### Apresentação

Este relatório tem como propósito apresentar o resultado do Trabalho Final da disciplina de Paradigmas de Programação. O objetivo do trabalho é a implementação de um sistema completo com interface gráfica que seja capaz de armazenar dados e utilizar os padrões da programação orientada à objetos.

A velocidade do acesso as informações com o uso da Internet, a possibilidade de baixar muitos arquivos em pouco tempo e a falta de tempo das pessoas, ocasionou problemas de organização em seus computadores. Arquivos dos mais variados tipos guardados em uma única pasta dificulta a localização de um específico. Separar por pastas arquivos que possuem características em comum possibilita organizar a área de trabalho do usuário e melhorar significativamente o seu visual, além de encontrar com rapidez um arquivo .A partir disto, foi-se criado um Organizador de músicas, documentos, vídeos e imagens utilizando a linguagem de programação Java.

### Estrutura

|  |
| --- |
| **InterfaceDiretorios** |
| + AbrirDiretorio() : Void |
| + CriarDiretorio() : Void |

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  |  | |  |
|  |  |  |  |
|  |  |  | |

|  |
| --- |
| **Diretorio** |
| - dir : String |

|  |  |
| --- | --- |
|  |  |
|  |  |

|  |
| --- |
| **Imagens** |
| - image : ArrayList<File> |
| + pastaImage(): void |

|  |
| --- |
| **Documentos** |
| - documents : ArrayList<File> |
| + pastaDocuments() : void |

|  |
| --- |
| **­­Musicas** |
| - music : ArrayList<File> |
| + pasta Music() : void |

|  |
| --- |
| **Videos** |
| - video : Arraylist<File> |
| + pastaVideos(): void |

|  |
| --- |
| **Outros** |
| - other : ArrayList<File> |

**Interface InterfaceDiretorios:**

Contém a assinatura dos métodos que são usados nas classes implementadas

**Abstract Class Diretorio:**

É uma superclasse abstrata e é implementado a InterfaceDiretorios, possui o atributo denominado “dir” do tipo String, ele é responsável por guardar o diretório recebido, no construtor é recebido uma String como parâmetro (Diretório) e é armazenado na variável dir.

**Class Documentos:**

Possui um atributo denominado “documents” do tipo ArrayList, este é capaz de indexar objetos do tipo File e possui um vetor cujo nome é “aux” do tipo File, é usado para receber todos os diretórios filtrados

Método AbrirDiretorio: Possui um objeto denominado “filtro” do tipo FilenameFilter, este é usado filtrar apenas as extensões que do tipo que queremos (pdf, doc, txt,...).

Método CriarDiretório: chama o método pastaDocuments.

Método pastaDocuments: Possui um objeto denominado pastaDocuments do tipo File, ele é usado para criar uma nova pasta e indexar os arquivos pertencentes àquela pasta, como os arquivos estão indexados em um vetor adicionamos em documents para trabalhar melhor e com a pasta já criada, movemos dentro de um laço de repetição um por um para a nova pasta.

**Class Imagens:**

Possui um atributo denominado “image” do tipo ArrayList, este é capaz de indexar objetos do tipo File e possui um vetor cujo nome é “aux” do tipo File, é usado para receber todos os diretórios filtrados

Método AbrirDiretorio: Possui um objeto denominado “filtro” do tipo FilenameFilter, este é usado filtrar apenas as extensões que do tipo que queremos (jpeg, gif, jpg,...).

Método CriarDiretório: chama o método pastaImage.

Método pastaImage: Possui um objeto denominado pastaImage do tipo File, ele é usado para criar uma nova pasta e indexar os arquivos pertencentes àquela pasta, como os arquivos estão indexados em um vetor adicionamos em image para trabalhar melhor e com a pasta já criada, movemos dentro de um laço de repetição um por um para a nova pasta.

**Class Musicas:**

Possui um atributo denominado “music” do tipo ArrayList, este é capaz de indexar objetos do tipo File e possui um vetor cujo nome é “aux” do tipo File, é usado para receber todos os diretórios filtrados

Método AbrirDiretorio: Possui um objeto denominado “filtro” do tipo FilenameFilter, este é usado filtrar apenas as extensões que do tipo que queremos (mp3, wmv, ogg,...).

