

### **Engenharia da Computação – 3ª série**

## **Manipulação de Ícones, Senhas e Datas** **(L1/1, L2/1 e L3/1)**

**2025**

# ECM251 – Linguagens de Programação I

## Aula 14 – L1/1, L2/1 e L3/1

### Horário

Terça-feira: 2 x 2 aulas/semana

- L1/1 (07h40min-09h20min): *Prof. Calvetti*;
- L1/2 (09h30min-11h10min): *Prof. Calvetti*;
- L2/1 (07h40min-09h20min): *Prof. Menezes*;
- L2/2 (11h20min-13h00min): *Prof. Calvetti*;
- L3/1 (09h30min-11h10min): *Prof. Evandro*;
- L3/2 (11h20min-13h00min): *Prof. Evandro*.

# ECM251 – Linguagens de Programação I

## Manipulação de Ícones, Senhas e Datas

### Tópico

- Listas *drop-down*

## Lista drop-down

### Definição



- Uma caixa de combinação, ou lista *drop-down*, fornece uma lista de itens a partir da qual o usuário pode fazer uma única seleção;
- As caixas de combinação são implementadas com a classe **JComboBox**, que estende a classe **JComponent**;
- *JComboBoxes* geram *ItemEvents* como *JCheckBoxes* e *JRadioButtons*;
- O exemplo a seguir demonstra uma forma especial de classe interna que costuma ser utilizada no tratamento de evento.

## Lista drop-down

### Exemplo



```
1 // Usando uma JComboBox para selecionar uma imagem a ser mostrada.
2 import java.awt.FlowLayout;
3 import java.awt.event.ItemListener;
4 import java.awt.event.ItemEvent;
5 import javax.swing.JFrame;
6 import javax.swing.JLabel;
7 import javax.swing.JComboBox;
8 import javax.swing.Icon;
9 import javax.swing.ImageIcon;
10
11 public class ComboBoxFrame extends JFrame
12 { private JComboBox imagesJComboBox; // combobox para armazenar os nomes dos icones
13   private JLabel label; // label para mostrar icone selecionado
14   private String names[] = { "bug1.gif", "bug2.gif", "travelbug.gif", "buganim.gif" };
15   private Icon icons[] = { new ImageIcon( getClass().getResource( names[ 0 ] ) ),
16                             new ImageIcon( getClass().getResource( names[ 1 ] ) ),
17                             new ImageIcon( getClass().getResource( names[ 2 ] ) ),
18                             new ImageIcon( getClass().getResource( names[ 3 ] ) )
19   };
16
```

## Lista drop-down

### Exemplo



```
20 // Construtor do ComboBoxFrame adiciona JComboBox ao JFrame
21 public ComboBoxFrame()
22 { super( "Testando JComboBox" );
23   setLayout( new FlowLayout() ); // especifica um layout
24   imagesJComboBox = new JComboBox( names ); // Programa JComboBox
25   imagesJComboBox.setMaximumRowCount( 3 ); // Mostra 3 linhas
26   imagesJComboBox.addItemListener
27   ( new ItemListener()
28     { // Manipula o evento da JComboBox
29       public void itemStateChanged( ItemEvent event )
30       { // Determine se check box foi selecionado
31         if ( event.getStateChange() == ItemEvent.SELECTED )
32         { label.setIcon( icons[ imagesJComboBox.getSelectedIndex() ] );
33         }
34       }
35     }
36   );
37   add( imagesJComboBox ); // adiciona o combobox ao JFrame
38   label = new JLabel( icons[ 0 ] ); // apresenta o 1o ícone
39   add( label ); // adiciona o label ao JFrame
40 }
41 }
```

