Entrega 1 Relatório - IA941

Lucas Gaspar

13 de Março de 2020

1 Introdução

Na primeira aula dia 06/03 nos foi apresentado o conteúdo da disciplina, bem como os métodos de avaliação. Logo após a introdução da matéria, foi nos dado a primeira atividade bem como a atividade para se fazer em casa com data de entrega na sexta feira dia 13/03.

2 Primeira Atividade

A primeira atividade se constituia de alguns passos para entender melhor o que é o *WS3D* e o que é possivel fazer com ele. Os passos eram:

1. Importar o *WordServer3D (WS3D)*
2. Explorar a aplicação para ver do que ela é capaz
3. Importar o projeto *WordServer3DProxy (WS3DProxy)*
4. Criar um projeto para manipular o *WS3D* através do *WS3DProxy*

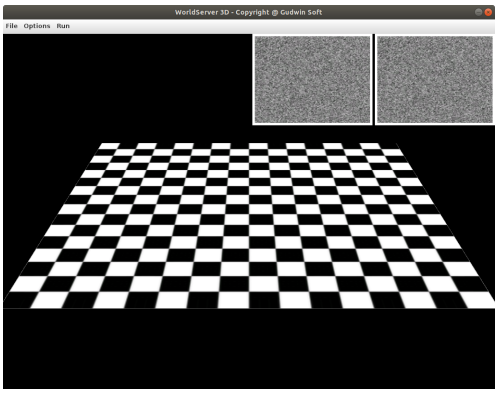


Figure 1: WS3D - WorldServer3D

Durnte sua execução, foi possível observar que o *WS3D* possui comandos em sua própria interface que permitem a criação de objetos como criaturas, alimentos perecíveis e não perecíveis, jóias, paredes e outros objetos. O WS3DProxy por sua vez é um projeto que provê uma camada intermediária entre um projeto *Java* e o *WS3D*, tornando possivel que o projeto criado interaja com o *WS3D* através de comandos enviados ao *WS3DProxy*.

Para que seja possível utilizar o *WS3DProxy* em um outro projeto, é necessário rodar o comando abaixo para efetuar a *build* do *WS3DProxy* gerando assim um arquivo jar que será usado no projeto gerado.

```
./gradlew clean build
```

Após esse comando, o *jar* da aplicação será gerado dentro de um arquivo *tar* no diretório `gradle/distribution/ws3d-proxy-0.0.1.tar`. Após esse passo é necessário pegar os jars gerados e copiá-los para um diretório criado dentro da pasta raiz do projeto. Feito isso, basta adicionar os seguintes trechos de código no arquivo `build.gradle` :

```
repositories {
    mavenCentral()
    flatDir {
        dirs 'lib'
    }
}

dependencies {
    implementation fileTree(dir: "lib", includes: ['*.jar'])
    testImplementation      'junit:junit:4.13'
}
```

Após preparar o projeto para utilizar o *WS3DProxy*, uma classe foi criada contendo o seguinte código:

```
import ws3dproxy.CommandExecException;
import ws3dproxy.WS3DProxy;
import ws3dproxy.model.Creature;
import ws3dproxy.model.World;
import ws3dproxy.model.WorldPoint;

public class Main {

    public static void main(String[] args) {
        WS3DProxy proxy = new WS3DProxy();
```

```
        try {
            World w = World.getInstance();
            w.reset();
            World.createFood(0, 350, 75);
            World.createFood(0, 100, 220);
            World.createFood(0, 250, 210);
            Creature c = proxy.createCreature(100,450,0);
            c.start();
            WorldPoint position = c.getPosition();
            double pitch = c.getPitch();
            double fuel = c.getFuel();
            c.moveto(10, 0, 0);
        } catch (CommandExecException e) {
            System.out.println("Erro capturado");
        }
    }
}
```

Para ver o projeto interagir com o WS3D, é necessário primerio executar o WS3D e depois o projeto criado. Como resultado desta classe tivemos um mundo virtual com três alimentos perecíveis e uma criatura que se move até certo ponto.

3 Segunda Atividade

A próxima atividade passada em aula, deve ser entregue até dia 13/03 e consiste de uma aplicação *Java* para controlar o movimento através do *WS3DProxy* de uma criatura no *WS3D*. Os ítens a serem entregues são:

1. Código do controlador manual.
2. Arquivo *jar* do *WS3D* compilado.
3. Arquivo *jar* do contolador.
4. *Script* em bash para rodar o *WS3D* e o controlador
5. Relatório das atividades em *PDF*.

A entrega deve ser feita com um *link* do *Google Drive* contendo um arquivo *zip* com os itens acima.

3.1 Controlador manual

Após ser gerado um projeto Java contendo o WS3DProxy, deu-se inicio a criação da tela principal, que contém os botões que irão inciar as outras telas. As ações possíveis na tela principal são:

- *Creature* - inicia a tela para criar e movimentar criaturas.
- *Food* - inicia a tela responsável por instanciar alimentos.
- *Jewel* - inicia a tela que efetua a criação de jóias.
- *Wall* - inicia a tela que cria paredes.
- *Reset* - botão responsável por reiniciar o mundo no WS3D.
- *Exit* - botão que sai da aplicação.

