Ministério da Educação
Secretaria de Educação Profissional e Tecnológica
Instituto Federal Catarinense
Campus Rio do Sul – Unidade Urbana

VINICIUS MARZALL LIPPEL

TRABALHO FINAL POO:
GERADOR DE GRÁFICOS DA REFLEXÃO DA LUZ

Rio do Sul
2019

SUMÁRIO

1. INTRODUÇÃO

Óptica é o ramo da física que estuda os fenômenos relacionados a luz visível e outros espectros de ondas eletromagnéticas. Um desses fenômenos é a reflexão, quando a luz volta a se propagar no meio de origem após incidir em um objeto ou superfície. Inicialmente o objetivo deste trabalho era o de criar um software que fosse capaz de gerar gráficos da reflexão da luz em espelhos, mostrando seu ângulo de incidência e de saída, entretanto esta ideia sofreu uma alteração a partir de uma sugestão do professor. O objetivo do trabalho passou a ser mostrar a luz refletindo dentro de um objeto com duas dimensões, sendo inicialmente um quadrado.

2. DESENVOLVIMENTO

2.1. Planejamento

O desenvolvimento deste sistema começou com seu planejamento, para que este pudesse ser bem executado. Decidiu-se pela aplicação do conceito de matrizes para a montagem do gráfico, de forma que esta conteria a forma do objeto e a posição da luz dentro de si, onde haveria um caractere para representar as paredes do objeto, um para representar a luz e outro para representar o espaço vazio. Foi também criado um protótipo de tela para uma melhor visualização de como ficaria a parte visual do software após sua conclusão, estes foram criados no website Wireframe.cc. Além disso ainda foi desenvolvido um diagrama de classes que permitiu verificar como ficaria a relação entre as classes no sistema. Uma estratégia de persistência também foi pensada para o sistema, para que os dados necessários pudessem ser guardados de forma adequada e usados posteriormente para o upload de um gráfico no sistema.

2.2. Desenvolvimento

Este sistema foi desenvolvido em Java, utilizando o paradigma de orientação a objetos, de forma a aplicar o que foi aprendido nas aulas. O front-end foi criado a partir de uma ferramenta chamada Scene Builder, que permite, de forma visual, criar páginas XML, tornando este trabalho muito mais simples e rápido. As IDEs usadas para a edição dos códigos foram o Netbeans (que permite criar um projeto integrado a uma página XML) e o Eclipse.

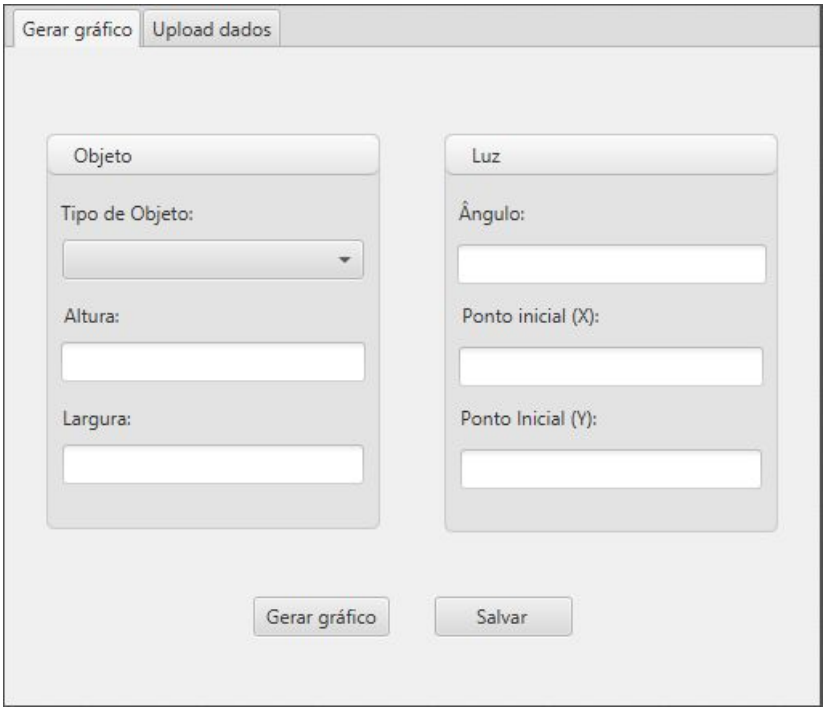
Como citado anteriormente na introdução, o desenvolvimento do software sofreu diversas alterações no seu decorrer, alterando o objetivo principal a ser atendido por ele. Inicialmente, questões relacionadas ao planejamento do software tiveram que ser alteradas e, portanto, algumas coisas presentes nelas não estão presentes no software final. A principal alteração ocorreu na parte visual do sistema, onde esta ficou claramente diferente daquela planejada através do wireframe, isto ocorreu pois as informações que necessitavam ser passadas pelo usuário foram alteradas.

