Imprimir livro

Semana 1: Swing

Site:	moodle23.iscte-iul.pt	Impresso por:	Diana Andreia Amaro
Disciplina:	Programação Concorrente e Distribuída	Data:	terça-feira, 12 de setembro de 2023 às 14:00
Livro:	Semana 1: Swing		

Descrição

Exercícios de interfaces gráficas

Índice

1. Janela - JFrame, JLabel, JTextField, JButton, JCheckBox, FlowLayout, GridLayout

O objetivo deste exercício é tomar contacto com as classes do *Swing* mais elementares, para representar janelas (`JFrame`), painéis (`JPanel`), etiquetas (`JLabel`), campos de texto (`JTextField`), botões (`JButton`, `JCheckBox`), e formas de os organizar (`FlowLayout`, `GridLayout`).

1. Experimente executar a classe em anexo (`FrameTest`), alterando alguns dos valores utilizados no exemplo, tais como o texto utilizado nos elementos e os valores utilizados no `GridLayout`.
2. Com base na estrutura da classe dada, desenvolva uma outra classe cujo aspeto da interface gráfica seja semelhante ao apresentado na figura em baixo. Deverá ser possível executar a classe configurando a janela com um texto para o título e uma dimensão inicial:

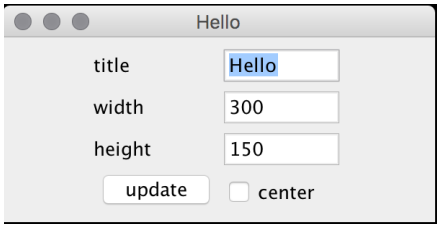
```
public static void main(String[] args) {

     MyFrame window = new MyFrame("Hello",300, 150);

     window.open();

}
```

Ao carregar no botão a janela deverá ser redimensionada de acordo com os valores nas caixas de texto, e caso a checkbox esteja selecionada, a localicação da janela deverá também ser alterada para o centro do ecrã.



Dica: A `JFrame` pode redimensionada utilizando o método `setSize(...)` da classe `JFrame`. O título pode ser alterado através de `setTitle(...)`

Dica: É possível obter a dimensão do ecrã da seguinte forma:

```
Dimension dimension = Toolkit.getDefaultToolkit().getScreenSize();
```

Dica: Para alterar a posição da janela deve usar-se o método *setLocation(int x,int y)* da classe *JFrame*.

Dica: não é preciso que o resultado seja exatamente como apresentado na imagem acima: não considerar alinhamentos e outros detalhes de apresentação.

Anexo

2. Visualizador de imagens - BorderLayout, ImageIcon

Desenvolva uma aplicação que recebe um argumento de execução (parâmetro do `main()`) com o caminho para uma pasta, de modo a permitir visualizar as imagens aí contidas (ver imagem e vídeo abaixo). O utilizador pode percorrer as imagens utilizando os botões. Quando o utilizador tentar ir para a frente da última imagem ou para trás da primeira deve ser mostrada uma mensagem na zona de exibição de imagens. No topo deverá ser mostrado o nome da imagem que está a ser mostrada. Por fim, deverá ser possível recarregar as imagens do diretório através do botão que se encontra na zona inferior.

Para dispor os elementos gráficos da forma apresentada, pode utilizar-se um `BorderLayout` no painel, o que implica que ao adicionar um elemento se deva indicar a sua localização no painel (este, oeste, norte, sul, centro).

frame.add(..., BorderLayout.EAST/WEST/NORTH/SOUTH/CENTER)

A zona de exibição de imagens deverá ser uma `JLabel`, onde é possível colocar uma imagem fazendo:

```
ImageIcon icon = new ImageIcon("ficheiro.png");
label.setIcon(icon);
```

Os ficheiros podem ser lidos de uma pasta filtrando os ficheiros nele contidos (de acordo com determinado critério) da seguinte forma:

```
String path = "images"; // pasta criada dentro do projeto Eclipse,
                        //onde devem ser colocadas as imagens
File[] files = new File(path).listFiles(new FileFilter() {
    public boolean accept(File f) {
        // se retornar verdadeiro, f será incluido
        // colocar return true para serem escolhidos todos os ficheiros
    }
});
```

