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. Como a AWT é uma biblioteca de baixo-nível que depende de código nativo da plataforma ela traz alguns problemas de compatibilidade entre as plataformas, fazendo com que nem sempre o programa tenha a aparência desejada em todos os sistemas operacionais. Além disso, 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, sliders, selecionadores de cores, alteração do tipo de borda para os componentes, maior controle de como desenhar os mínimos detalhes de apresentação e muito mais. No entanto, a performance é um pouco pior devido a alta abstração, consumindo assim mais memória RAM. Os principais componentes do Swing são resumidos abaixo:

- JFrame representa a janela do programa com barra de título, ícone, botões de comando, etc. Entre os principais métodos temos o pack() que compacta a janela para o tamanho dos componentes, setSize(int, int) que define a largura e altura da janela, setLocation(int, int) que define a posição da janela na tela (x,y), setBounds(int, int, int, int) que define posição e tamanho, setVisible(boolean) que exibe a janela e setDefaultCloseOperation(int) que define o que ocorre quando o usuário tenta fechar a janela (as opções são: DO_NOTHING_ON_CLOSE, HIDE_ON_CLOSE, DISPOSE_ON_CLOSE, EXIT_ON_CLOSE).
- JPanel representa um tipo básico de container para inserção de componentes. Entre os principais métodos temos add(Component, int) que adiciona o componente definindo sua posição e setLayout(LayoutManagaer) que altera o tipo de layout.
- JLabel representa um rótulo de texto. Entre os principais métodos temos o setText(String) que altera o texto e getText() que retorna o texto atual.
- JTextFiel representa um campo de texto onde o usuário pode informar um texto em uma linha. Entre os principais métodos temos setText(String) que altera o texto e getText() que retorna o texto atual.
- JPasswordField representa um campo de texto protegido, subclasse de JTextField. O principal método é o setEchoChar(char) que define o caractere que aparece ao digitar um texto.
- JTextArea representa uma caixa onde o usuário pode informar várias linhas de texto. Entre os principais métodos temos o setText(String) que altera o texto, getText() que retorna o texto atual, getSelectedText() que retorna o texto selecionado pelo usuário e insert(String, int) que insere um texto na posição especificada.
- JCheckBox representa uma caixa de seleção e permite selecionar ou não uma opção. Entre os principais métodos temos o setSelected(boolean) que altera o estado da caixa de seleção e o

método isSelected() que retorna true se a caixa estiver marcada e false se não estiver marcada.

- JRadioButton representa um componente que permite seleciona uma entre diversas opções. O JRadioButton é semelhante ao JCheckBox, inclusive com os mesmos construtores e métodos.
- JComboBox representa uma caixa de combinação, da qual o usuário pode selecionar uma opção. Entre os principais métodos temos o addItem(Object) que adiciona um item à lista de opções, setEditable(boolean) que permite ao usuário digitar uma opção, getSelectedIndex() que retorna a posição do item atualmente selecionado, getSelectedItem() que retorna o texto do item atualmente selecionado, setSelectedIndex(int) que seleciona o item da posição especificada e setSelectedIndex(Object) que seleciona o objeto especificado na lista.
- JList representa uma lista de opções que permite a seleção de mais de um item simultaneamente. Entre os principais métodos temos o setListData(Object[]) que preenche ou altera os itens de uma lista, getSelectedValues() que retorna um array de objetos contendo itens selecionados na lista.
- JButton representa um botão destinado a executar uma ação. Entre os principais métodos temos o setText(String) que altera o texto do botão e setIcon(Icon) que altera o ícone do botão.