

# Princípios e Padrões de Projeto

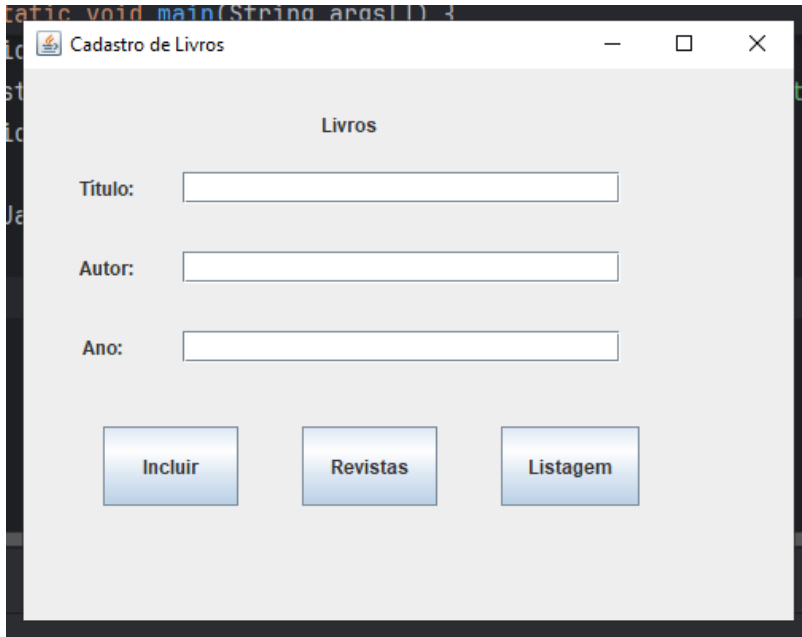
## Prática 01

Eduarda Lopes Santos Moura

12311BCC033

O código-fonte está organizado em 7 classes, sendo elas: Biblioteca, JanelaLista, JanelaLivro, JanelaRevista, Livros, Revistas e Main.

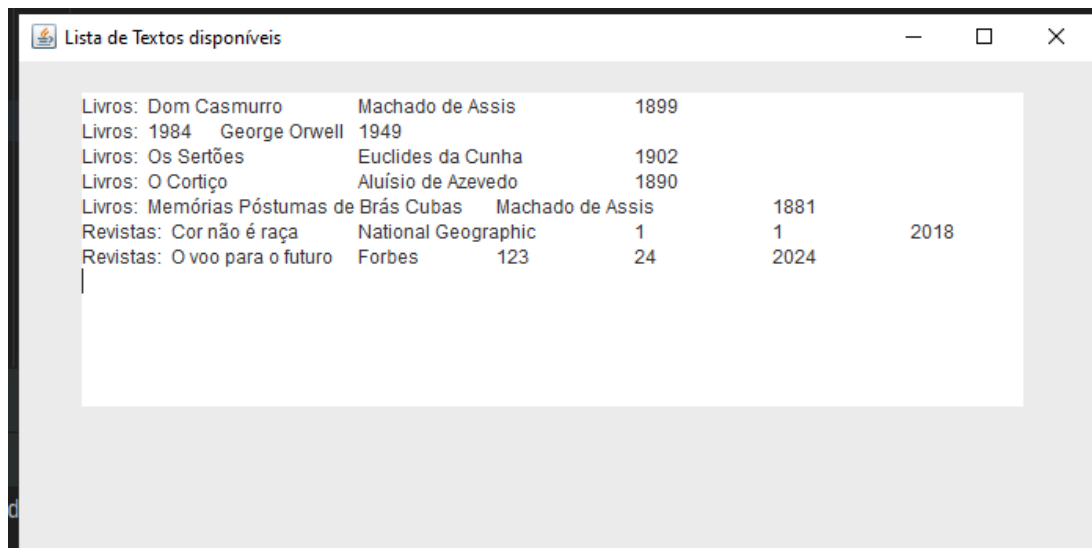
- Execução do aplicativo



The screenshot shows a window titled "Cadastro de Livros". Inside the window, the title "Livros" is centered at the top. Below the title, there are three text input fields labeled "Título:", "Autor:", and "Ano:". At the bottom of the window, there are three buttons: "Incluir", "Revistas", and "Listagem".



The screenshot shows a window titled "Cadastro de revistas". Inside the window, the title "Revistas" is centered at the top. Below the title, there are three text input fields labeled "Título:", "Org.:", and "Vol.:". Below the "Vol.:" field, there are two more text input fields labeled "Nro:" and "Ano:". At the bottom of the window, there are three buttons: "Incluir", "Livros", and "Listagem".



