UNIVERSIDADE FEDERAL DE SÃO CARLOS

CENTRO DE CIÊNCIAS E TECNOLOGIAS PARA A SUSTENTABILIDADE

CAMPUS SOROCABA

DEPARTAMENTO DE COMPUTAÇÃO

PROGRAMAÇÃO ORIENTADA A OBJETOS

TRABALHO INTEGRADO

CAIO BONAFÉ CACURO: 489654

GABRIELA DE JESUS MARTINS: 489689

MATHEUS MORAES PINHEIRO: 489719

THAIS AKEMI IWANAGA INABA: 489891

VALDEIR SOARES PEROZIM: 489786

TURMA A

Sorocaba

2014

Sumário

1 Introdução **2**

2 Arquitetura do Sistema **2**

3 Classes do Sistema **2**

3.1 Bibliotecas 2

3.2 Model 3

3.3 Control 3

3.4 View 4

4 Diagrama de classes **5**

5 Estruturas de Dados Utilizadas **6**

6 Operação do Sistema **6**

1 Introdução

Documentação da implementação do sistema de administração de salas de Cinema Faceli & Sakata Ltda, onde descrevemos as decisões arquiteturais do sistema, estruturas de dados utilizadas. Também descrevemos as classes em que o sistema foi decomposto.

2 Arquitetura do Sistema

O projeto foi organizado baseando-se no padrão arquitetural Model-View-Control(MVC), que consiste em separar as responsabilidades das classes de um sistema em três tipos fundamentais de componentes.

Os componentes na pasta model, são a implementação das entitidades descritas no diagrama de classes. Aqueles na pasta control e View equivalem aos itens control e boundary do diagrama de classes, respectivamente.

As classes do model armazenam os dados das entidades e as regras de negócios da aplicação.

As classes do control instanciam classes da view e da model e estabelecem sua comunicação. Também instanciam classes auxiliares para realizar a interface com os arquivos onde são salvos os dados das entidades ou de onde são lidos para recuperar informações previamente cadastradas.

Estas classes auxiliares que implementamos para realizar a interface entre as entidades e operações em arquivo, seguem o padrão Data Access Object(DAO). Elas também se encontram na pasta model e tem seu nome segundo o padrão <nome entite>DAO.

3 Classes do Sistema

3.1 Bibliotecas

Criamos algumas classes para servir de auxílio na implementação das funcionalidades do sistema e organizamo-nas na pasta lib do projeto.

Nela estão as classes Data, Hora e Lista.

A classa Data contém os métodos necessários para alterar e modificar as datas do programa (data de compra do ingresso, data da sessão). A classe também possui a sobrecarga de três operadores (==, != e «) que são necessários para comparar duas eventuais datas.

A classe Hora contém os métodos necessários para definir os horários do programa (hora da compra e hora da sessão). A classe também possui a sobrecarga de dois operadores (== e «) que são necessários para comparar dois eventuais horários.

A classe Lista contém os métodos necessários para aplicar a estrutura de lista na forma de template para que possa ser utilizada por todas as classes.

3.2 Model

Classes Sala, Fileira e Assentos que formam uma objeto por composição.

A classe Sala possui todos os métodos necessários para administrar uma sala de cinema, bem como definir o número da sala, sua capacidade e a quantidade de fileiras, alocar ou desalocar uma sala, reservar ou liberar assentos que serão comprados e determinar qual sua situação de uso (removida, disponível, equipamentos em manutenção, em reforma ou em manutenção geral)

A classe Fileira possui os métodos para administrar a situação dos assentos de cada fileira. É possível adicionar, remover e verificar a disponibilidade do assento.

A classe Assentos possui o número de cada assento e de cada fileira. Verifica a disponibilidade, realiza reserva e liberação dos assentos desejados durante a compra dos ingressos.

Sessão, Ingressos e Venda, que interagem para realizar a funcionalidade de emissão e venda de ingressos.

A classe Sessão possui boa parte dos métodos que definem o funcionamento do cinema. A classe define o número, o horário e o status das sessões, bem como o filme a ser exibido e a quantidade de ingressos disponíveis.

A classe Ingresso define o tipo do ingresso, seu valor e a data da compra.

A classe Venda define os métodos necessários para realizar a venda de ingressos e as formas de pagamento possíveis. Possui o método para calcular o valor dos ingressos, para adicionar, remover e visualizar ingressos.

As classes ´DAO´ são interfaces para gravar e ler os dados das entidades a partir de arquivos.

3.3 Control

Classe GerenciarSala para intermediar as ações de CRUD(Inclusão, alteração, remoção e consulta) de salas. Equivale ao elemento gerenciarSala do diagrama.

GerenciarSessao, equivalente ao item homônimo no diagrama, para intermediar as ações de CRUD de sessoes.

VenderIngresso, equivalente ao item homônimo no diagrama, para intermediar as ações de venda de ingressos.

3.4 View

Classe ViewSala, utilizada para apresentar os dados de Salas cadastradas e receber informações de cadastro. É a implementação do boundary GerenciaSala.