Figure 2: Tela Principal

3.2 Tela de Criatura

Nesta tela é possível instanciar uma criatura em uma posição *(X;Y)* , com uma rotação θ em graus *(pitch)* e com a cor desejada pelo usuário. A classe utilizara da função abaixo para criar a criatura:

```
this.proxy.createCreature(xPosition, yPosition, pitch, color);
```

O *combo box* do lado direito da tela, perimite que o usuário selecione a criatura que deseja controlar fazendo o uso dos botões abaixo da caixa de seleção. A criatura selecionada irá se mover enquanto o botão estiver pressionado, e irá parar de se mover quando o botão for solto.

A movimentação e a rotação da criatura é feita através dos comandos:

```
creature.start(); // inicia o "motor" da criatura
creature.move(VelovidadeR, VelocidadeL, Pitch); // move a criatura
//aplicando as velocidades em suas rodas direitas e esquerdas
creature.rotate(velocidadeAngular); // rotaciona a criatura
creature.stop(); //remove o "motor" da criatura
```

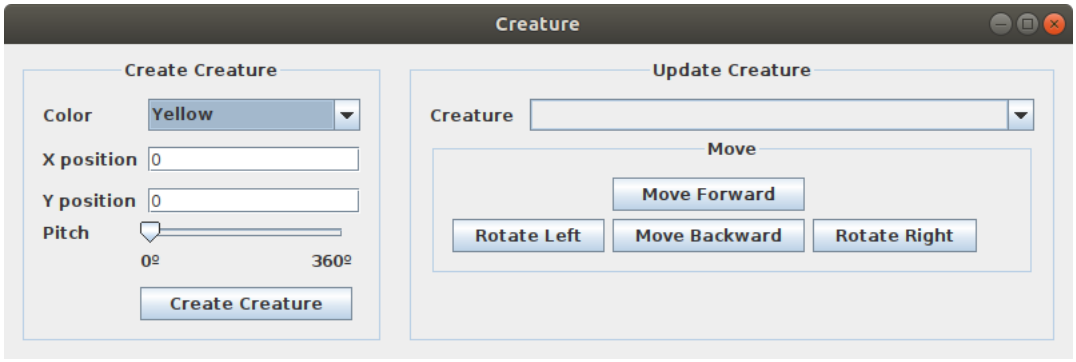
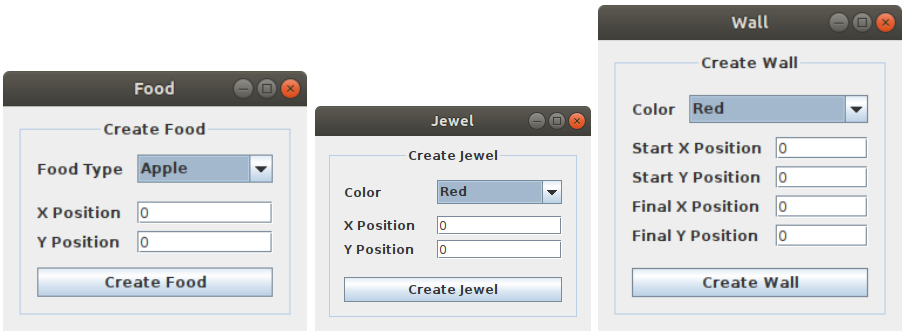


Figure 3: Tela Criatura

3.3 Tela de Comida, Jóia e Parede

Todas as telas possuem o mesmo comportamento, passadas algumas informações de posição *(X;Y)* e cor, o controlador instância o objeto desejado. As criações são feitas pelos comandos:

```
World.createFood(foodType, xPosition, yPosition); // Instancia um alimento
World.createJewel(color, xPosition, yPosition); // Instancia uma joia
World.createBrick(color, startXPosition, startYPosition, finalXPosition,
finalYPosition); // Intancia uma parede
```



(a) Tela Comida (b) Tela Jóia (c) Tela Parede

Figure 4: Talas de Comida, Jóia e Parede

4 Resultados

Como resultado foi obtido um *software Java* capaz de criar e controloar objetos no WS3D atráves do WS3DProxy de interfaces gráficas *Swing*.

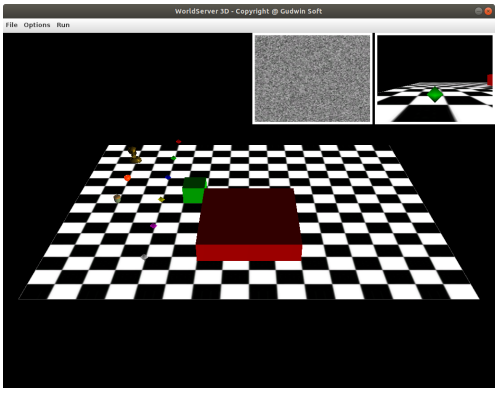


Figure 5: Tela com objetos criados através do controlador

O projeto encontra-se disponível no repositório do GitHub.