- Pontos a melhorar

De modo geral houve pouca repetição de código, mas isso poderia ter sido otimizado, por exemplo, o método `setAutor` da classe `Livros` é o mesmo que o `setOrganizacao` da classe `Revistas`, ou também os métodos `setNumero` e `setVolume` da classe `Revistas` também são idênticos, o que configura uma repetição de código. Além disso, há como melhorar o tratamento das exceções - poderia ter feito uma comunicação explícita ao usuário do por quê o livro ou revista teve um erro ao ser incluído. As classes das janelas ficaram grandes com repetição de código também por causa do `setBounds`. Por fim, eu poderia ter implementado um sistema que verifica se o livro ou revista já foi cadastrado, o que melhoraria a usabilidade do programa.

## EXPLICAÇÕES E COMENTÁRIOS SOBRE AS CLASSES

- Classe Biblioteca

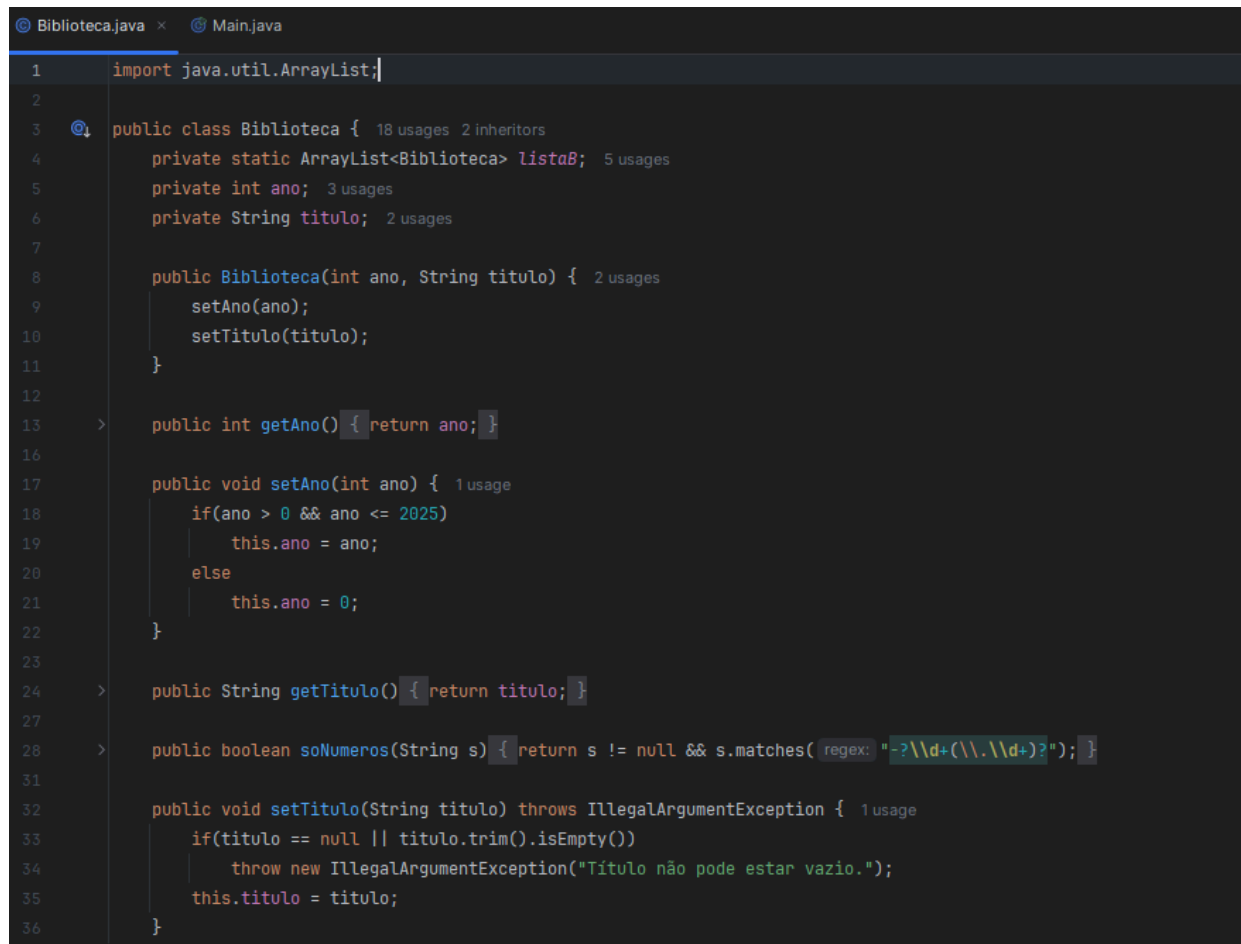
A classe `Biblioteca` possui como atributos as características em comum entre `Livros` e `Revistas`: ano e título, além de que ela guarda uma lista do tipo `ArrayList` que é uma lista que poderá conter livros e também revistas. Para que ela funcione, temos o método estático `criarLista` que pode ser invocado sem estar associado a um objeto na classe que o utiliza, e no caso, ele será invocado na `main` para que seja alocada a memória para a lista e seja retornada uma lista vazia recém-criada. Para popular ela temos o método `addLivro` que adiciona um livro (objeto da classe `Livros` passado no argumento da função) na lista especificada nos atributos e criada com o chamamento da função na `Main`, e o método `addRevista` que adiciona uma revista (objeto da classe `Revistas` passado no argumento da função) seguindo a mesma lógica.