2.3. Funcionamento

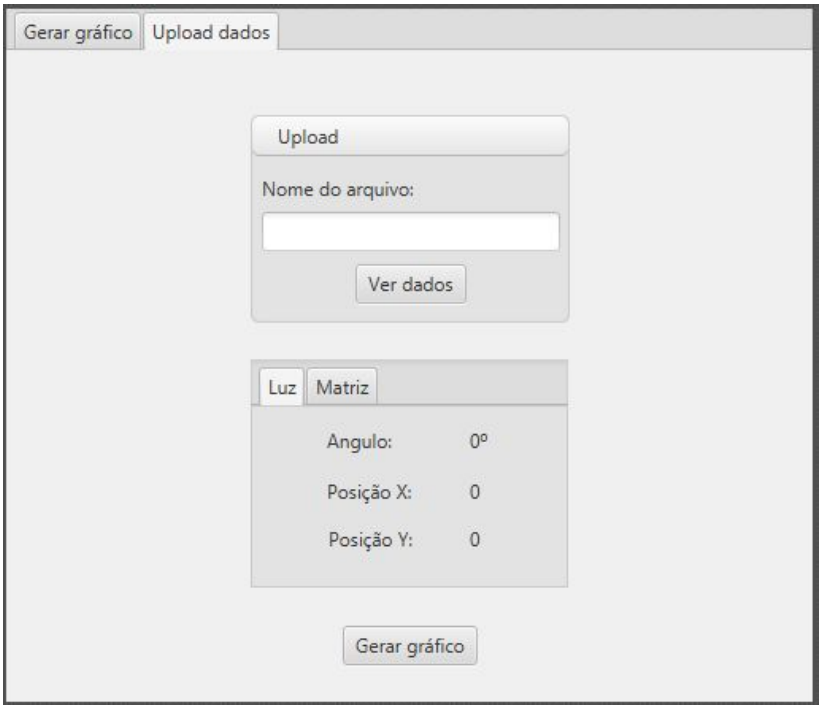
Inicialmente, uma tela irá aparecer para o usuário onde, no canto superior esquerdo, existem duas opções de tela para inserir os dados de um gráfico, uma inserindo-os manualmente e outra para fazer um upload de um arquivo Json.

Na primeira temos os campos para o usuário digitar, em relação ao objeto, o seu tipo, altura e largura, e em relação à luz sua angulação e ponto X e Y da saída. Após os dados estarem digitados, o usuário tem a opção de gerar um gráfico ou de salvar estes dados. Ao clicar em “Gerar gráfico”, uma matriz aparecerá no terminal da IDE mostrando o caminho que a luz seguirá até ela bater em si mesma. Se este clicar em “Salvar”, os dados inseridos nos campos serão adicionados a um arquivo Json em uma ordem específica

predeterminada.



Caso o usuário decida ir para a página de upload, aparecerá um campo para que este digite o nome do arquivo. A partir do arquivo selecionado, o programa pega os dados do Json e os mostra abaixo, dando também a opção para que seja gerado um gráfico com estes dados.



