ALPOO

Profs.: Elias, Marcelo e Roberto

JAVA

Interface Gráfica com Usuário

Componentes AWT

- **Os componentes Java AWT são dependentes da plataforma**
- **Os componentes são exibidos de acordo com a visão do sistema operacional.**
- **AWT é pesado, ou seja, seus componentes estão usando os recursos do sistema operacional (SO) subjacente.**

Componentes AWT

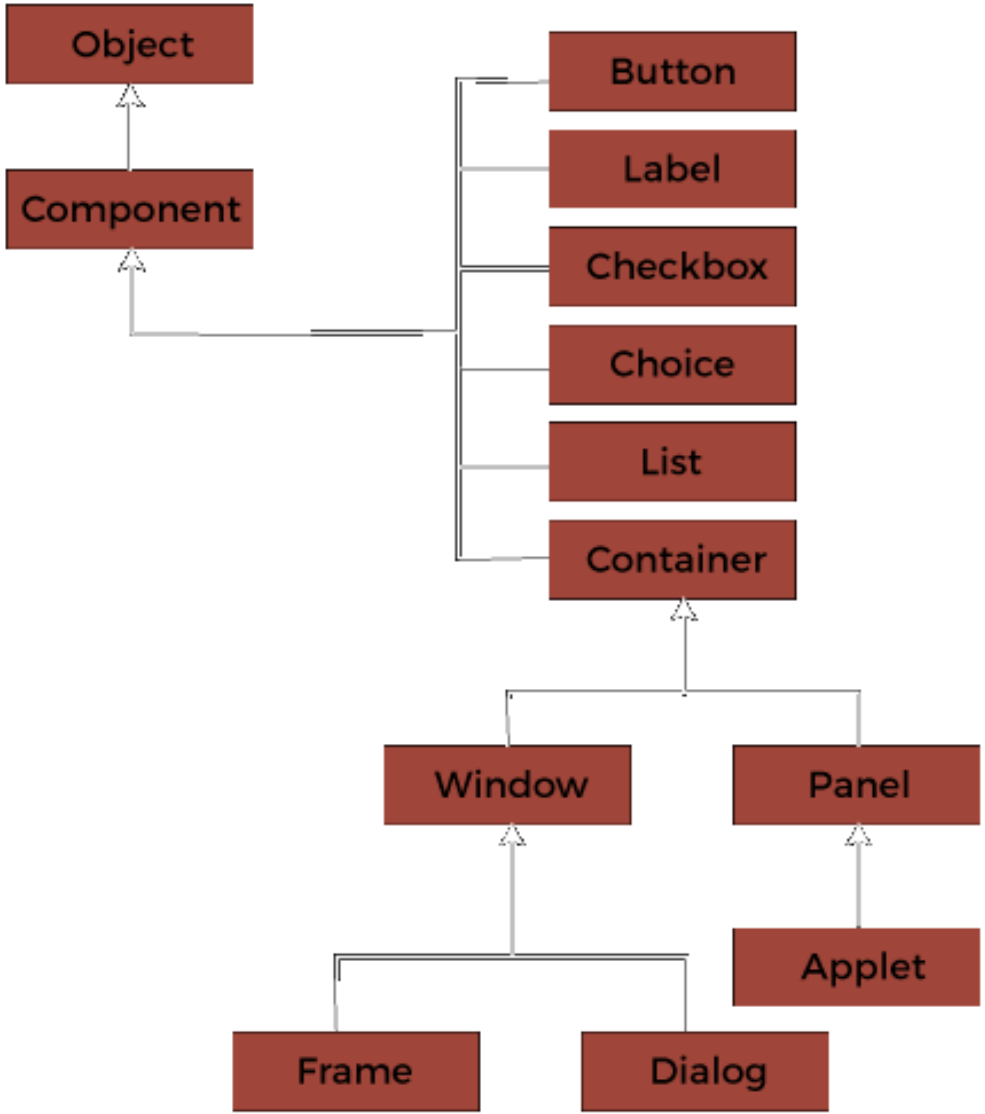
O pacote *java.awt* fornece classes para API AWT como:

- **TextField ,**
- **Label ,**
- **TextArea ,**
- **RadioButton ,**
- **CheckBox ,**
- **Choice ,**

Componentes AWT

- **Java AWT chama a plataforma nativa chama os *sistemas operacionais* para criar componentes de API como TextField, ChechBox, botão, etc.**
- **Por exemplo, uma GUI AWT com componentes como TextField, rótulo e botão terá aparência e comportamento diferentes para as diferentes plataformas, como Windows, MAC OS e Unix.**
- **Um aplicativo AWT se parecerá com um aplicativo do Windows no sistema operacional Windows, ao passo que se**

Hierarquia Java AWT



Componentes

- Todos os elementos como o botão, campos de texto, barras de rolagem, etc. são chamados de componentes.
- Em Java AWT, existem classes para cada componente, conforme mostrado no diagrama acima.
- Para colocar cada componente em uma posição específica na tela, precisamos adicioná-los a um contêiner.

Container (recipiente)

- O Container é um componente no AWT que pode conter outros componentes como botões , campos de texto, rótulos etc.
- As classes que estendem a classe Container são conhecidas como container como Frame, Dialog e Panel .
- É basicamente uma tela onde os componentes são colocados em seus locais específicos. Assim, ele contém e controla o layout dos componentes.
- Nota: Um container em si é um componente (veja o diagrama

Tipos de containers

Existem quatro tipos de contêineres no Java AWT:

- **Window (janela)**
- **Panel (Painel)**
- **Frame (Quadro)**
- **Dialog (Diálogo)**

Window

- A Window é o contêiner que não possui bordas e barras de menu.
- Você deve usar frame, dialog ou outra janela para criar uma janela.
- Precisamos criar uma instância da classe Window para criar este container.

Panel

- O Panel é o container que não contém barra de título, borda ou barra de menu.
- É um recipiente genérico para conter os componentes.
- Ela pode ter outros componentes como botão, campo de texto etc.
- Uma instância da classe Panel cria um container, no qual podemos adicionar componentes.
- Equivalente swing JPanel

Frame

- O Frame é o container que contém barra de título e borda e pode ter barras de menu.
- Ele pode ter outros componentes como botão, campo de texto, barra de rolagem, etc.
- Frame é o contêiner mais amplamente usado durante o desenvolvimento de um aplicativo AWT.
- Equivalente swing JFrame

Métodos úteis de classe de componente

Método	Descrição
public void add(Component c)	Insere um componente neste componente.
public void setSize(int width,int height)	Define o tamanho (largura e altura) do componente.
public void setLayout(LayoutManager m)	Define o gerenciador de layout para o componente.
public void setVisible(boolean status)	Altera a visibilidade do componente, por padrão false.

Java Swing

- **Ao contrário do AWT, o Java Swing fornece componentes leves e independentes de plataforma.**
- **O pacote javax.swing fornece classes para a API java swing, como JButton, JTextField, JTextArea, JRadioButton, JCheckbox, JMenu, JColorChooser etc.**