Para o atributo ano temos uma regra de criação de que o ano deve ser maior que 0 e menor ou igual a 2025 que é o ano vigente, e na falha disso, o ano fica zerado. Nessa parte eu poderia ter criado uma exceção própria que seria lançada na falha e

que pedisse para o usuário tentar novamente, para evitar que o ano fosse zerado. Já para o atributo título há uma regra de criação de que não pode ser inserido um espaço vazio, e se for um espaço vazio, uma exceção do tipo `IllegalArgumentException` é lançada.

Por fim, o método `toString` assegura que as informações do livro ou revista serão impressas de acordo com o polimorfismo realizado na leitura do `ArrayList`.

Código-fonte da classe na próxima folha:



```
1  import java.util.ArrayList;
2
3  @ public class Biblioteca { 18 usages 2 inheritors
4      private static ArrayList<Biblioteca> lista8; 5 usages
5      private int ano; 3 usages
6      private String titulo; 2 usages
7
8      public Biblioteca(int ano, String titulo) { 2 usages
9          setAno(ano);
10         setTitulo(titulo);
11     }
12
13     > public int getAno() { return ano; }
14
15
16
17     public void setAno(int ano) { 1 usage
18         if(ano > 0 && ano <= 2025)
19             this.ano = ano;
20         else
21             this.ano = 0;
22     }
23
24     > public String getTitulo() { return titulo; }
25
26
27
28     > public boolean soNumeros(String s) { return s != null && s.matches(regex: "-?\\d+(\\.\\d+)?"); }
29
30
31
32     public void setTitulo(String titulo) throws IllegalArgumentException { 1 usage
33         if(titulo == null || titulo.trim().isEmpty())
34             throw new IllegalArgumentException("Título não pode estar vazio.");
35         this.titulo = titulo;
36     }
```

```

52     public void setTitulo(String titulo) throws IllegalArgumentException { 1 usage
53         if(titulo == null || titulo.trim().isEmpty())
54             throw new IllegalArgumentException("Título não pode estar vazio.");
55         this.titulo = titulo;
56     }
57
58     public static void addLivro(Livros l) { 6 usages
59         if(l != null)
60             listaB.add(l);
61     }
62
63     public static void addRevista(Revistas r) { 3 usages
64         if(r != null)
65             listaB.add(r);
66     }
67
68     public static ArrayList<Biblioteca> criarLista() { 1 usage
69         listaB = new ArrayList<Biblioteca>();
70         return listaB;
71     }
72
73     > public static ArrayList<Biblioteca> getLista() { return listaB; }
74
75     @Override 2 overrides
76     > public String toString() { return ""; }
77
78 }

```

- Classe Livros

Um livro é um objeto de uma biblioteca, e por isso essa classe herda a classe Biblioteca. Na classe Livros temos como atributo o único atributo específico de livro: o autor. Um autor não pode ter um nome só composto de números, controle esse que é realizado pela função `soNumeros` determinada na classe Biblioteca que identifica se uma string é composta só de números pelo padrão regex compatível ao Java. Esse controle é feito por meio do lançamento de uma exceção `IllegalArgumentException`. Há também o método `toString`, que é escrito e formulado dessa maneira com overriding para que ao utilizar o polimorfismo na lista criada pela classe Biblioteca, sejam impressas informações apenas referentes aos livros.

Código fonte da classe:

```

1 public class Livros extends Biblioteca { 13 usages
2     private String autor; 2 usages
3
4     public Livros(int ano, String titulo, String autor) { 6 usages
5         super(ano, titulo);
6         setAutor(autor);
7     }
8
9     public String getAutor() { return autor; }
10
11
12
13     public void setAutor(String autor) throws IllegalArgumentException { 1 usage
14         if(autor == null || super.soNumeros(autor) || autor.trim().isEmpty())
15             throw new IllegalArgumentException("Nome do autor não pode ser vazio ou só números.");
16         this.autor = autor;
17     }
18
19     @Override
20     public String toString() { return "Livros: " + getTitulo() + "\t" + getAutor() + "\t" + getAno(); }
21
22
23 }

```

- Classe Revistas

Seguindo a lógica dos livros, uma revista também é um objeto de uma biblioteca e por isso essa classe também herda a classe Biblioteca. Nessa classe temos os atributos específicos de uma revista: organização, volume e número. A organização está restrita da mesma forma que o autor nos livros: controlada por meio da função soNumeros e pela condição de não ser null ou um espaço em branco, gerando uma exceção IllegalArgumentException ao falhar. Também temos volume e número restringidos da mesma maneira: se o número for menor que 0 ou 0, o atributo será setado em zero. Usando o polimorfismo da mesma maneira, temos também o método toString com overriding para fazer a impressão da forma correta.

