Trabalho POO

Nome: Evandro Pereira dos Santos **RA:** 1197471

Curso: Sistemas de Informação **Turma** : 8º Período Noturno

Respostas

1. O **Swing é um framework** que disponibiliza um conjunto de elementos gráficos para ser utilizado na plataforma Java. O Swing é compatível com o **Abstract Window Toolkit (AWT)**, mas trabalha de forma totalmente diferente. A **API Swing**, diferente do **AWT**, não delega a tarefa de renderização ao sistema operacional, ele renderiza os elementos por conta própria.

O **Swing** é mais completo e os programas têm uma aparência muito parecida, independente do sistema operacional que está sendo utilizado, possui uma enorme gama de controles extras disponíveis, tais como áreas de texto que nativamente podem mostrar conteúdo como RTF ou HTML, botões com suporte a imagens, slides, selecionadores de cores, alteração do tipo de borda para os componentes, maior controle de como desenhar os mínimos detalhes de apresentação e muito mais.

A tecnologia é tão difundida atualmente pois oferecem uma série de recursos que possibilitam agilidade na produção de softwares. As ferramentas, por exemplo na biblioteca do jframe, ajudam a construção de códigos automáticos conforme a montagem do design de forma simplificada

2. Padrão de projeto e foca seu estudo no modelo Model-View-Controller citando as vantagens e desvantagens na sua utilização. Demonstra e exemplifica como fazer o uso correto das três camadas do **padrão MVC**.

O MVC é um padrão de arquitetura de software. O MVC sugere uma maneira para você pensar na divisão de responsabilidades, principalmente dentro de um software web.

O princípio básico do MVC é a divisão da aplicação em três camadas: a camada de interação do usuário (**view**), a camada de manipulação dos dados (**model**) e a camada de controle (**controller**).

Com o MVC, é possível separar o código relativo à interface do usuário das regras de negócio, o que sem dúvida traz muitas vantagens que veremos mais à frente.

3. Os componentes GUI Swing estão dentro do pacote **javax.swing** que são utilizados para construir as interfaces gráficas. Conhecida como Interface Gráfica com Usuário (Graphical User Interface GUI), é onde os resultados são apresentados em modo gráfico. Essa interface é formada através de componentes GUI, conhecidos por controles ou widgets. Esses componentes são objetos que fazem a interação com usuário por teclado, mouse ou outros dispositivos que venham a servir para entrada de dados.

A tecnologia é maior difundida atualmente quando é necessário desenvolver sistemas que precisam de alguma interação mais aprimorada com o usuário, utiliza-se as interfaces gráficas, então a ideia começa a evoluir, para ajudar nesse tipo de desenvolvimento usamos a interface GUI.A seguir os componentes mais usados;

- **JLabel** - Exibe texto não editável ou ícones.
- **JTextField** – Insere dados do teclado e serve também para exibição do texto editável ou não editável.
- **JButton** – Libera um evento quando o usuário clicar nele com o mouse.
- **JCheckBox** – Especifica uma opção que pode ser ou não selecionada.
- **JComboBox** – Fornece uma lista de itens onde possibilita o usuário selecionar um item ou digitar para procurar.
- **JList** – Lista de itens onde pode ser selecionado vários itens.
- **JPanel** – É a área onde abriga e organiza os componentes inseridos.

4. Para entendermos exatamente do que se trata a orientação a objetos, vamos entender quais são os requerimentos de uma linguagem para ser considerada nesse paradigma. A seguir os paradigmas da POO.

Abstração: significa "esconder" partes da implementação do objeto expondo apenas uma interface simples para seu uso. Pense por exemplo num forno de micro-ondas, você não precisa entender toda a complexidade de como os componentes internos trabalham para gerar as ondas e produzir calor, você quer apenas apertar um ou dois botões e ter uma refeição quente pra comer.

O ***encapsulamento*** é uma das principais técnicas que define a programação orientada a objetos. Se trata de um dos elementos que adicionam

segurança à aplicação em uma programação orientada a objetos pelo fato de esconder as propriedades, criando uma espécie de caixa preta.

Ao **encapsularmos** um objeto estamos agrupando propriedades e métodos que estão diretamente relacionados dentro de um mesmo objeto, permitindo que essas propriedades sejam acessadas apenas através de métodos públicos. Desta forma tratamos de questões importantes como segurança e confiabilidade do estado do objeto.

A **Herança** é uma forma de eliminar repetição de código onde, como o próprio nome sugere, um objeto pode herdar características (ou seja, propriedades e métodos) de outra classe, sem a necessidade de se reescrever essas mesmas características.

Polimorfismo: **Poli** significa muitas e **Morphos** significa forma, então **Polimorfismo** significa muitas formas. Em **POO** Polimorfismo é caracterizado quando duas ou mais classes possuem métodos com o mesmo nome, mas podendo ter implementações diferentes. Assim, é possível utilizar qualquer objeto que implemente o mesmo método sem nos preocuparmos com o tipo do objeto. Na prática isso nos possibilita remover do nosso código diversos *if statements* ou *switch cases*.