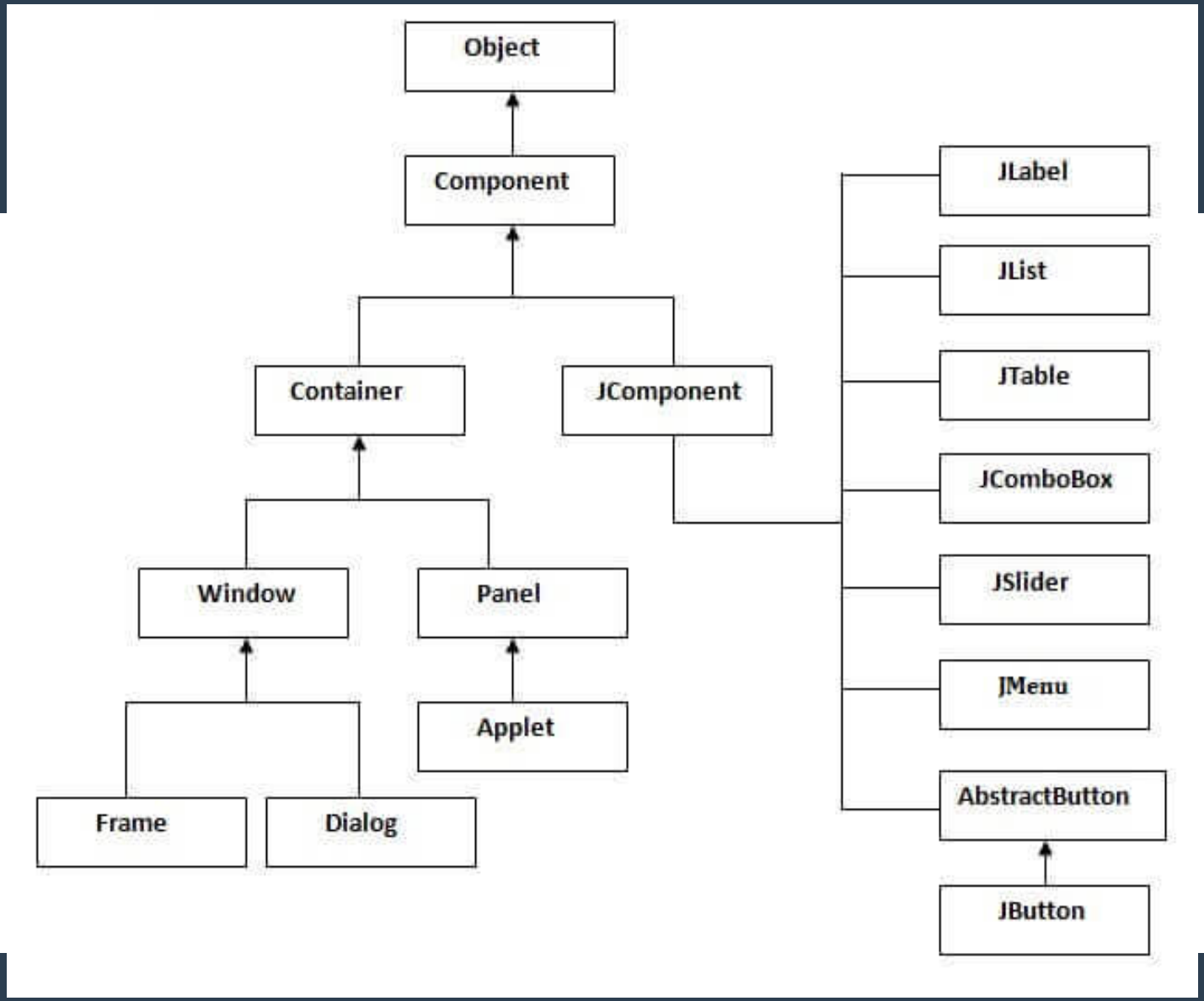
Diferença entre AWT e Swing

Java AWT	Java Swing
Os componentes AWT são dependentes da plataforma .	Os componentes Java swing são independentes de plataforma .
Os componentes AWT são pesados .	Os componentes do balanço são leves .
O AWT não oferece suporte a aparência e comportamento conectáveis .	Swing suporta aparência conectável .

Diferença entre AWT e Swing

Java AWT	Java Swing
O AWT fornece menos componentes do que o Swing.	Swing fornece componentes mais poderosos , como tabelas, listas, scrollpanes, colorchooser, tabbedpane etc.
AWT não segue o MVC (Model View Controller) onde o modelo representa os dados, a visualização representa a apresentação e o controlador atua como uma interface entre o modelo e a visualização.	Swing segue MVC .