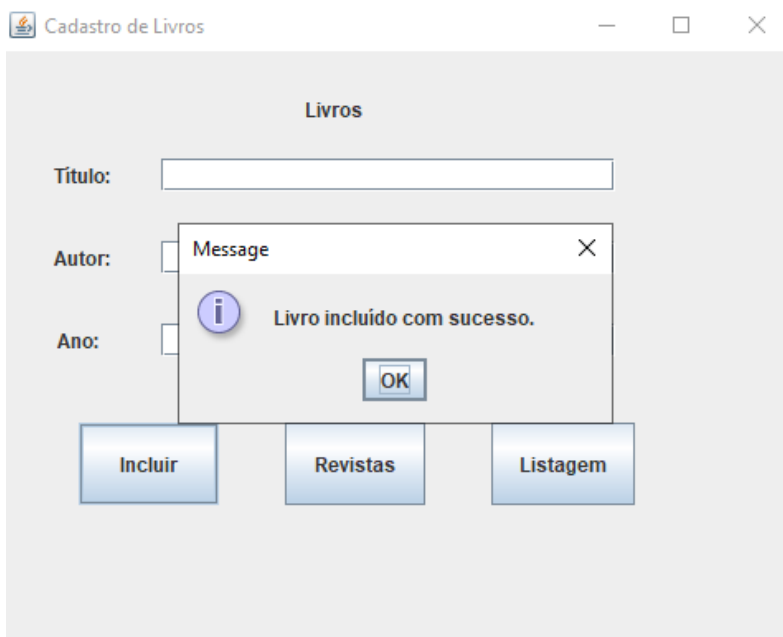
Código-fonte da classe Revistas na próxima página:

```
© Biblioteca.java  © Livros.java  © Revistas.java x  © Main.java
1 public class Revistas extends Biblioteca { 7 usages
2     private String organizacao; 2 usages
3     private int volume; 3 usages
4     private int numero; 3 usages
5
6     public Revistas(int ano, String titulo, String organizacao, int volume, int numero) { 3 usages
7         super(ano, titulo);
8         setOrganizacao(organizacao);
9         setVolume(volume);
10        setNumero(numero);
11    }
12
13    > public String getOrganizacao() { return organizacao; }
14
15
16
17    public void setOrganizacao(String organizacao) throws IllegalArgumentException { 1 usage
18        if(organizacao == null || super.soNumeros(organizacao) || organizacao.trim().isEmpty())
19            throw new IllegalArgumentException("Organização não pode estar vazia ou ser só números.");
20        this.organizacao = organizacao;
21    }
22
23    > public int getVolume() { return volume; }
24
25
26
27    public void setVolume(int volume) { 1 usage
28        if(volume > 0)
29            this.volume = volume;
30        else
31            this.volume = 0;
32    }
33
34    > public int getNumero() { return numero; }
```

```
© Biblioteca.java  © Livros.java  © Revistas.java x  © Main.java
1 public class Revistas extends Biblioteca { 7 usages
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23    > public int getVolume() { return volume; }
24
25
26
27    public void setVolume(int volume) { 1 usage
28        if(volume > 0)
29            this.volume = volume;
30        else
31            this.volume = 0;
32    }
33
34    > public int getNumero() { return numero; }
35
36
37
38    public void setNumero(int numero) { 1 usage
39        if(numero > 0)
40            this.numero = numero;
41        else
42            this.numero = 0;
43    }
44
45    @Override
46    public String toString() {
47        return "Revistas: " + getTitulo() + "\t" + getOrganizacao() + "\t" + getVolume() + "\t" + getNumero() + "\t" + getAno();
48    }
49 }
```

- Classe JanelaLivro

Conforme explicado em sala de aula, utilizei a tecnologia Swing para construir a janela dos livros, com os labels, text fields e botões conectados no painel que por sua vez é conectado com a janela. Fiz também a regulagem do alinhamento de forma manual utilizando o método `setBounds` e colocando as coordenadas. Por meio do `ActionListener` foram configurados os botões incluir, revistas e listagem. No botão incluir é feita a coleta dos textos digitados nos textfields e um livro é criado como objeto e então é adicionado por meio do o método estático `addLivro` pela classe `Biblioteca`, e se foi feito com sucesso é mostrada a mensagem de sucesso e caso não, é mostrado uma mensagem de erro. Exemplificando a inclusão de um livro com sucesso:



Exemplificando a inclusão de um livro com erro:



Cadastro de Livros

Livros

Título:

Autor:

Ano:

Incluir Revistas Listagem

Não possui título, portanto será lançada uma exceção e a inclusão não acontecerá, e então a mensagem de erro será mostrada.



Cadastro de Livros

Livros

Título:

Autor:

Ano:

Incluir Revistas Listagem

Message

Erro ao incluir livro.

OK

Também há o botão listagem que leva para a JanelaLista que listará todos os livros e revistas existentes na Biblioteca. Por fim, o botão de revistas leva para a JanelaRevista que possibilita o cadastro de uma revista.