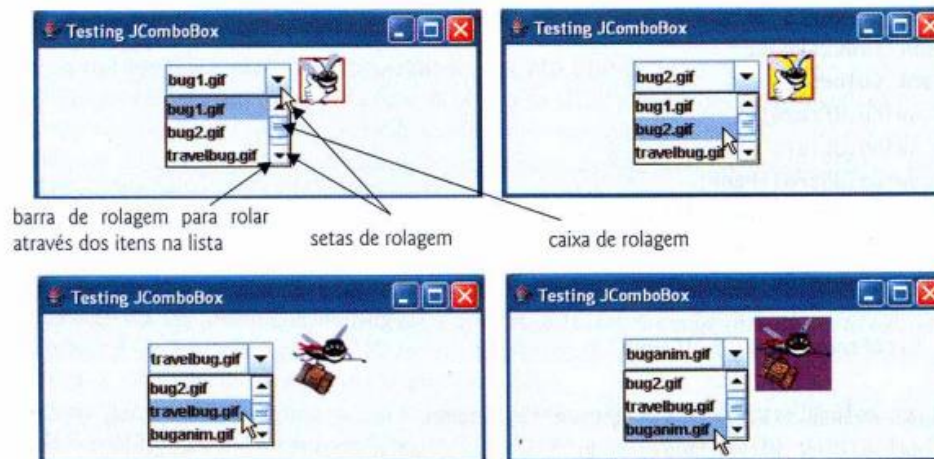
# ECM251 – Linguagens de Programação I

## Lista drop-down

### Exemplo



```
1 // Testando a JComboBoxFrame.  
2 import javax.swing.JFrame;  
3 public class JComboBoxTest  
4 { public static void main( String args[] )  
5   { JComboBoxFrame comboBoxFrame = new JComboBoxFrame();  
6     comboBoxFrame.setDefaultCloseOperation( JFrame.EXIT_ON_CLOSE );  
7     comboBoxFrame.setSize( 350, 150 ); // configura o tamanho do frame  
8     comboBoxFrame.setVisible( true ); // apresenta o frame  
9   }  
10 }
```





### Tópico

- Demonstrando *JTextField* e *JPasswordField*



## Demonstrando JTextField e JPasswordField

### Definição



- Os componentes GUI denominados **JTextField** e **JPasswordField** geram eventos quando o usuário neles digitar os dados desejados e depois pressionar a tecla *Enter*;
- A classe **JTextField** estende a classe **JTextComponent**, do pacote *javax.swing.text*, que possui muitos recursos comuns com os componentes baseados em texto do *Swing*;
- A classe **JPasswordField** estende **JTextField** e adiciona vários métodos específicos do processamento de senhas;
- O exemplo a seguir utiliza as classes **JTextField** e **JPasswordField** para criar e manipular quatro campos de texto.

# ECM251 – Linguagens de Programação I

## Demonstrando JTextField e JPasswordField

### Exemplo



```
1 // Demonstrando a classe JTextField
2 import java.awt.FlowLayout;
3 import java.awt.event.ActionListener;
4 import java.awt.event.ActionEvent;
5 import javax.swing.JFrame;
6 import javax.swing.JTextField;
7 import javax.swing.JPasswordField;
8 import javax.swing.JOptionPane;
9
10 public class TextFieldFrame extends JFrame
11 {   private JTextField textField1; // Campo texto com tamanho
12     private JTextField textField2; // Campo texto com texto
13     private JTextField textField3; // Campo texto com texto e tamanho
14     private JPasswordField passwordField; // Campo senha com texto
```

# ECM251 – Linguagens de Programação I

## Demonstrando JTextField e JPasswordField

### Exemplo



```
15 // Construtor TextFieldFrame adiciona JTextFields ao JFrame
16 public TextFieldFrame()
17 { super( "JTextField e JPasswordField" );
18   setLayout( new FlowLayout() ); // configura layout do frame
19   // Campo texto com 10 colunas
20   textField1 = new JTextField( 10 );
21   add( textField1 ); // adiciona textField1 ao JFrame
22   // Campo texto com texto default
23   textField2 = new JTextField( "Entre com o texto aqui" );
24   add( textField2 ); // adiciona textField2 ao JFrame
25   // Campo texto com texto default e 21 colunas
26   textField3 = new JTextField( "Campo Texto nao editavel", 21 );
27   textField3.setEditable( false ); // desabilita edicao
28   add( textField3 ); // adiciona textField3 ao JFrame
29   // Campo senha com texto default
30   passwordField = new JPasswordField( "Texto oculto" );
31   add( passwordField ); // adiciona passwordField ao JFrame
32   // Manipuladores de eventos
33   TextFieldHandler handler = new TextFieldHandler();
34   textField1.addActionListener( handler );
35   textField2.addActionListener( handler );
36   textField3.addActionListener( handler );
37   passwordField.addActionListener( handler );
38 }
39
```