Hierarquia Swing



Métodos comumente usados da classe Component

Método	Descrição
public void add(Componente c)	adicionar um componente em outro componente.
public void setSize(int largura,int altura)	define o tamanho do componente.
public void setLayout(LayoutManager m)	define o gerenciador de layout para o componente.
public void setVisible(boolean b)	define a visibilidade do componente. É por padrão falso.

Exemplo

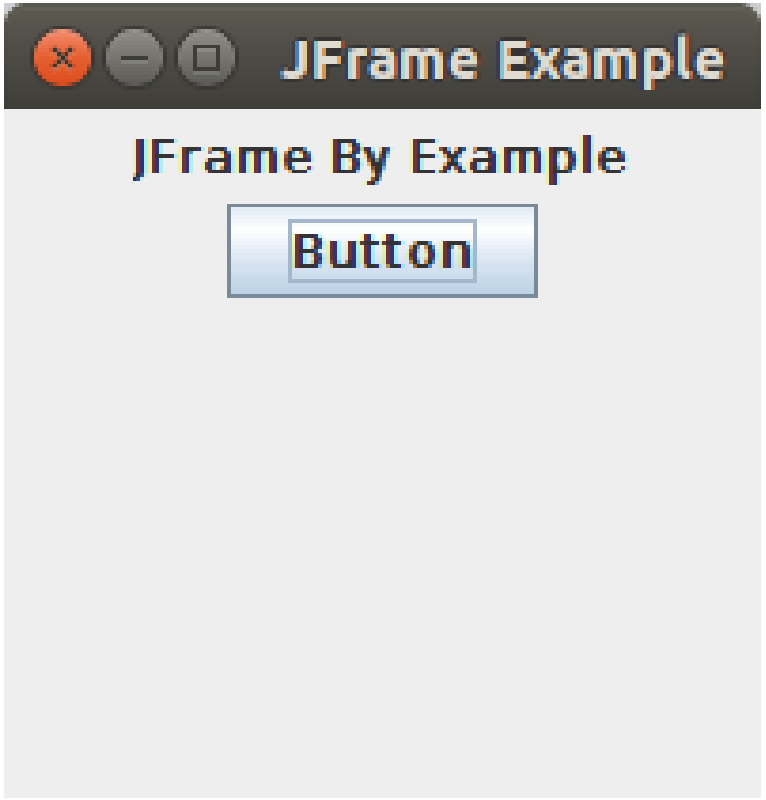
Crie um projeto no netbeans e inclua o código fonte do arquivo: first2.zip

Java JFrame

- A classe javax.swing.JFrame é um tipo de contêiner que herda a classe java.awt.Frame. JFrame funciona como a janela principal onde componentes como rótulos, botões, campos de texto são adicionados para criar uma GUI.
- Ao contrário do Frame, o JFrame tem a opção de ocultar ou fechar a janela com a ajuda do método setDefaultCloseOperation(int).
- Ref.: https://www.javatpoint.com/java-jframe

Java JFrame

- **Exemplo:**



Java GridLayout

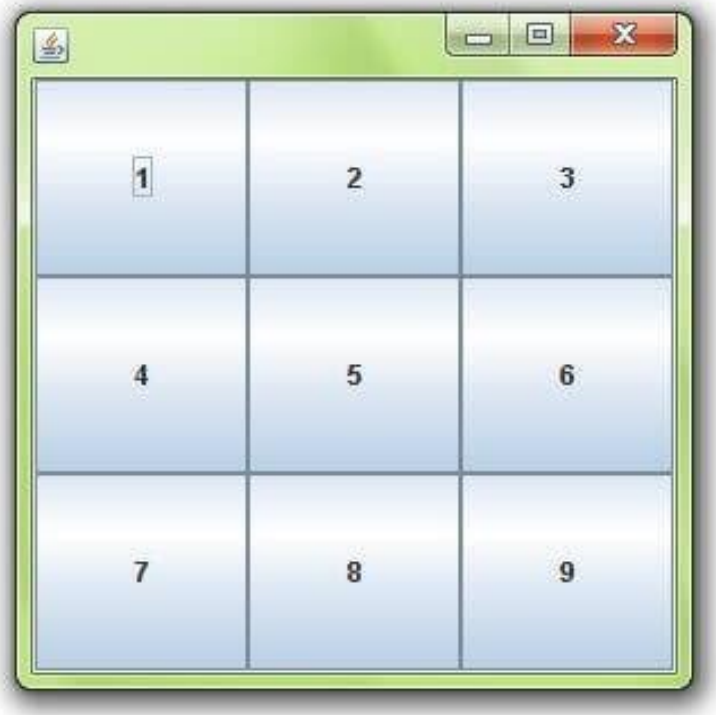
- **A classe Java GridLayout é usada para organizar os componentes em uma grade retangular. Um componente é exibido em cada retângulo.**