Código-fonte nas páginas a seguir:

```
© Biblioteca.java  © Livros.java  © Revistas.java  © JanelaLivro.java ×  © Main.java

1  > import ...
4
5  public class JanelaLivro extends JFrame { 3 usages
6  public JanelaLivro() { 3 usages
7      JFrame janela = new JFrame( title: "Cadastro de Livros");
8
9      JPanel painel = new JPanel();
10
11      JLabel labelLivros = new JLabel( text: "Livros");
12      JLabel labelTitulo = new JLabel( text: "Título: ");
13      JLabel labelAutor = new JLabel( text: "Autor: ");
14      JLabel labelAno = new JLabel( text: "Ano: ");
15
16      JTextField campotitulo = new JTextField( columns: 25);
17      JTextField campoautor = new JTextField( columns: 25);
18      JTextField campoano = new JTextField( columns: 4);
19
20      JButton botoaincluir = new JButton( text: "Incluir");
21      JButton botaorevistas = new JButton( text: "Revistas");
22      JButton botoalista = new JButton( text: "Listagem");
23
24      painel.add(labelLivros);
25      painel.add(labelTitulo);
26      painel.add(campotitulo);
27      painel.add(labelAutor);
28      painel.add(campoautor);
29      painel.add(labelAno);
30      painel.add(campoano);
31      painel.add(botoaincluir);
32      painel.add(botaorevistas);
```

```

5      public class JanelaLivro extends JFrame { 3 usages
6      public JanelaLivro() { 3 usages
32         painel.add(botaorevistas);
33         painel.add(botaolista);
34
35         janela.getContentPane().add(painel);
36         janela.setVisible(true);
37         janela.setSize( width: 500, height: 400);
38
39         labelLivros.setBounds( x: 187, y: 10, width: 250, height: 50);
40         labelTitulo.setBounds( x: 35, y: 50, width: 250, height: 50);
41         labelAutor.setBounds( x: 35, y: 100, width: 250, height: 50);
42         labelAno.setBounds( x: 37, y: 150, width: 250, height: 50);
43
44         campotitulo.setBounds( x: 100, y: 65, width: 275, height: 20);
45         campoautor.setBounds( x: 100, y: 115, width: 275, height: 20);
46         campoano.setBounds( x: 100, y: 165, width: 275, height: 20);
47
48         botaoincluirl.setBounds( x: 50, y: 225, width: 85, height: 50);
49         botaorevistas.setBounds( x: 175, y: 225, width: 85, height: 50);
50         botaolista.setBounds( x: 300, y: 225, width: 87, height: 50);
51
52         painel.setLayout(null);
53
54         botaoincluirl.addActionListener(new ActionListener() {
55             public void actionPerformed(ActionEvent e) {
56                 try {
57                     String titulo = campotitulo.getText();
58                     String autor = campoautor.getText();
59                     int ano = Integer.parseInt(campoano.getText());

```

```
Biblioteca.java  Livros.java  Revistas.java  JanelaLivro.java x  Main.java

5  public class JanelaLivro extends JFrame { 3 usages
6  public JanelaLivro() { 3 usages
54  botoaincluir.addActionListener(new ActionListener() {
55      public void actionPerformed(ActionEvent e) {
61          Livros l = new Livros(ano, titulo, autor);
62
63          Biblioteca.addLivro(l);
64
65          campotitulo.setText("");
66          campoautor.setText("");
67          campoano.setText("");
68
69          JOptionPane.showMessageDialog(janela, message: "Livro incluído com sucesso.");
70      } catch (Exception e1) {
71          JOptionPane.showMessageDialog(janela, message: "Erro ao incluir livro.");
72      }
73  }
74  });
75
76  botoarevistas.addActionListener(new ActionListener() {
77      public void actionPerformed(ActionEvent e) {
78          new JanelaRevista();
79          janela.setVisible(false);
80      }
81  });
82
83  botoalista.addActionListener(new ActionListener() {
84      public void actionPerformed(ActionEvent e) {
85          new JanelaLista();
86          janela.setVisible(false);
87      }
88  });
89  }
90  }
```

- Classe JanelaRevista

Segue a mesma lógica da janela de livros: só possui os atributos diferentes, pois será cadastrada uma revista. Assim, quando é pressionado o botão incluir, o ActionListener performa a ação de criação de uma revista a partir dos dados

digitados nos textfields da classe, se é bem sucedido há uma mensagem informando e se há erro também há uma mensagem informando o erro. Os outros botões levam para as janelas de livros ou de listagem e se caso o usuário feche a janela de revistas, o programa volta para a janela de livros por meio do WindowListener que funciona de forma semelhante ao Action.

