Erick de Lima Mascarenhas

Alterações realizadas

Agenda de Pagamentos

Criei uma lista que armazena todas as agendas disponíveis no sistema, que volta ao padrão quando o sistema é zerado. Acrescentei o método “criarAgendaDePagamentos” na classe “EmpregadoServices”, que adiciona novas agendas à lista, alterei os métodos “getAtributoEmpregado” e “alteraEmpregado” para funcionarem com “agendaPagamento”, e adicionei novas exceções de acordo com os erros nos testes. Por fim, modifiquei a classe “FolhaDePagamentoServices” para se adequar às agendas personalizadas, adicionando principalmente um método para verificar se é dia de pagamento, essencial para executar corretamente a totalFolha.

Abaixo segue a primeira documentação enviada

Pacote Core (wepayu.core)

Sistema.java

Como o núcleo do código, ela é quem gerencia o ciclo de vida da aplicação, inicializando os serviços necessários e cuidando para que os dados sejam salvos e carregados corretamente. Para isso, ela usa os métodos carregar e salvar, que lidam com um arquivo XML para que as informações dos empregados não se percam. Uma das suas características mais interessantes é o controle de "desfazer" e "refazer" (undo/redo), que usa duas pilhas para guardar as ações do usuário e permitir que elas sejam revertidas facilmente. Há também o método zerarSistema, que serve para restaurar a aplicação ao seu estado inicial.

Pacote de Modelos (wepayu.models)

Empregado.java

Esta é a classe-base, o molde para qualquer tipo de empregado. Ela define tudo o que eles têm em comum: ID, nome, endereço, salário e informações sobre pagamento e filiação a sindicatos. Também inclui a lógica para lidar com taxas de serviço para os sindicalizados.

Assalariado.java

Derivada de Empregado, a classe Assalariado representa o funcionário com salário fixo mensal. É o tipo mais direto, sem cálculos complexos além do seu salário base.

Horista.java

Também baseada em Empregado, a classe Horista é para quem recebe por hora trabalhada. Ela controla uma lista de cartões de ponto e possui a lógica para adicionar novos registros (lancaCartao) e para somar as horas normais e extras em um período, o que define quanto o funcionário irá receber.

Comissionado.java

Este tipo de empregado, que também herda de Empregado, tem um salário fixo e ganha uma comissão sobre as vendas. A classe gerencia uma lista de vendas e tem métodos para registrar novas vendas e calcular o total vendido, determinando assim o valor final da comissão.

Sindicato.java

Esta é uma classe de apoio para gerenciar as informações do sindicato. Ela garante que os IDs dos membros sejam sempre únicos, mantendo uma lista estática para controle. Seus métodos ajudam a verificar e manipular esses IDs de forma segura.

Pacote de Serviços (wepayu.services)

EmpregadoServices.java

Essa classe é a responsável por todo o ciclo de vida de um empregado no sistema. Ela usa um HashMap para guardar os funcionários de forma organizada. Suas funções vão desde criar e remover empregados até buscar e alterar suas informações. Ela também cuida de tarefas mais específicas, como converter um empregado de um tipo para outro e lançar cartões de ponto ou taxas de serviço. Um ponto importante é que toda alteração de dados é encapsulada como um "Comando", o que viabiliza as funções de desfazer e refazer.

FolhaDePagamentoServices.java

Como o nome diz, esta classe cuida dos cálculos e da geração da folha de pagamento. O método totalFolha faz uma prévia do valor total a ser pago em uma data específica. Já o rodaFolha é quem realmente processa os dados, gerando um relatório completo em um arquivo. Ele calcula salários brutos, descontos e salários líquidos para cada funcionário, agrupando-os por tipo para maior clareza. Assim como no EmpregadoServices, as operações são tratadas como comandos para que possam ser desfeitas se necessário.

Pacote de Específicos (wepayu.specifics)

Cartao.java, TaxaServico.java e Venda.java

Representam o cartão de ponto de um horista, guardando a data e as horas trabalhadas, a taxa de serviço de um membro de sindicato, com data e valor, e vendas feitas por um comissionado, armazenando datas e valores.

Pacote de Utilitários (wepayu.utils)

Comando.java

Uma interface que define o padrão Command. Com os métodos executar() e desfazer(), ela permite "empacotar" uma operação para que ela possa ser revertida depois, sendo a base para o sistema de undo/redo.

Conversao.java

Uma classe de apoio fundamental, cheia de métodos estáticos para validar e converter dados. Ela transforma texto em números (salários, comissões), garantindo que os valores sejam válidos. Também cuida da conversão e validação de datas, formatação de horas e geração de IDs, além de conter regras de negócio importantes para o cálculo da folha de pagamento.

Pacote de Exceções (wepayu.Exception)

Para lidar com erros de forma clara, este pacote define várias exceções personalizadas. Em vez de um erro genérico, o sistema pode lançar exceções como

EmpregadoNaoExisteException ou SalarioNuloException, necessário para o tratamento de erros. Cada exceção carrega uma mensagem que descreve exatamente o que deu errado.