- **https://www.javatpoint.com/GridLayout**

Construtores da classe GridLayout

- **GridLayout(): cria um layout de grade com uma coluna por componente em uma linha.**
- **GridLayout(int rows, int columns): cria um layout de grade com as linhas e colunas fornecidas, mas sem intervalos entre os componentes.**
- **GridLayout(int rows, int columns, int hgap, int vgap): cria um layout de grade com as linhas e colunas fornecidas juntamente com as lacunas horizontais e verticais fornecidas.**

Java GridLayout



- Grid criado com 3 linhas e 3 colunas.
- O preenchimento do layout é da direita para esquerda e de cima para baixo.

Java JLabel

- O objeto da classe JLabel é um componente para colocar texto em um container. Ele é usado para exibir uma única linha de texto somente leitura. O texto pode ser alterado por um aplicativo, mas um usuário não pode editá-lo diretamente. Ele herda a classe JComponent.

Construtores comumente usados:

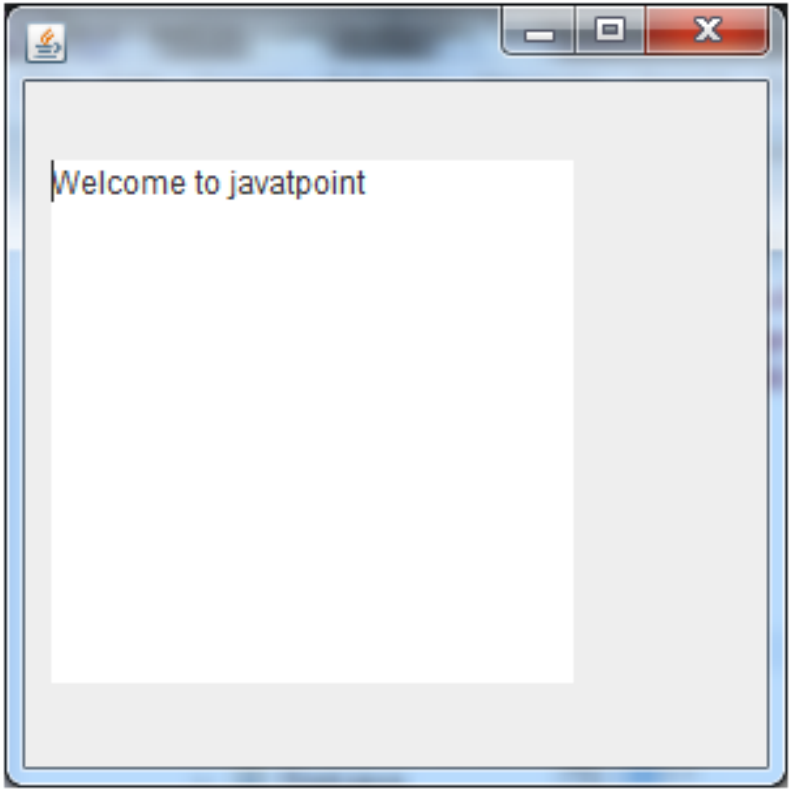
Construtor	Descrição
JLabel()	Cria uma instância JLabel sem imagem e com uma string vazia para o título.
JLabel(String s)	Cria uma instância JLabel com o texto especificado.
JLabel(Ícone i)	Cria uma instância JLabel com a imagem especificada.
JLabel(String s, Icon i, int horizontalAlignment)	Cria uma instância JLabel com o texto, a imagem e o alinhamento horizontal especificados.