# ECM251 – Linguagens de Programação I

## Demonstrando JTextField e JPasswordField

### Exemplo



```
40 private class TextFieldHandler implements ActionListener
41 { // processa eventos do campo texto
42     public void actionPerformed((ActionEvent event) )
43     { String string = ""; // declara string ao display
44       // usuario pressionou Enter no JTextField textField1
45       if ( event.getSource() == textField1 )
46           string = String.format( "textField1: %s", event.getActionCommand() );
47       // usuario pressionou Enter no JTextField textField2
48       else if ( event.getSource() == textField2 )
49           string = String.format( "textField2: %s", event.getActionCommand() );
50       // usuario pressionou Enter no JTextField textField3
51       else if ( event.getSource() == textField3 )
52           string = String.format( "textField3: %s", event.getActionCommand() );
53       // usuario pressionou Enter no JTextField passwordField
54       else if ( event.getSource() == passwordField )
55           string = String.format(
56               "passwordField: %s", new String( passwordField.getPassword() )
57           );
58       // apresenta conteúdo do JTextField
59       JOptionPane.showMessageDialog( null, string );
60     }
61 }
62 }
```

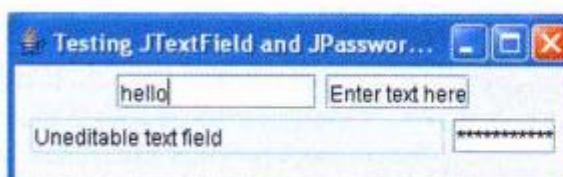
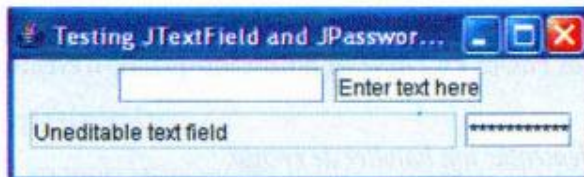
# ECM251 – Linguagens de Programação I

## Demonstrando JTextField e JPasswordField

### Exemplo



```
1 // Testando TextFieldFrame
2 import javax.swing.JFrame;
3
4 public class TextFieldTest
5 {   public static void main( String args[] )
6     {   TextFieldFrame textFieldFrame = new TextFieldFrame();
7         textFieldFrame.setDefaultCloseOperation( JFrame.EXIT_ON_CLOSE );
8         textFieldFrame.setSize( 325, 125 ); // tamanho do frame (L, H)
9         textFieldFrame.setVisible( true ); // apresenta frame
10    }
11 }
```



### Tópico

- Testando *JTable*



## Testando JTable

### Definição



- O componente GUI denominado **JTable**, integrante da classe **JFrame**, gera uma tabela preenchida por uma matriz de *String*, capaz de apresentar as informações dinâmica e eficientemente através de linhas por colunas;
- A tabela poderá ser preenchida pelo usuário ou lida de um banco de dados, por exemplo;
- O exemplo a seguir utiliza a classe **JTable** para criar e manipular uma tabela (matriz) de *Strings*.



## Testando JTable

### Exemplo



```
1 // Testando JTable
2 import java.awt.GridLayout;
3 import javax.swing.JFrame;
4 import javax.swing.JPanel;
5 import javax.swing.JScrollPane;
6 import javax.swing.JTable;
7
8 public class PhoneBook extends JFrame
9 {
10     JPanel background;
11     JTable table;
12     JScrollPane bar;
13     Object [][] data =
14     {
15         {"20080001", "Antonio", "11 98888-0001", "antonio@gmail.com"},
16         {"20080002", "José", "11 98111-3333", "jose@hotmail.com"},
17         {"20080003", "Ricardo", "13 9876-5432", "ricardo@bol.com.br"},
18         {"20080004", "Roberto", "11 98765-4321", "roberto@gmail.com"},
19         {"20080005", "Valter", "11 2666-6666", "valter@yahoo.com.br"}
20     };
21
22     String [] column = {"RA", "Nome", "Telefone", "Email"};
23
24     public PhoneBook()
25     { super ("Contatos");
26     }
```

