Relatório do Trabalho Final INE5404
Grid de RPG Interativo

Gustavo Kundlatsch Lucas Godoy Teo Gallarza

Universidade Federal de Santa Catarina,
Brasil
28 de Novembro de 2017

1 Análise do Problema

Role Playing Games (RPG) são jogos constituídos por um mestre e seus jogadores, onde o primeiro narra uma história e o segundo atua, como em um teatro de mesa. Para isso, utiliza-se um sistema de jogo, que dita as regras que precisam ser seguidas durante a partida. Nosso programa tem como objetivo fornecer uma ferramenta simples e poderosa que agrega diversos módulos em um lugar só, para resolver a dificuldade de encontrar boas ferramentas gratuitas que funcionem offline e tenham todas as características que nosso projeto engloba.

2 Uso do Programa

O programa é dividido em seis módulos:

- Banco de Fichas, onde o usuário pode criar fichas, editar o código html delas e salvar em um txt, para que ter acesso mais tarde.
- Rolagem de Dados, em que o mestre pode escolher a quantidade e tipo de dados que vai fazer uma rolagem e o programa retorna números aleatórios correspondentes;
- Bloco de Notas, que é apenas uma ferramenta de interface gráfica para o usuário inserir qualquer texto, que não será salvo pelo programa;
- Biblioteca, que também é a versão com persistência do bloco de notas (você pode escrever qualquer coisa, atribuir um título e salvar para usar mais tarde).
- Mesa de Som, onde o usuário escolhe um dos botões para reproduzir seu som correspondente. Ao clicar no último botão, é possível escolher o próprio arquivo .wav para ser reproduzido;
- o Grid, que é um plano quadriculado onde podem ser adicionados *tokens* (imagens png ou jpg fornecidas pelo usuário) que representam tanto os jogadores da partida quanto monstros e NPCS.

Além dos módulos, ainda é possível alterar a cor de fundo, para realizar ambientações de acordo com a situação do jogo.

Nosso programa foi desenvolvido para ser utilizado em duas telas, sendo uma o computador do mestre, que pode ser acessado a qualquer momento de uma sessão de jogo, e a tela do grid, que pode ser um projetor, uma televisão ou qualquer outra tela secundária que todos os outros jogadores podem vizualizar completamente.

2.1 Exemplo de uso

Vamos propor uma situação de jogo onde o mestre está narrando uma história para outros três jogadores. Em dado momento, o grupo de jogadores (aventureiros) entram em combate com uma gangue de goblins. Nesse instante o mestre insere no Grid os *tokens* dos jogadores (um guerreiro, um arqueiro e um mago) e de três goblins. Então abre a quatro guias de fichas: uma para cada personagem e uma para a ficha dos monstros. O mestre também abre o módulo de rolagem de dados e de sons, para ambientar. Para concluir o arranjo do software, ele troca a cor do fundo para verda, para combinar com os inimigos.

O mestre então usa a Mesa de Som, apertando o ícone de monstro, que retorna um grunhido feral. Em seguida, ele vê o Ataque corpo-a-corpo dos goblins, e utiliza a rolagem de dados para realizar o teste para tentar acertar o guerreiro. Utilizando um dado de vinte faces (1d20), consegue um acerto, e calcula o dano. Em seguida, na ficha do guerreiro, o mestre pode clicar para editar, se abrirá uma tela com o html da ficha e ele pode alterar o valor de vida.

Esse exemplo simples mostra o quão prático é montar as ferramentas de acordo com a demanda do momento da sessão de RPG. A ideia é que o software seja bastante intuitivo e agradável, atraindo novos usuários.

3 Arquitetura e Projeto

O projeto é dividido em dias grandes partes: o Grid e a Interface do Mestre. O grid é a parte visível para os usuários, e como já foi explicado, não precisa de persistência de dados, os ícones entrados são temporários, e a aparência de painel quadriculado vem de uma matriz de botões com borda cuja aparência é transparente. Assim, ficam apenas as linhas, tendo um grid clicável. Para adicionar um novo *token* deve-se clicar em algum quadrado com o botão direito do mouse, abrindo um JPopupMenu, que contém a opção de Adicionar Ícone. Ao clicar nessa opção, o usuário deve escolher uma imagem no formato JPG ou PNG.

A parte da Interface do Mestre é um JDesktopPane, que chama cada um dos outros panes (cada um uma classe própria, com exceção do bloco de notas) que é chamado de acordo com sua necessidade. Cada um dos módulos possui sua própria classe de controle (similar a um modelo MVC), sendo que a Biblioteca e o Banco de Fichas utilizam as classes FileWriter e GenericFileReader para realizar as operações de persistência, e o banco de fichas foi modelado a partir da interface Sheet, que é implementada por cada tipo de ficha (NPCSheetManager, MonsterSheetManager e CharacterSheetManager).

4 Comportamento e Resultados

A integração entre os módulos da aplicação já foi descrita no item anterior, e esse design permite que o programa funcione de maneira bastante independente. Os principais instrumentos utilizados que não vimos em aula foi a classe FileChooser e as classes de som do javax, ambas essenciais para que nosso projeto atingisse seu objetivo. Abaixo, estão explicados esses trechos de código:

```
try
    {
    AudioInputStream audioInputStream = AudioSystem.getAudioInputStream
      (new File(soundName).getAbsoluteFile( ));
    Clip clip = AudioSystem.getClip( );
    clip.open(audioInputStream);
    clip.start( );
    }
    catch(UnsupportedAudioFileException e)
    {
    JOptionPane.showMessageDialog(null,
        "Formato invalido, escolha um arquivo .wav");
    }
    catch(Exception ex)
    {
      System.out.println("Error with playing sound.");
      ex.printStackTrace( );
    }
```

O código acima faz a reprodução do som. Ele está dentro do construtor da classe Soundboard, que recebe uma string como parametro (soundName), e será utilizado pela classe SoundBoardPanel nos *listeners* de cada um dos botões desse módulo.

```
class PopupActionListener implements ActionListener{
    public void actionPerformed(ActionEvent e){
            if(e.getSource()==items[0]){
                    JFileChooser chooser = new JFileChooser();
                    chooser.setFileSelectionMode(JFileChooser.FILES_ONLY);
                }
        }
}
```

Esse segundo código é o *listener* da opção "Adicionar Ícone"do popup menu que aparece quando algum dos botões do grid é clicado. A classe JFileChooser é utilizada para pegar um arquivo, que posteriormente será ícone do button correspondente.

Considerações finais

Nosso programa entrega todas as funcionalidades que planejamos, exceto a possibilidade de desenhar sobre o grid quadriculado, para que o mestre faça marcações e desenhos para simular o mapa. Apesar disso, o funcionamento do restante do programa está bastante

claro, intuitivo e reutilizável. Entretanto, o código foi mal escalonado, pois em etapas finais do código já havia muitas linhas nas classes principais, o que em certos momentos tornava a implementação de outras funcionalidades um tanto quanto confuso. Mesmo assim, a decisão de fazer o código em módulos independentes permitiu que trabalhássemos em partes pequenas, que no final se juntaram e constituíram o todo. Nenhum padrão de projeto foi utilizado, mas as classes DMInterface e Grid poderiam ser singletons, por exemplo.

A parte mais longa e trabalhosa foi a decisão e implementação da interface gráfica, mas no final atingimos um resultado bom, com painéis bastante simples e elegantes, sem nenhuma dificuldade para o usuário final.

Bibliografia

Foram utilizadas todos os slides fornecidos pela professora, principalmente o referenta a aula do dia 17/10 (GUI 4).