Método CriarDiretório: chama o método pastaMusic.

Método pastaMusic: Possui um objeto denominado pastaMusic do tipo File, ele é usado para criar uma nova pasta e indexar os arquivos pertencentes àquela pasta, como os arquivos estão indexados em um vetor adicionamos em music para trabalhar melhor e com a pasta já criada, movemos dentro de um laço de repetição um por um para a nova pasta.

**Class Videos:**

Possui um atributo denominado “video” do tipo ArrayList, este é capaz de indexar objetos do tipo File e possui um vetor cujo nome é “aux” do tipo File, é usado para receber todos os diretórios filtrados

Método AbrirDiretorio: Possui um objeto denominado “filtro” do tipo FilenameFilter, este é usado filtrar apenas as extensões que do tipo que queremos (mp4, avi, 3gp,...).

Método CriarDiretório: chama o método pastaVideo.

Método pastaVideo: Possui um objeto denominado pastaVideo do tipo File, ele é usado para criar uma nova pasta e indexar os arquivos pertencentes àquela pasta, como os arquivos estão indexados em um vetor adicionamos em video para trabalhar melhor e com a pasta já criada, movemos dentro de um laço de repetição um por um para a nova pasta.

**Class Outros:**

Possui um atributo denominado “other” do tipo ArrayList, este é capaz de indexar objetos do tipo File e possui um vetor cujo nome é “aux” do tipo File, é usado para receber todos os diretórios filtrados

Método AbrirDiretorio: Possui um objeto denominado “filtro” do tipo FilenameFilter, este é usado filtrar apenas as extensões que são diferentes das que já foram indexadas (verificamos se é difente de: mp3, mp4, jpeg,...) .

Método CriarDiretório: chama o método pastaOther.

Método pastaOther: Possui um objeto denominado pastaOther do tipo File, ele é usado para criar uma nova pasta e indexar os arquivos pertencentes àquela pasta, como os arquivos estão indexados em um vetor adicionamos em other para trabalhar melhor e com a pasta já criada, movemos dentro de um laço de repetição um por um para a nova pasta.

|  |
| --- |
| **Dados** |
| - diretorios: ArrayList<String> |
| - caminhoArquivo: String |
| + addDiretorio (String): void |
| + lerRelatorio(): void |
| +gravarRelatorio(): void |

**Class Dados:**

Possui o atributo denominado “diretorios” do tipo ArrayList, esté capaz de armazenar objetos do tipo String, é usado para armazenar o caminho dos diretórios, possui também outro atributo de nome “caminhoArquivo” , este é usado para receber o caminho da pasta escolhida pelo usuário.

Método addDiretorio: Recebe no parâmetro o caminho da pasta escolhida e adiciona no ArrayList.

Método lerDiretorio: Abre um arquivo em formato de leitura ( FileReader ) ler linha a linha do arquivo a partir do marcador desejado e adiciona a String no ArrayList e então o arquivo é fechado.

Método gravarRelatório: Abre o arquivo no formato de escrever ( FileWriter ) e grava todos os diretórios que estão dentro do ArraylistList linha a linha através de um laço de repetição, por fim, o arquivo é fechado.

|  |
| --- |
| **ActionListener** |
| + actionPerformed(ActionEvent):void |

|  |
| --- |
| **JFrame** |
|  |

­­­­­

|  |
| --- |
| **JanelaDiretorios** |
| - diretorios: JFrame |
| - caixa: Container |
| - layout: BorderLayout |
| - item4: JButton |
| - titulo, resultado: Jlabel |
| - background\_imagem: JImagePanel |
| + inicializarComponentes(): void |
| + lastDiretorios(): void |

|  |
| --- |
| **JanelaCreditos** |
| - credit: JFrame |
| - caixa: Container |
| - layout: BorderLayout |
| - item4: JButton |
| - titulo, nome1 ,nome2, nome3: Jlabel |
| - background\_imagem: JImagePanel |
| + inicializarComponentes(): void |