## Testando JTable

### Exemplo



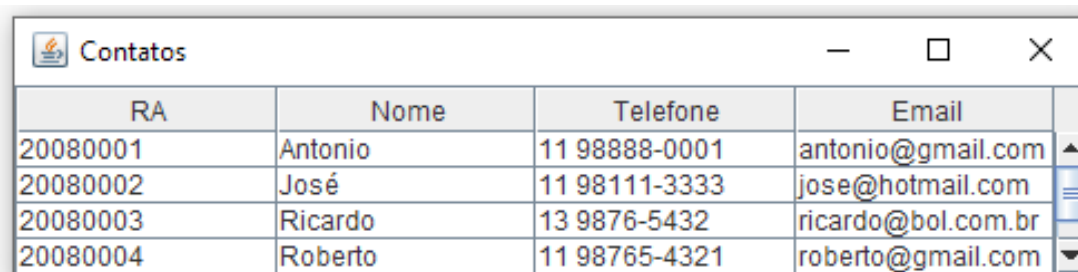
```
26
27     public void makeWindow()
28     {   background = new JPanel();
29         background.setLayout(new GridLayout(1, 1));
30         table = new JTable(data, column);
31         bar = new JScrollPane(table);
32         background.add(bar);
33         getContentPane().add(background);
34         setDefaultCloseOperation(EXIT_ON_CLOSE);
35         setSize(500, 125);
36         setVisible(true);
37     }
38 }
```

## Testando JTable

### Exemplo



```
1 // Testando PhoneBook
2 public class PhoneBookTest
3 {   public static void main(String[] args)
4     {   PhoneBook pb1 = new PhoneBook();
5         pb1.makeWindow();
6     }
7 }
```



RA	Nome	Telefone	Email
20080001	Antonio	11 98888-0001	antonio@gmail.com
20080002	José	11 98111-3333	jose@hotmail.com
20080003	Ricardo	13 9876-5432	ricardo@bol.com.br
20080004	Roberto	11 98765-4321	roberto@gmail.com

### Tópico

- A classe *Calendar*

## A classe Calendar

### Definição



- A classe **Calendar** pode produzir os valores de todos os campos de calendário necessários para implementar a formatação de data e hora, para um determinado idioma e estilo de calendário;
- Por exemplo, Japonês, Inglês/USA, Italiano, Português/Brasil entre outros;
- A classe **Calendar** é a mais usada quando se trata de datas, mas como é uma classe abstrata, não pode ser instanciada, portanto, para obter um calendário é necessário usar o método estático *getInstance( )*, apresentado no exemplo a seguir.

# ECM251 – Linguagens de Programação I

## A classe Calendar

### Exemplo



```
1 import java.util.Calendar;
2 public class DateCalendarTest
3 { public static void main(String[] args)
4   { Calendar c = Calendar.getInstance();
5     System.out.println("Data e Hora atual: " + c.getTime());
6     System.out.println("Ano: " + c.get(Calendar.YEAR));
7     System.out.println("Mês: " + (c.get(Calendar.MONTH) + 1));
8     System.out.println("Dia do Mês: " + c.get(Calendar.DAY_OF_MONTH));
9
10    c.set(Calendar.YEAR, 1963);
11    c.set(Calendar.MONTH, Calendar.MARCH);
12    c.set(Calendar.DAY_OF_MONTH, 8);
13    System.out.println("\nData reprogramada e hora atual: " + c.getTime());
14    System.out.println("Ano: " + c.get(Calendar.YEAR));
15    System.out.println("Mês: " + (c.get(Calendar.MONTH) + 1));
16    System.out.println("Dia do Mês: " + c.get(Calendar.DAY_OF_MONTH));
17  }
```