3. IMAGENS CÓDIGO

```
39    public void montarGrafico(){
40        grafico = new String[matriz.getLargura()][matriz.getAltura()];
41        if(verificarLuz()) {
42            for(int i=0; i<matriz.getLargura(); i++) {
43                grafico[i][0] = "X";
44                grafico[i][matriz.getAltura()-1] = "X";
45            }
46            for(int i=0; i<matriz.getAltura(); i++) {
47                grafico[0][i] = "X";
48                grafico[matriz.getLargura()-1][i] = "X";
49            }
50            for(int i=1; i<matriz.getLargura()-1; i++) {
51                for(int x=1; x<matriz.getAltura()-1; x++) {
52                    grafico[i][x] = " ";
53                }
54            }
55
56            grafico[luz.getX()][luz.getY()] = "#";
57        }
58    }
59
60    public Paredes olharVolta() {
61        if((luz.getX()-1 == 0 && luz.getY()-1 == 0) || (luz.getX()-1 == 0 && luz.getY()+1 == matriz.getAltura()) ||
62                (luz.getX()+1 == matriz.getLargura() && luz.getY()-1 == 0) || (luz.getX()+1 == matriz.getLargura() && luz.getY()+1 == matriz.getAltura()))
63            return Paredes.CANTO;
64
65        if(luz.getX()-1 == 0)
66            if(luz.getAngulo() == Angulos.A135 || luz.getAngulo() == Angulos.A225)
67                return Paredes.ESQUERDA;
68        if(luz.getX()+1 == matriz.getLargura()-1)
69            if(luz.getAngulo() == Angulos.A45 || luz.getAngulo() == Angulos.A315)
70                return Paredes.DIREITA;
71        if(luz.getY()-1 == 0)
72            if(luz.getAngulo() == Angulos.A45 || luz.getAngulo() == Angulos.A135)
73                return Paredes.CIMA;
74        if(luz.getY()+1 == matriz.getAltura()-1)
75            if(luz.getAngulo() == Angulos.A225 || luz.getAngulo() == Angulos.A315)
76                return Paredes.BAIXO;
77        return null;
78    }
```

```
79
80    public void andarLuz() {
81        if(olharVolta() == null) {
82            if(luz.getAngulo() == Angulos.A45) {
83                luz.setX(luz.getX()+1);
84                luz.setY(luz.getY()-1);
85            }
86            else if(luz.getAngulo() == Angulos.A135) {
87                luz.setX(luz.getX()-1);
88                luz.setY(luz.getY()-1);
89            }
90            else if(luz.getAngulo() == Angulos.A225) {
91                luz.setX(luz.getX()-1);
92                luz.setY(luz.getY()+1);
93            }
94            else if(luz.getAngulo() == Angulos.A315) {
95                luz.setX(luz.getX()+1);
96                luz.setY(luz.getY()+1);
97            }
98            grafico[luz.getX()][luz.getY()] = "#";
99        }
```

```
else {
    if(olharVolta() == Paredes.ESQUERDA) {
        if (luz.getAngulo() == Angulos.A135)
            luz.setAngulo(Angulos.A45);
        else if(luz.getAngulo() == Angulos.A225)
            luz.setAngulo(Angulos.A315);
    }
    else if(olharVolta() == Paredes.DIREITA) {
        if (luz.getAngulo() == Angulos.A45)
            luz.setAngulo(Angulos.A135);
        else if(luz.getAngulo() == Angulos.A315)
            luz.setAngulo(Angulos.A225);
    }
    else if(olharVolta() == Paredes.CIMA) {
        if (luz.getAngulo() == Angulos.A45) {
            luz.setAngulo(Angulos.A315);
        }
        else if(luz.getAngulo() == Angulos.A135)
            luz.setAngulo(Angulos.A225);
    }
    else if(olharVolta() == Paredes.BAIXO) {
        if (luz.getAngulo() == Angulos.A315)
            luz.setAngulo(Angulos.A45);
        else if(luz.getAngulo() == Angulos.A225)
            luz.setAngulo(Angulos.A135);
    }
    else if(olharVolta() == Paredes.CANTO) {
        System.exit(0);
    }
}
```

```
1  package trabalhofinalpoo;
2
3  public enum Paredes {
4      ESQUERDA, DIREITA, CIMA, BAIXO, CANTO;
5  }
6
```

```
129
130     @FXML
131     public void salvar(ActionEvent event) throws IOException{
132         Luz l = new Luz();
133         l.setAngulo(Angulos.A45);
134         l.setX(Integer.parseInt(posx.getText()));
135         l.setY(Integer.parseInt(posy.getText()));
136
137         Matriz m = new Matriz();
138         m.setLargura(Integer.parseInt(largura.getText()));
139         m.setAltura(Integer.parseInt(altura.getText()));
140         Json json = new Json();
141     json.gravar(l, m, "teste");
142     }
143
144     @FXML
145     public void verDados(ActionEvent event){
146         Json json = new Json();
147         Grafico g = new Grafico();
148         Persistencia per = new Persistencia (json);
149         g = per.ler(nomeArquivo.getText());
150         anguloUp.setText(Integer.toString(g.getLuz().anguloToInt()));
151         posXUp.setText(Integer.toString(g.getLuz().getX()));
152         posYUp.setText(Integer.toString(g.getLuz().getY()));
153 //      tipoUp.setText(g.getMatriz().);
154         alturaUp.setText(Integer.toString(g.getMatriz().getAltura()));
155         alturaUp.setText(Integer.toString(g.getMatriz().getLargura()));
156     }
157
158     @FXML
159     public void gerarGraficoUpload(ActionEvent event) throws InterruptedException {
160         Luz l = new Luz();
161         l.setAngulo(Angulos.A45);
162         l.setX(Integer.parseInt(posXUp.getText()));
163         l.setY(Integer.parseInt(posYUp.getText()));
164
165         Matriz m = new Matriz();
166         m.setLargura(Integer.parseInt(larguraUp.getText()));
167         m.setAltura(Integer.parseInt(alturaUp.getText()));
168
169         Grafico g = new Grafico();
170         g.setLuz(l);
171         g.setMatriz(m);
172         g.montarGrafico();
173
174         mostraMatriz(g.getGrafico());
175
176         boolean repeat = true;
177         int inicialX = l.getX();
178         int inicialY = l.getY();
179         while(repeat){
180             g.andarLuz();
181             mostraMatriz(g.getGrafico());
182             if(g.getLuz().getX() == inicialX && g.getLuz().getY() == inicialY)
183                 repeat = false;
184             Thread.sleep(100);
185         }
186     }
187
```

4. CONCLUSÃO

Podemos concluir, a partir daquilo que foi desenvolvido, que o software ainda não está completamente finalizado. Entre algumas funcionalidades que ainda poderiam ser implementadas estão a criação e utilização de diferentes objetos para a luz ser refletida, uma representação mais visual do gráfico, pois este aparece apenas no terminal da IDE, e a possibilidade de utilizar mais ângulos para a luz. Entretanto Uma boa parte do sistema foi desenvolvida com êxito, possuindo um bom funcionamento e uma integração com a parte visual do XML.