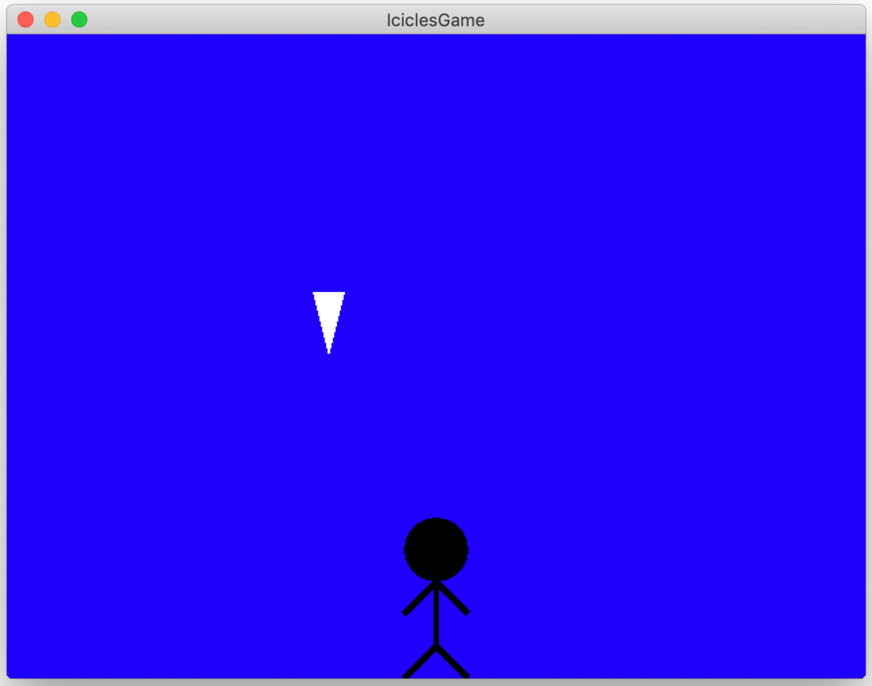
Classe Constants.java

- Definimos as características do jogador (constantes iniciadas por PLAYER_...)
- Raio da cabeça
- Altura da cabeça
- Largura do corpo
- Cor do jogador

Constants.java

```
1   package com.udacity.gamedev.icicles;
2
3   import com.badlogic.gdx.graphics.Color;
4
5
6   public class Constants {
7       public static final float WORLD_SIZE = 10.0f;
8       public static final Color BACKGROUND_COLOR = Color.BLUE;
9
10      // TODO: Add constant for player head radius
11      public static final float PLAYER_HEAD_RADIUS = 0.5f;
12      // TODO: Add constant for player head height
13      public static final float PLAYER_HEAD_HEIGHT = 4.0f * PLAYER_HEAD_RADIUS;
14      // TODO: Add constant for player limb width
15      public static final float PLAYER_LIMB_WIDTH = 0.1f;
16      // TODO: Add constant for circle segments for the player's head
17      public static final int PLAYER_HEAD_SEGMENTS = 20;
18      // TODO: Add constant for the player's color
19      public static final Color PLAYER_COLOR = Color.BLACK;
20
21      public static final float ICICLES_HEIGHT = 1.0f;
22      public static final float ICICLES_WIDTH = 0.5f;
23      public static final Color ICICLE_COLOR = Color.WHITE;
24
25  }
```