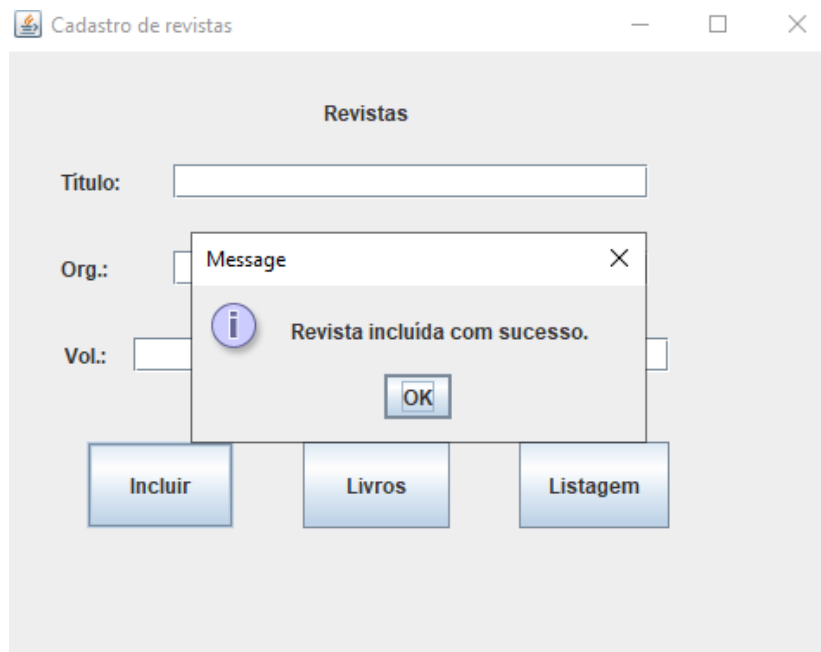
Exemplificando a inclusão bem sucedida de uma revista:



The screenshot shows a window titled "Cadastro de revistas" with a subtitle "Revistas". It contains several text input fields and three buttons. The fields are filled with the following data:

Field	Value
Título:	Eles sabiam de tudo
Org.:	Veja
Vol.:	23
Nro:	2
Ano:	2014

Below the fields are three buttons: "Incluir", "Livros", and "Listagem". The "Incluir" button is highlighted, indicating it was just clicked.



This screenshot shows the same "Cadastro de revistas" window, but with a message dialog box overlaid. The dialog box is titled "Message" and contains the text "Revista incluída com sucesso." (Magazine included successfully). There is an "OK" button at the bottom of the dialog. The background window is slightly dimmed, and the "Incluir" button remains highlighted.

Agora, tentarei colocar apenas números na organização:

Cadastro de revistas

Revistas

Título:

Org.:

Vol.:  Nro:  Ano:

O programa deverá lançar uma exceção e comunicar o erro na inclusão, e assim:

Cadastro de revistas


Revistas

Título:

Org.:

Vol.:

Message

 Erro ao incluir revista.

## Código-fonte da classe a seguir:

```
1 > import ...
6
7 public class JanelaRevista extends JFrame { 1 usage
8     public JanelaRevista() { 1 usage
9         JFrame janelaR = new JFrame( title: "Cadastro de revistas");
10
11         JPanel painelR = new JPanel();
12
13         JLabel labelRevistas = new JLabel( text: "Revistas");
14         JLabel labelTituloR = new JLabel( text: "Título:");
15         JLabel labelOrgR = new JLabel( text: "Org.:");
16         JLabel labelVolR = new JLabel( text: "Vol.:");
17         JLabel labelNroR = new JLabel( text: "Nro:");
18         JLabel labelAnoR = new JLabel( text: "Ano:");
19
20         JTextField campoTituloR = new JTextField( columns: 25);
21         JTextField campoOrgR = new JTextField( columns: 25);
22         JTextField campoVolR = new JTextField( columns: 3);
23         JTextField campoNroR = new JTextField( columns: 3);
24         JTextField campoAnoR = new JTextField( columns: 4);
25
26         JButton botaoIncluir = new JButton( text: "Incluir");
27         JButton botaoLivros = new JButton( text: "Livros");
28         JButton botaoLista = new JButton( text: "Listagem");
29
30         painelR.add(labelRevistas);
31         painelR.add(labelTituloR);
32         painelR.add(campoTituloR);
33         painelR.add(labelOrgR);
34         painelR.add(campoOrgR);
```