Métodos comumente usados:

Métodos	Descrição
String getText()	t retorna a string de texto que um rótulo exibe.
void setText(String text)	Ele define a única linha de texto que este componente exibirá.
void setHorizontalAlignment(int alinhamento)	Ele define o alinhamento do conteúdo do rótulo ao longo do eixo X.
Ícone getIcon()	Retorna a imagem gráfica que a etiqueta exibe.
int getHorizontalAlignment()	Retorna o alinhamento do conteúdo do rótulo ao longo do eixo X.

Java JTextArea

- O objeto de uma classe JTextArea é uma região de várias linhas que exibe texto. Permite a edição de texto de várias linhas. Ele herda a classe JTextComponent



Construtores

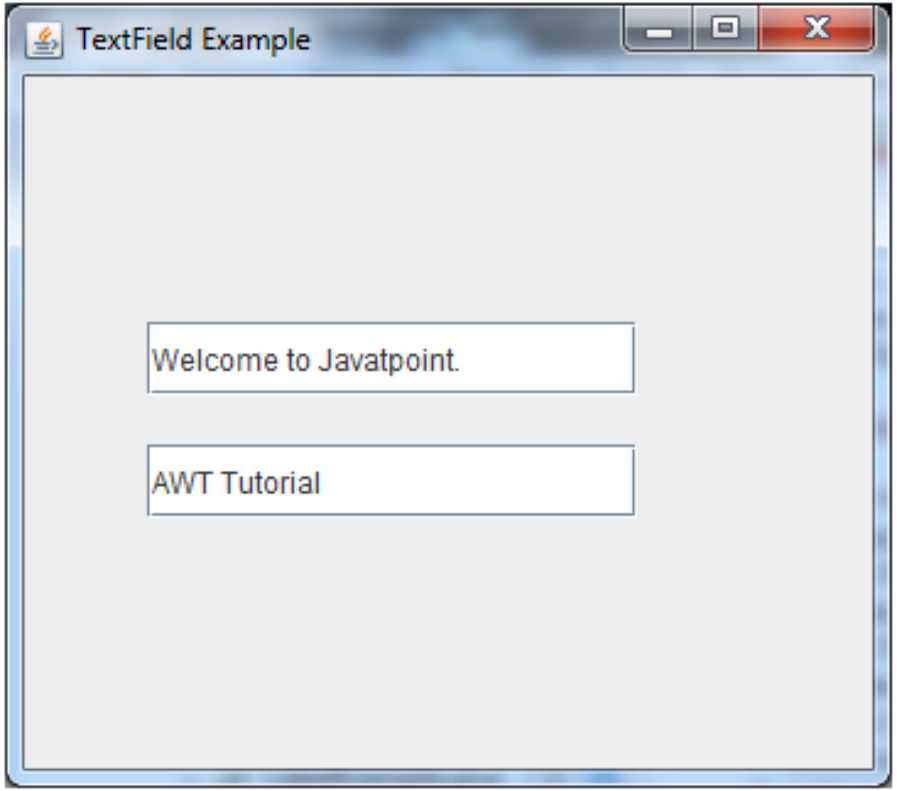
Construtor	Descrição
JTextArea()	Cria uma área de texto que não exibe nenhum texto inicialmente.
JTextArea(String s)	Cria uma área de texto que exibe o texto especificado inicialmente.
JTextArea(int linha, int coluna)	Cria uma área de texto com o número especificado de linhas e colunas que não exibe nenhum texto inicialmente.
JTextArea(String s, int linha, int coluna)	Cria uma área de texto com o número especificado de linhas e colunas que exibe o texto especificado.