# ECM251 – Linguagens de Programação I

## A classe Calendar

### Exemplo



```
18     c = Calendar.getInstance();
19     System.out.println("\nData e Hora atual: " + c.getTime());
20     System.out.println("Ano: " + c.get(Calendar.YEAR));
21     System.out.println("Mês: " + (c.get(Calendar.MONTH) + 1));
22     System.out.println("Dia do Mês: " + c.get(Calendar.DAY_OF_MONTH));
23     int hora = c.get(Calendar.HOUR_OF_DAY);
24     if(hora > 6 && hora < 12)
25     { System.out.println("Bom Dia");
26     }
27     else
28     {   if(hora > 12 && hora < 18)
29         { System.out.println("Boa Tarde");
30         }
31         else
32         { System.out.println("Boa Noite");
33         }
34     }
35 }
36 }
```



### Tópico

- A classe *DateFormat*

## A classe *DateFormat*

### Definição



- A classe **DateFormat** permite converter informações do tipo *String* para data do tipo **Date**, permitindo seguir um formato;
- Consegue-se trabalhar de modo inverso, convertendo um dado do tipo **Date** para uma *String*;
- Por ser uma classe abstrata, não é possível instanciá-la, por isso deve ser usado para método estático *getDateInstance( )*;; sendo necessário importar o pacote *java.text*;
- Os exemplos a seguir apresentam algumas formatações de datas importantes.

# ECM251 – Linguagens de Programação I

## A classe DateFormat

### Exemplos



```
1 import java.util.Calendar;
2 import java.text.DateFormat;
3 import java.util.Date;
4
5 public class Formatando_Datas
6 { public static void main(String[] args)
7   {
8     Calendar c = Calendar.getInstance();
9     c.set(2013, Calendar.FEBRUARY, 28);
10    Date data = c.getTime();
11    System.out.println("Data atual sem formatação: " + data);
12    //Formata a data
13    DateFormat formataData = DateFormat.getDateInstance();
14    System.out.println("Data atual com formatação: "+ formataData.format(data));
15    //Formata Hora
16    DateFormat hora = DateFormat.getTimeInstance();
17    System.out.println("Hora formatada: " + hora.format(data));
18    //Formata Data e Hora
19    DateFormat dtHora = DateFormat.getDateTimeInstance();
20    System.out.println(dtHora.format(data));
21  }
22 }
```

## A classe DateFormat

### Exemplos



```
1 import java.util.Calendar;
2 import java.text.DateFormat;
3 import java.util.Date;
4
5 public class Formatando_Saida_Datas
6 {   public static void main(String[] args)
7     {
8         Calendar c = Calendar.getInstance();
9         Date data = c.getTime();
10        DateFormat f = DateFormat.getDateInstance(DateFormat.FULL); //Data Completa
11        System.out.println("Data brasileira: "+f.format(data));
12        f = DateFormat.getDateInstance(DateFormat.LONG);
13        System.out.println("Data sem o dia descrito: "+f.format(data));
14        f = DateFormat.getDateInstance(DateFormat.MEDIUM);
15        System.out.println("Data resumida 1: "+f.format(data));
16        f = DateFormat.getDateInstance(DateFormat.SHORT);
17        System.out.println("Data resumida 2: "+f.format(data));
18    }
19 }
```

# ECM251 – Linguagens de Programação I

## Manipulação de Ícones, Senhas e Datas

### Tópico

- Conversões para datas

## Conversões para datas

### Definição



- Às vezes é preciso transformar um texto em uma data ou vice-versa;
- Para isso utiliza-se o método *parse()* e a classe **SimpleDateFormat**;
- A classe **SimpleDateFormat** é mais usada quando se trata de formatação de datas, pois já no seu construtor, quando instanciada, permite passar como argumento o formato da data desejada, como apresentada no exemplo a seguir.