Classe Player.java

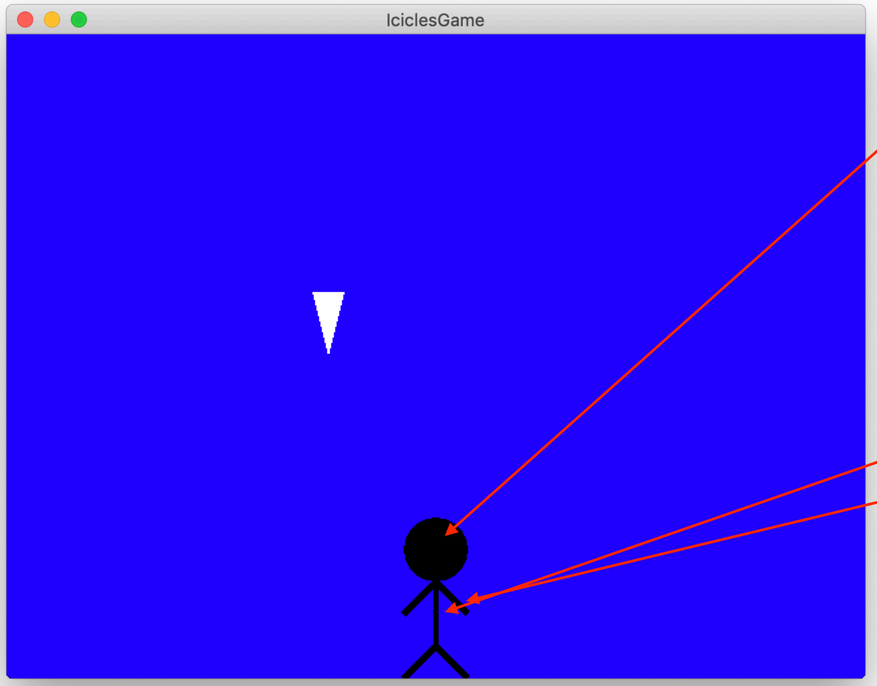


Player.java

```
2
3   import com.badlogic.gdx.graphics.glutils.ShapeRenderer;
4   import com.badlogic.gdx.graphics.glutils.ShapeRenderer.ShapeType;
5   import com.badlogic.gdx.math.Vector2;
6   import com.badlogic.gdx.utils.viewport.Viewport;
7
8
9   public class Player {
10
11      public static final String TAG = Player.class.getName();
12
13      // TODO: Add a position (add constants to Constants.java first)
14      Vector2 position;
15      // TODO: Add a viewport
16      Viewport viewport;
17
18      // TODO: Add constructor that accepts and sets the viewport, then calls init()
19      public Player(Viewport viewport) {
20          this.viewport = viewport;
21          init();
22      }
23
24      // TODO: Add init() function that moves the character to the bottom center
25      //  of the screen
26      public void init() {
27          position = new Vector2(
28                  x: viewport.getWorldWidth() / 2,
29                  Constants.PLAYER_HEAD_HEIGHT);
30      }
```

Classe Player.java

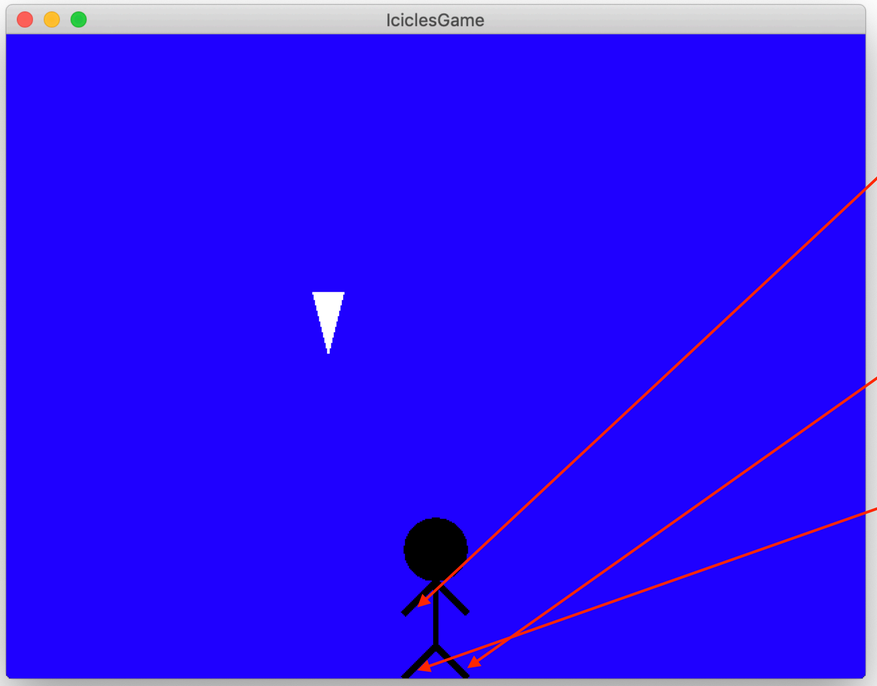
- No método render(), desenhamos o jogador
- O método rectLine() desenha uma linha e o circle() desenha um círculo.