Métodos

Métodos	Descrição
void setRows(int linhas)	É usado para definir o número especificado de linhas.
void setColumns(int cols)	É usado para definir o número especificado de colunas.
void setFont(Font f)	É usado para definir a fonte especificada.
void insert(String s, posição int)	É usado para inserir o texto especificado na posição especificada.
void append(String s)	É usado para anexar o texto fornecido ao final do documento.

Java JTextField

- O objeto de uma classe JTextField é um componente de texto que permite a edição de um texto de uma única linha. Ele herda a classe JTextComponent.



Construtores

Construtor	Descrição
JTextField()	Cria um novo TextField
JTextField(String texto)	Cria um novo TextField inicializado com o texto especificado.
JTextField(String text, int colunas)	Cria um novo TextField inicializado com o texto e as colunas especificados.
JTextField(int colunas)	Cria um novo TextField vazio com o número especificado de colunas.

Métodos

Métodos	Descrição
void addActionListener(ActionListener l)	Ele é usado para adicionar o ouvinte de ação especificado para receber eventos de ação deste campo de texto.
Ação getAção()	Ele retorna a ação atualmente definida para esta origem de ActionEvent ou null se nenhuma ação for definida.
void setFont(Font f)	É usado para definir a fonte atual.
void removeActionListener(ActionListener l)	Ele é usado para remover o ouvinte de ação especificado para que ele não receba mais eventos de ação desse campo de texto.
void setEnabled(bolean b)	Para desabilitar JtextField/JTextArea, chame o método setEnabled() e passe o valor “false”
void setEditable(bolean b)	Para desabilitar edição passe o valor false.

Java JComboBox

- O objeto da classe Choice é usado para mostrar o menu popup de opções.
- A escolha selecionada pelo usuário é mostrada no topo de um menu .
- Ele herda JComponent .

Construtores

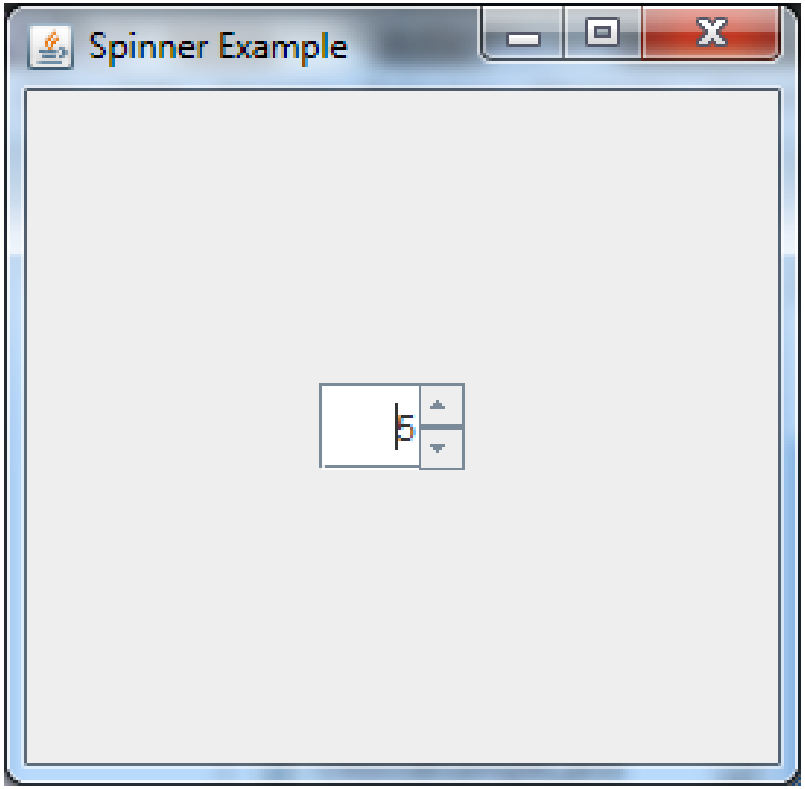
Construtor	Descrição
JComboBox()	Cria um JComboBox com um modelo de dados padrão.
JComboBox(Object[] itens)	Cria um JComboBox que contém os elementos na matriz .
JComboBox(Vetor<?> itens)	Cria um JComboBox que contém os elementos no Vector .