ViewSessao, da mesma, forma, é a responsável por Sessoes e é a implementação de GerenciaSessao.

ViewVenda é a implementação do boundary VendaIngresso e reponsável por cadastrar vendas.

4 Diagrama de classes

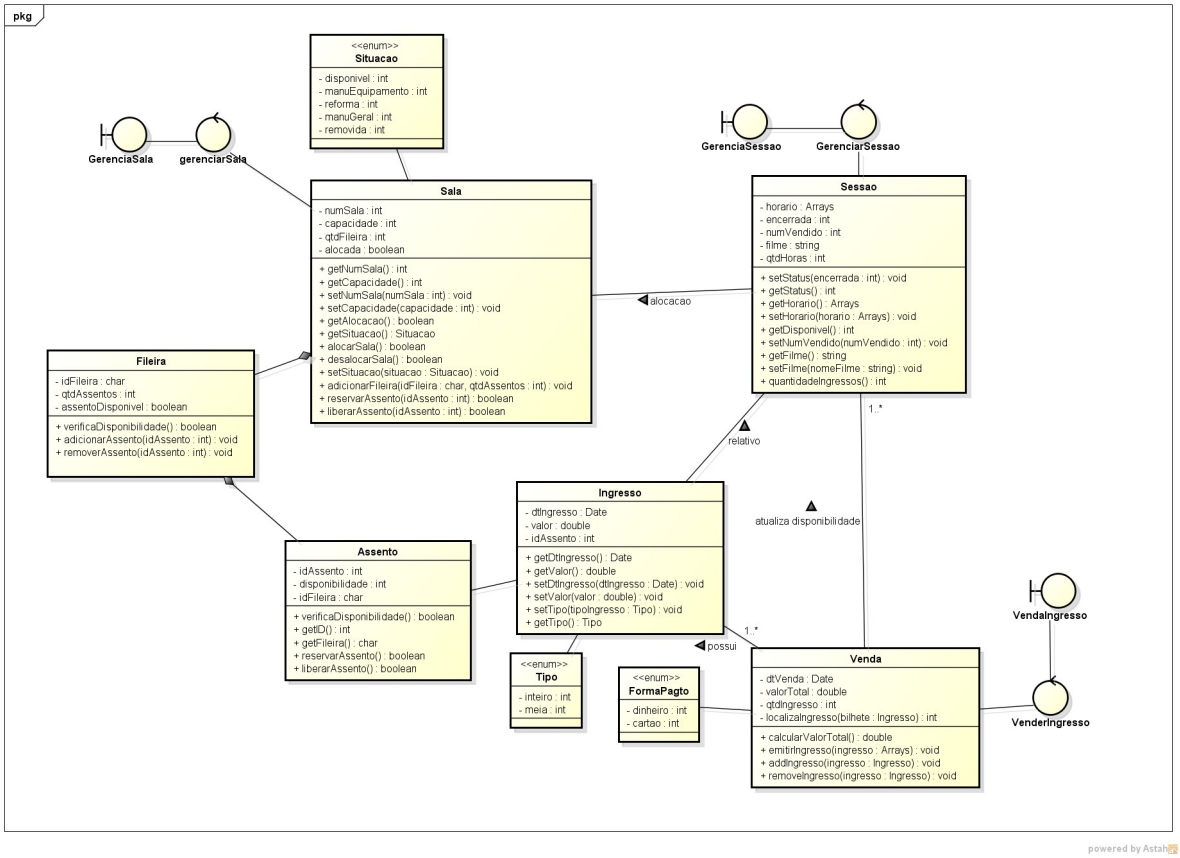

Figura 1: Diagrama de classes

[1] Aumentando o *zoom* do leitor de PDF é possível ver o diagrama em detalhes.

5 Estruturas de Dados Utilizadas

Utilizamos uma lista implementada em vetor estático devido ao fato de ser de simples implementação e a quantidade de objetos que necessitamos armazenar não ir além de 300 objetos.

A lista é ordenada sempre que uma busca precisa ser feita, então utilizamos uma algoritmo de busca binária para realizá-la com maior eficiência.

Implementamos os seguintes métodos:

- elementoEm() : Encontra um elemento na posição <pos> e o retorna ou lança exceção do tipo string.
- localiza() : Tenta encontrar um elemento igual ao elemento passado como parâmetro e retorna sua posição ou -1, caso contrário.
- tamanho() : Retorna a quantidade de elementos na lista.
- vazia() : Retorna true se a lista está vazia, false, caso contrário.
- insere() : Insere um elemento do tipo sobrecarregado no template(T). Retorna exceção do tipo string se atingiu o limite de elementos da Lista.
- ordena() : Faz a ordenação da lista pelo método selection sort. A cada inserção o atributo ordenado é definido false, caso haja necessidade de uma busca, o método localiza, ordenará a lista antes.
- remove() : Remove o elemento passado como parâmetro, caso não o encontre, retorna uma exceção do tipo string.
- mostra() : Imprime os elementos da lista.

6 Operação do Sistema

O sistema foi compilado com o utilitário make do ambiente cygwin64.