Player.java

```
32      // TODO: Create a render function that accepts a ShapeRenderer and
33      //  does the actual drawing
34  @   public void render(ShapeRenderer renderer) {
35
36          renderer.setColor(Constants.PLAYER_COLOR);
37          renderer.set(ShapeType.Filled);
38          renderer.circle(
39                  position.x,
40                  position.y,
41                  Constants.PLAYER_HEAD_RADIUS,
42                  Constants.PLAYER_HEAD_SEGMENTS);
43
44          Vector2 torsoTop = new Vector2(
45                  position.x,
46                  y: position.y - Constants.PLAYER_HEAD_RADIUS);
47          Vector2 torsoBottom = new Vector2(torsoTop.x,
48                  y: torsoTop.y - 2 * Constants.PLAYER_HEAD_RADIUS);
49
50          renderer.rectLine(torsoTop, torsoBottom, Constants.PLAYER_LIMB_WIDTH);
51
52          renderer.rectLine(
53                  torsoTop.x, torsoTop.y,
54                  x2: torsoTop.x + Constants.PLAYER_HEAD_RADIUS,
55                  y2: torsoTop.y - Constants.PLAYER_HEAD_RADIUS,
56                  Constants.PLAYER_LIMB_WIDTH);
57
```

Classe Player.java



Player.java

```
54
55
56
57
58          renderer.rectLine(
59                  torsoTop.x, torsoTop.y,
60                  x2: torsoTop.x - Constants.PLAYER_HEAD_RADIUS,
61                  y2: torsoTop.y - Constants.PLAYER_HEAD_RADIUS,
62                  Constants.PLAYER_LIMB_WIDTH);
63
64          renderer.rectLine(
65                  torsoBottom.x, torsoBottom.y,
66                  x2: torsoBottom.x + Constants.PLAYER_HEAD_RADIUS,
67                  y2: torsoBottom.y - Constants.PLAYER_HEAD_RADIUS,
68                  Constants.PLAYER_LIMB_WIDTH);
69
70          renderer.rectLine(
71                  torsoBottom.x, torsoBottom.y,
72                  x2: torsoBottom.x - Constants.PLAYER_HEAD_RADIUS,
73                  y2: torsoBottom.y - Constants.PLAYER_HEAD_RADIUS,
74                  Constants.PLAYER_LIMB_WIDTH);
75      }
76  }
77
```

Classe ICiclesScreen.java

- Precisamos criar o objeto que representa o jogador.

IciclesScreen.java

```
12
13  public class IciclesScreen implements Screen {
14
15      public static final String TAG = IciclesScreen.class.getName();
16
17      ExtendViewport iciclesViewport;
18      ShapeRenderer renderer;
19
20      Player player;
21      Icicle icicle;
22
23      @Override
24      public void show() {
25          iciclesViewport = new ExtendViewport(Constants.WORLD_SIZE, Constants.
26
27          renderer = new ShapeRenderer();
28          renderer.setAutoShapeType(true);
29
30          player = new Player(iciclesViewport);
31          icicle = new Icicle(new Vector2( x: Constants.WORLD_SIZE / 2,  y: Consta
32
33      }
34
35      @Override
36      public void resize(int width, int height) {
37          iciclesViewport.update(width, height,  centerCamera: true);
38          player.init();
39      }
```

Classe ICiclesScreen.java

- E atualizar sua posição na tela

IciclesScreen.java

```
46
47      @Override
48      public void render(float delta) {
49          // TODO: Call update() on player
50          player.update(delta);
51
52          iciclesViewport.apply( centerCamera: true);
53          Gdx.gl.glClearColor(Constants.BACKGROUND_COLOR.r, Constants.BACKGROUN
54          Gdx.gl.glClear(GL20.GL_COLOR_BUFFER_BIT);
55
56          renderer.setProjectionMatrix(iciclesViewport.getCamera().combined);
57          renderer.begin(ShapeType.Filled);
58          renderer.setColor(Constants.ICICLE_COLOR);
59          icicle.render(renderer);
60          player.render(renderer);
61          renderer.end();
62
63      }
```