|  |  |
| --- | --- |
|  |  |
|  |  |

|  |
| --- |
| **Menu** |
| - menuPrincipal: JMenuBar |
| - item1, item2, item3, item4: JMenuItem |
| - menu1, menu2: JMenu |
| - escolherPasta: JfileChooser |
| - painel\_imagem: JImagePanel |
| - dados: Dados |
| + criarJanela(): void |
| + criarMenu:void |
| + abrirPasta(): String |

|  |
| --- |
| **App** |
| + main(): void |

**Interface ActionLister:**

Usada para dar ações aos botões

**Class JFrame:**

É uma superclasse responsável pela criação da janela

**Class App:**

Classe que contém o método main e é criado um objeto do tipo Menu

**Classe Menu:**

Possui atributos do tipo JMenuBar, barra de menu localizada na parte superior da janela (em azul) , JMenu, são os itens que estão disponíveis na barra de menu, JMenuItem, os itens que estão dentro dos JMenu, JFileChooser, usado para abrir a janela de seleção das pasta a serem organizadas , um objeto do tipo JImagePanel, usado para carregar a imagem de fundo e finalmente um objeto do tipo Dado, usado para criar um arquivo de texto e gravar os diretórios organizados.

Método criarJanela: como essa classe herda de JFrame não há a necessidade de criar um objeto deste tipo, logo basta chamar os metodos de JFrame diretamente, neste método defini-se a altura e largura da janela principal, a função do botão de fechar e torna a janela não expansível.

Método criarMenu: adiciona-se a cor azul a barra de menu, adicona-se os itens e sub itens do menu ,o método implementado através da interface ActionLister e é adicionado a imagem como plano de fundo.

Método abrirPasta: É aberto uma caixa de dialogo em que o usuário deve escolher uma pasta a ser organizada e é retornado o endereço dessa pasta.

**Class JanelaDiretorios:**

Possui os atributos: diretorios, caixa, layout, item4, titulo, resultado, background\_imagem dos seguintes tipos respectivamente JFrame, Container, BorderLayout, JButton, JLabel, JLabel, JImagePanel. No construtor da classe é definido o tamanho da janela, borda, função ao apertar o botão para fechar e o layout.

Método incializarComponentes: Renderiza a imagem como fundo de tela, adiciona o botão à tela e em qual posição ele deve ficar.

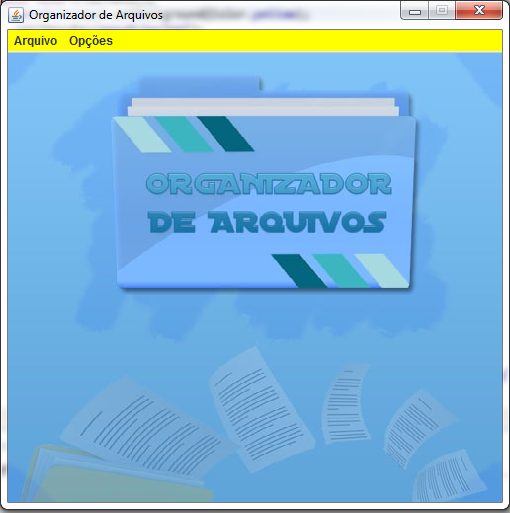
Método lastDiretorios: Cria uma JLabel, acessa o arquivo de texto através do objeto do tipo Dados , recebe o quantas linhas o arquivo de texto possui e mostra na tela as últimas 5 linhas do arquivo de texto se houver 5, se não mostra as que possuem ou nenhuma, posiciona onde deve ficar cade JLabel.

**Class JanelaCreditos:**

Segue o mesmo padrão da Class JanelaDiretorios, os atributos presentes possuem a mesma função e o redirecionamento da tela é igual a dirença é a imagem, pois nesta, é carregado apenas uma imagem e um botão de voltar.