Métodos

Métodos	Descrição
void addItem(Object anObject)	É usado para adicionar um item à lista de itens.
void removeItem(Object anObject)	É usado para excluir um item da lista de itens.
void removeAllItems()	Ele é usado para remover todos os itens da lista.
void setEditable(boolean b)	Ele é usado para determinar se o JComboBox é editável.
void addActionListener(ActionListener a)	Ele é usado para adicionar o ActionListener .
void addItemListener(ItemListener i)	Ele é usado para adicionar o ItemListener

Java JSpinner

- **O objeto da classe JSpinner é um campo de entrada de linha única que permite ao usuário selecionar um número ou valor de objeto de uma sequência ordenada.**



Construtores

Construtor	Descrição
JSpinner()	Ele é usado para construir um spinner com um Integer SpinnerNumberModel com valor inicial 0 e sem limites mínimo ou máximo.
JSpinner (modelo SpinnerModel)	Ele é usado para construir um spinner para um determinado modelo.

Métodos comumente usados:

Método	Descrição
void addChangeListener(ouvinte ChangeListener)	Ele é usado para adicionar um ouvinte à lista que é notificado sempre que ocorre uma alteração no modelo.
Objeto getValue()	É usado para retornar o valor atual do modelo.

Singleton

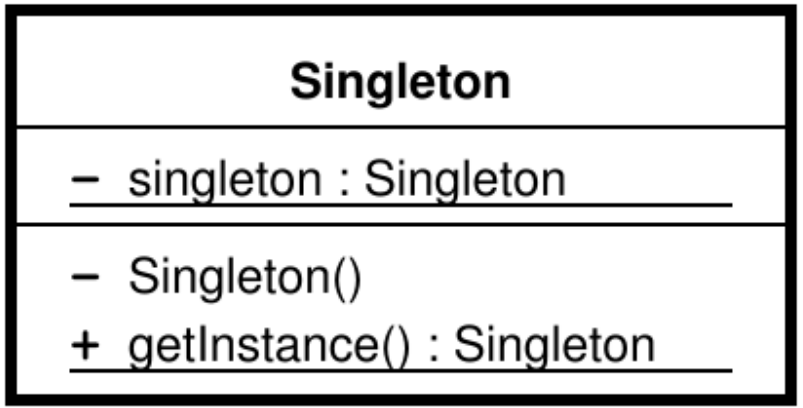
- Os Design Patterns (Padrões de Projetos) são arquiteturas testadas para construir softwares orientados a objetos flexíveis e sustentáveis.
- Os padrões ajudam a reduzir substancialmente a complexidade do processo de design.

Singleton

- O padrão Singleton permite criar objetos únicos para os quais há apenas uma instância.
- Este padrão oferece um ponto de acesso global, assim como uma variável global, porém sem as desvantagens das variáveis globais.

Singleton

- No Diagrama de classe o atributo singleton que é do tipo da sua própria classe e é estático, nessa variável tem-se a única instância da classe.
- Nos métodos pode-se observar a presença do construtor da classe Singleton() que é PRIVADO, que não permite que a classe seja instanciada a não ser por ela mesmo
- Ela é instanciada pelo método getInstance() que é estático e assim pode ser acessado de qualquer outra classe sem precisar instanciar Singleton.



Singleton

```
1  public class Singleton {
2
3      private static Singleton uniqueInstance;
4
5      private Singleton() {
6      }
7
8      public static synchronized Singleton getInstance() {
9          if (uniqueInstance == null)
10             uniqueInstance = new Singleton();
11
12         return uniqueInstance;
13     }
14 }
```

Bibliografia

- **JavaTpoint, Tutorial java disponível em:**

https://www.javatpoint.com/

- **Devmedia,Padrão de Projeto Singleton em Java disponível em:**

https://www.devmedia.com.br/padrao-de-projeto-singleton-em-java/26392