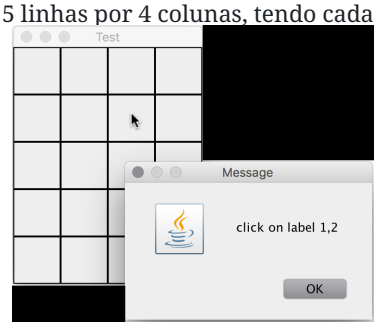
Imagens para teste.
Dica: para obter o caminho completo de um ficheiro, incluindo o nome, em formato *String*, pode usar-se o método `getAbsolutePath()` da classe `File`.

3. Extra: Componente Reutilizável

Desenvolva uma classe para representar grelhas com posições quadradas, oferecendo a possibilidade de definir o número de linhas, o número de colunas, e a dimensão das posições em píxeis (lado de cada quadrado). Deverá ser possível executar a classe da seguinte forma:

```
public static void main(String[] args) {
    Grid grid = new Grid("Test", 5, 4, 50);
    grid.open();
}
```

de modo a ter o título “Test” e uma grelha de 5 linhas por 4 colunas, tendo cada posição 50 píxeis de largura.



1. Defina a disposição das posições da grelha. O *Layout* mais adequado é o `GridLayout`, o qual pode ser inicializado com o número exacto de linhas/colunas. Cada posição pode consistir numa `JLabel` configurada com uma linha de contorno

(`Border`) com uma cor e uma espessura de linha, o que pode ser feito da seguinte forma:

```
Border border = BorderFactory.createLineBorder(Color.black, 2);
label.setBorder(border);
```

De modo a que a grelha tenha a dimensão pretendida podemos utilizar o método `JPanel.setPreferredSize(Dimension)` indicando que o painel tem uma dimensão pretendida, o que será levado em conta no momento em que `JFrame.pack()` executa.

Dica: a Frame pode ser configurada para não permitir redimensionamento manual utilizado `JFrame.setResizable(false)`

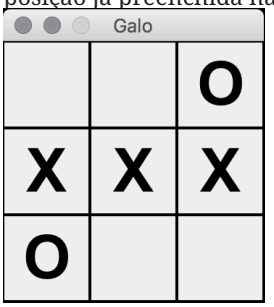
2. Defina o comportamento quando são pressionadas as posições da grelha. Isto pode ser conseguido registando uma sentinela para reagir a eventos do rato (interface `MouseListener`) em cada `JLabel`.

```
label.addMouseListener(new MouseAdapter() {
   public void mouseClicked(MouseEvent e) {
      //...
   }
});
```

`MouseListener` é uma interface com diversas operações relacionadas com eventos do rato. Para conveniência, existe uma classe abstrata MouseAdapter que implementa a interface com todos os métodos sem comportamento, e desta forma pode ser estendida sobrecarregando apenas a operação de interesse (como no exemplo em cima, onde é redefinido apenas o método associado aos cliques do rato).

3. Altere a classe desenvolvida para que a mesma possa ser estendida com vista a alterar o comportamento da aplicação quando é pressionada uma posição da grelha. Com uma solução baseada em herança, isto pode ser conseguido tendo o comportamento das sentinelas definido num método da classe que possa ser sobrecarregado em subclasses.

4. Desenvolva um subclasse de `Grid` para representar a grelha do Jogo do Galo (ver figura), incluindo as marcações de jogadas (com as letras “X” e “O”). As marcas dos jogadores deverão ser inseridas em alternância, sendo que um clique numa posição já preenchida não terá efeito.



Dica: o tipo de letra da `JLabel` pode ser redefinido, bem como o alinhamento horizontal:

```
label.setFont(new Font("Arial", Font.BOLD, 42));
label.setHorizontalAlignment(SwingConstants.CENTER);
```