⌕ Biblioteca.java   ⌕ Livros.java   ⌕ Revistas.java   ⌕ JanelaLivro.java   ⌕ JanelaRevista.java ×   ⌕ JanelaLista.java   ⌕ Main.java

```
7 public class JanelaRevista extends JFrame { 1 usage
8     public JanelaRevista() { 1 usage
34         painelR.add(campoOrgR);
35         painelR.add(labelVolR);
36         painelR.add(campoVolR);
37         painelR.add(labelNroR);
38         painelR.add(campoNroR);
39         painelR.add(labelAnoR);
40         painelR.add(campoAnoR);
41         painelR.add(botaoIncluir);
42         painelR.add(botaoLivros);
43         painelR.add(botaoLista);
44
45         janelaR.getContentPane().add(painelR);
46         janelaR.setVisible(true);
47         janelaR.setSize(width: 500, height: 400);
48
49         labelRevistas.setBounds(x: 187, y: 10, width: 250, height: 50);
50         labelTituloR.setBounds(x: 35, y: 50, width: 250, height: 50);
51         labelOrgR.setBounds(x: 35, y: 100, width: 250, height: 50);
52         labelVolR.setBounds(x: 37, y: 150, width: 25, height: 50);
53         labelNroR.setBounds(x: 177, y: 150, width: 25, height: 50);
54         labelAnoR.setBounds(x: 297, y: 150, width: 25, height: 50);
55
56         campoTituloR.setBounds(x: 100, y: 65, width: 275, height: 20);
57         campoOrgR.setBounds(x: 100, y: 115, width: 275, height: 20);
58         campoVolR.setBounds(x: 77, y: 165, width: 50, height: 20);
59         campoNroR.setBounds(x: 217, y: 165, width: 50, height: 20);
60         campoAnoR.setBounds(x: 337, y: 165, width: 50, height: 20);
61     }
```

Ⓢ Biblioteca.java   Ⓢ Livros.java   Ⓢ Revistas.java   Ⓢ JanelaLivro.java   Ⓢ JanelaRevista.java ×   Ⓢ JanelaLista.java   Ⓢ Main.java

```
7 public class JanelaRevista extends JFrame { 1 usage
8 public JanelaRevista() { 1 usage
60 campoAnoR.setBounds( x: 337, y: 165, width: 50, height: 20);
61
62 botaoIncluir.setBounds( x: 50, y: 225, width: 85, height: 50);
63 botaoLivros.setBounds( x: 175, y: 225, width: 85, height: 50);
64 botaoLista.setBounds( x: 300, y: 225, width: 87, height: 50);
65
66 painelR.setLayout(null);
67
68 botaoIncluir.addActionListener(new ActionListener() {
69     public void actionPerformed(ActionEvent e) {
70         try {
71             String titulo = campoTituloR.getText();
72             String org = campoOrgR.getText();
73             int vol = Integer.parseInt(campoVolR.getText());
74             int nro = Integer.parseInt(campoNroR.getText());
75             int ano = Integer.parseInt(campoAnoR.getText());
76
77             Revistas r = new Revistas(ano, titulo, org, vol, nro);
78
79             Biblioteca.addRevista(r);
80
81             campoTituloR.setText("");
82             campoOrgR.setText("");
83             campoVolR.setText("");
84             campoNroR.setText("");
85             campoAnoR.setText("");
86
87             JOptionPane.showMessageDialog(janelaR, message: "Revista incluída com sucesso.");
```



```
Biblioteca.java  Livros.java  Revistas.java  JanelaLivro.java  JanelaRevista.java  JanelaLista.java  Main.java

7  public class JanelaRevista extends JFrame { 1 usage
8      public JanelaRevista() { 1 usage
68          botaoIncluir.addActionListener(new ActionListener() {
69              public void actionPerformed(ActionEvent e) {
86
87                  JOptionPane.showMessageDialog(janelaR, message: "Revista incluída com sucesso.");
88              } catch (Exception e1) {
89                  JOptionPane.showMessageDialog(janelaR, message: "Erro ao incluir revista.");
90              }
91          }
92      });
93
94      botaoLivros.addActionListener(new ActionListener() {
95          public void actionPerformed(ActionEvent e) {
96              new JanelaLivro();
97              janelaR.setVisible(false);
98          }
99      });
100
101      botaoLista.addActionListener(new ActionListener() {
102          public void actionPerformed(ActionEvent e) {
103              new JanelaLista();
104              janelaR.setVisible(false);
105          }
106      });
107
108      janelaR.addWindowListener(new WindowAdapter() {
109          public void windowClosing(WindowEvent e) {
110              new JanelaLivro();
111          }
112      });
113  }
114 }
```

- Classe JanelaLista

Nessa classe será mostrados todos os objetos presentes no array criado pela classe Biblioteca na Main, então há basicamente uma textArea que mostrará os itens cadastrados por meio do enhanced for que percorre toda a lista imprimindo o que está no método toString do objeto em questão por meio do comando append próprio

da textArea. Se a janela é fechada, a janela inicial dos livros é aberta automaticamente.

