Aluno: Anderson José dos Santos(back-end – terça/quinta-noite)

Aplique o padrão de projeto Observer para criar uma simples editor de texto. Considere a solução apresentada no Hipertexto 6 e utilize o código visto para implementar novas classes. Os requisitos para avaliar o projeto são:

- Implementar uma subclasse da classe Editor chamada TextEditor;

- Aplicar o método insertLine, que recebe os parâmetros lineNumber e text;

- Inserir o texto na ordem correspondente a lineNumber e deslocar as linhas posteriores se necessário;

- Aplicar o método removeLine, que rece lineNumber como parâmetro, deleta o texto da linha correspondente e desloca as linhas posteriores se necessário;

- Instanciar um TextEditor e disparar o evento "open";

- Receber as linhas de texto, que serão inseridas no objeto textEditor, do usuário até que ele envie o texto "EOF", que não deve ser inserido no objeto textEditor;

- Por fim, o textEditor deve disparar o evento "save" para que o conteúdo seja salvo no arquivo configurado e imprimir todo o conteúdo do arquivo na tela.

Resposta em JavaScript:

// Classe base Observador

class Observador {

    atualizar() {

      throw **new** *Error*('O método atualizar deve ser implementado nas subclasses.');

    }

  }

  // Classe Editor (Assunto)

  class Editor {

    constructor() {

*this*.observadores = [];

    }

    adicionarObservador(*observador*) {

*this*.observadores.push(*observador*);

    }

    removerObservador(*observador*) {

      const indice = *this*.observadores.indexOf(*observador*);

      if (indice !== -1) {

*this*.observadores.splice(indice, 1);

      }

    }

    notificar(*evento*) {

*this*.observadores.forEach(*observador* => {

*observador*.atualizar(*evento*);

      });

    }

  }

  // Subclasse TextEditor que estende Editor

  class TextEditor extends Editor {

    constructor() {

*super*();

*this*.linhas = [];

    }

    inserirLinha(*numeroLinha*, *texto*) {

      if (*numeroLinha* >= 0 && *numeroLinha* <= *this*.linhas.length) {

*this*.linhas.splice(*numeroLinha*, 0, *texto*);

*this*.notificar('inserirLinha');

      }

    }

    removerLinha(*numeroLinha*) {

      if (*numeroLinha* >= 0 && *numeroLinha* < *this*.linhas.length) {

*this*.linhas.splice(*numeroLinha*, 1);

*this*.notificar('removerLinha');

      }

    }

    abrir() {

*this*.notificar('abrir');

    }

    salvar() {

*this*.notificar('salvar');

    }

  }

  // Classe para imprimir o conteúdo

  class ImpressoraDeConteudo extends Observador {

    constructor(*textEditor*) {

*super*();

*this*.textEditor = *textEditor*;

*this*.textEditor.adicionarObservador(*this*);

    }

    atualizar(*evento*) {

      if (*evento* === 'abrir' || *evento* === 'salvar') {

        console.log('Conteúdo do arquivo:');

*this*.textEditor.linhas.forEach((*linha*, *indice*) => {

          console.log(`Linha ${*indice* + 1}: ${*linha*}`);

        });

        console.log('---------------------------');

      }

    }

  }

  // Função principal para interação com o usuário

  function principal() {

    const textEditor = **new** TextEditor();

    const impressoraDeConteudo = **new** ImpressoraDeConteudo(textEditor);

    console.log('Digite as linhas de texto (digite "EOF" para encerrar):');

    let numeroLinha = 0;

    let entrada;

    while (true) {

      entrada = prompt(`Linha ${numeroLinha + 1}: `);

      if (entrada === 'EOF') {

        break;

      }

      textEditor.inserirLinha(numeroLinha, entrada);

      numeroLinha++;

    }

    textEditor.salvar();

  }

  principal();