### Manual de Utilização

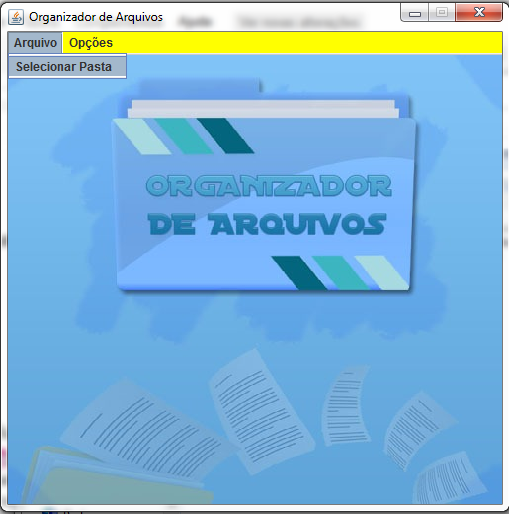
O Organizador de Arquivos tem o objetivo de auxiliar o usuário na organização e sistematização de arquivos. Sendo assim, quando o usuário abrir o aplicativo será exibida a tela inicial:



O menu do programa é composto pelos seguintes componentes Arquivos e Opções.

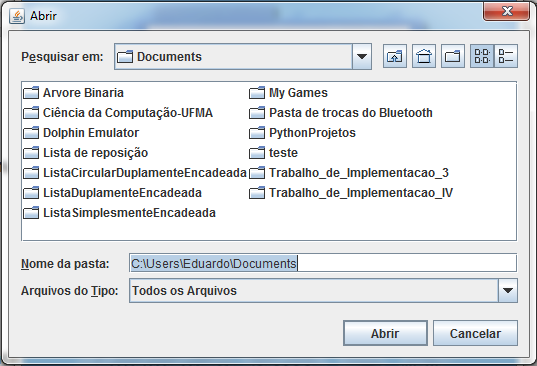
#### Arquivo

Esse é o principal componente do programa, é onde tem a opção para selecionar a pasta para que seja organizada;

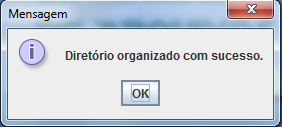


##### 1.1 Selecionar Pasta

Para escolher o diretório que deve ser organizado, o usuário deve clicar em Selecionar Pasta. Assim, aparecerá a seguinte janela:



Então, o usuário escolhe a pasta e clica em abrir. Se o diretório foi organizado, abrirá uma nova janela e será impressa a seguinte mensagem:

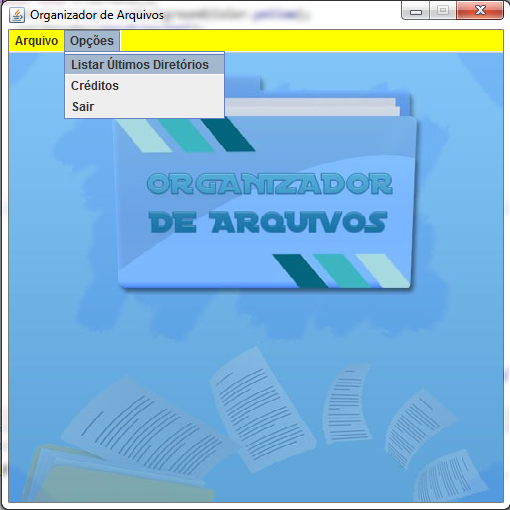


Caso não tenha arquivos no diretório escolhido, não será possível organizar a pasta e imprimirá a seguinte mensagem:

##### 

1. Opções

Contém outras opções do programa como Listar Diretórios, Créditos e Sair;



##### 2.1 Listar Diretórios

Esta opção o usuário tem o controle de ver quais foram os últimos diretórios organizados;

##### 2.2 Créditos

Essa opção mostra os nomes dos desenvolvedores do programa;

##### 2.3 Sair

Essa opção encerra o programa;

### Conclusão

De acordo com o problema levantado (arquivos desorganizados) percebemos a necessidade de criar um organizador de arquivos para facilitar a vida dos usuários na hora de procurar algo especifico, ele não ficará perdendo tanto tempo procurando em uma pasta com diversos tipos de arquivo, basta saber qual o tipo de arquivo que procura e ir na pasta específica, além de economizar tempo, o visual da tela ficará mais elegante.