Código-fonte:

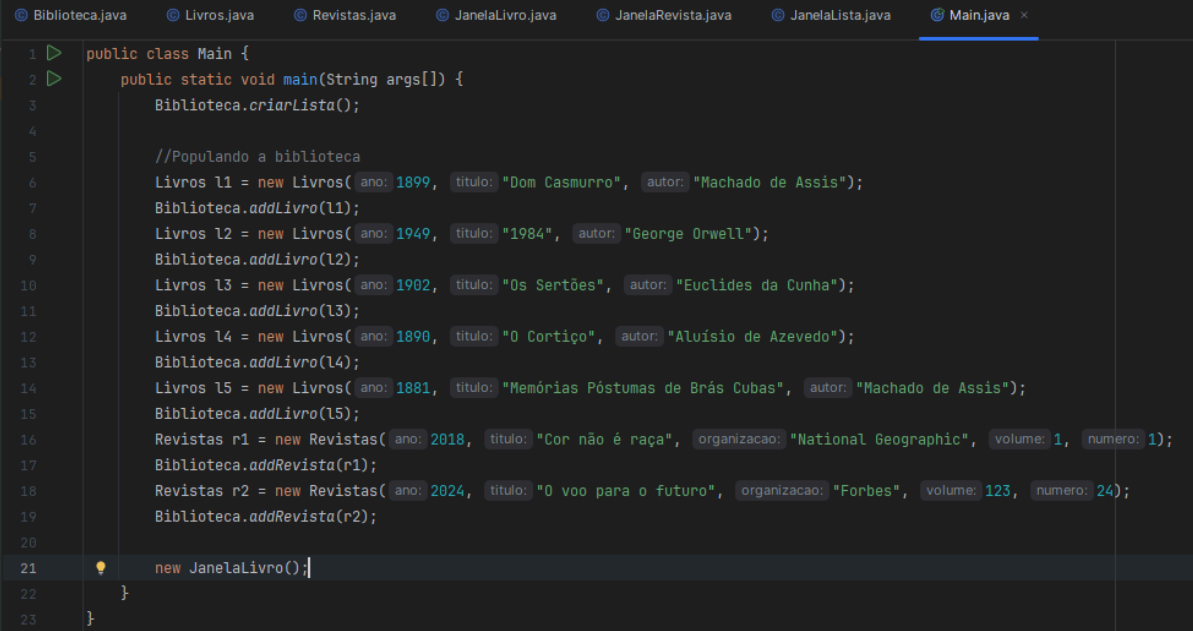
```
1  > import ...
5
6  public class JanelaLista { 2 usages
7      public JanelaLista() { 2 usages
8          JFrame janelaL = new JFrame( title: "Lista de Textos disponíveis");
9
10         JPanel painell = new JPanel();
11
12         JLabel labellista = new JLabel( text: "Listagem");
13
14         JTextArea info = new JTextArea();
15
16         String textoObj;
17
18         ArrayList<Biblioteca> listaB = Biblioteca.getList();
19
20         painell.add(info);
21         painell.setLayout(null);
22
23         janelaL.getContentPane().add(painell);
24         janelaL.setVisible(true);
25         janelaL.setSize( width: 700, height: 350);
26
27         info.setBounds( x: 40, y: 20, width: 600, height: 200);
28
29         for (Biblioteca biblioteca : listaB) {
30             textoObj = biblioteca.toString();
31             info.append(textoObj + "\n");
32         }
33
34
35
36
37
38
39
40 }
```

```
29      for (Biblioteca biblioteca : listaB) {
30          textoObj = biblioteca.toString();
31          info.append(textoObj + "\n");
32      }
33
34      janelaL.addWindowListener(new WindowAdapter() {
35          public void windowClosing(WindowEvent e) {
36              new JanelaLivro();
37          }
38      });
39
40 }
```

- Classe Main

Por fim, essa classe contém a execução do programa, tendo como único método o main que primeiramente cria a lista da Biblioteca por meio da função estática criarLista, depois popula a biblioteca com alguns livros e revistas que criei a fim de testar a aplicação e por fim chama a janelaLivro que é a janela principal e inicial do programa.

Código-fonte:



```
1 public class Main {
2     public static void main(String args[]) {
3         Biblioteca.criarLista();
4
5         //Populando a biblioteca
6         Livros l1 = new Livros(ano: 1899, titulo: "Dom Casmurro", autor: "Machado de Assis");
7         Biblioteca.addLivro(l1);
8         Livros l2 = new Livros(ano: 1949, titulo: "1984", autor: "George Orwell");
9         Biblioteca.addLivro(l2);
10        Livros l3 = new Livros(ano: 1902, titulo: "Os Sertões", autor: "Euclides da Cunha");
11        Biblioteca.addLivro(l3);
12        Livros l4 = new Livros(ano: 1890, titulo: "O Cortiço", autor: "Aluísio de Azevedo");
13        Biblioteca.addLivro(l4);
14        Livros l5 = new Livros(ano: 1881, titulo: "Memórias Póstumas de Brás Cubas", autor: "Machado de Assis");
15        Biblioteca.addLivro(l5);
16        Revistas r1 = new Revistas(ano: 2018, titulo: "Cor não é raça", organizacao: "National Geographic", volume: 1, numero: 1);
17        Biblioteca.addRevista(r1);
18        Revistas r2 = new Revistas(ano: 2024, titulo: "O voo para o futuro", organizacao: "Forbes", volume: 123, numero: 24);
19        Biblioteca.addRevista(r2);
20
21        new JanelaLivro();
22    }
23 }
```