# ECM251 – Linguagens de Programação I

## Conversões para datas

### Exemplo



```
1 import java.util.Calendar;
2 import java.text.DateFormat;
3 import java.util.Date;
4 import java.text.SimpleDateFormat;
5 import java.text.ParseException;
6
7 public class Conversao_Datas
8 {   public static void main(String[] args) throws ParseException
9     {   Calendar c = Calendar.getInstance();
10        Date data = c.getTime();
11        DateFormat f = DateFormat.getDateInstance();
12        System.out.println("Data: " + f.format(data));
13
14        SimpleDateFormat sdf = new SimpleDateFormat("dd/MM/yyyy");
15        System.out.println("Data formatada: " + sdf.format(data));
16
17        //Converte Objetos
18        System.out.println("Data convertida: " + sdf.parse("02/08/1970"));
19    }
20 }
```



# ECM251 – Linguagens de Programação I

## Manipulação de Ícones, Senhas e Datas

### Tópico

- Internacionalização de datas

## Internacionalização de datas

### Definição



- O Java oferece a classe **Locale**, que permite definir de qual país deseja-se obter o retorno das informações, como data e hora;
- No exemplo a seguir não foi necessário instanciar essa classe, pois as configurações regionais já são detectadas automaticamente do computador.

# ECM251 – Linguagens de Programação I

## Internacionalização de datas

### Exemplo



```
1 import java.util.Calendar;
2 import java.util.Locale;
3 import java.text.DateFormat;
4 import java.text.ParseException;
5 import java.util.Date;
6
7 public class Locale_Datas
8 {   public static void main(String[] args) throws ParseException
9     {   Calendar c = Calendar.getInstance();
10        Date data = c.getTime();
11        Locale brasil = new Locale("pt", "BR"); //Retorna do país e a língua
12        Locale eua = Locale.US;
13        Locale italia = Locale.ITALIAN;
14        DateFormat fb = DateFormat.getDateInstance(DateFormat.FULL, brasil);
15        System.out.println("Data e hora brasileira: " + fb.format(data));
16        DateFormat sb = DateFormat.getDateInstance(DateFormat.SHORT, brasil);
17        System.out.println("Data e hora brasileira: " + sb.format(data));
18        DateFormat fe = DateFormat.getDateInstance(DateFormat.FULL, eua);
19        System.out.println("Data e hora americana: " + fe.format(data));
20        DateFormat se = DateFormat.getDateInstance(DateFormat.SHORT, eua);
21        System.out.println("Data e hora americana: " + se.format(data));
22        DateFormat fi = DateFormat.getDateInstance(DateFormat.FULL, italia);
23        System.out.println("Data e hora italiana: " + fi.format(data));
24        DateFormat si = DateFormat.getDateInstance(DateFormat.SHORT, italia);
25        System.out.println("Data e hora italiana: " + si.format(data));
26    }
27 }
```

### Tópico

- Caracteres de conversão de datas

## Caracteres de conversão de datas

### Definição



- Com os caracteres de conversão **t** ou **T**, pode-se imprimir datas e horas em vários formatos;
- Os caracteres de conversão **t** ou **T** sempre são seguidos por um caractere de sufixo de conversão, que especifica o formato de data e/ou horas;
- Quando o caracteres de conversão **T** é utilizado, a saída é exibida em letras maiúsculas.

## Caracteres de conversão de datas

### Definição



- Caractere de sufixo de conversão para composição de data e hora:

Caractere de sufixo de conversão	Descrição
c	Exibe a data e hora formatadas como day month date hour:minute:second time-zone year com os três caracteres para day e month, dois dígitos para date, hour, minute e second e quatro dígitos para year — por exemplo, Wed Mar 03 16:30:25 GMT-05:00 2004. O relógio de 24 horas é utilizado. Neste exemplo, GMT-05:00 é o fuso horário.
F	Exibe a data formatada como year-month-date com quatro dígitos para o year e dois dígitos cada para month e a date (por exemplo, 2004-05-04).
D	Exibe a data formatada como month/day/year com dois dígitos cada para o month, day e year (por exemplo, 03/03/04).
r	Exibe a hora formatada como hour:minute:second AM PM com dois dígitos cada para hour, minute e second (por exemplo, 04:30:25 PM). O relógio de 12 horas é utilizado.
R	Exibe a hora formatada como hour:minute com dois dígitos cada para a hour e minute (por exemplo, 16:30). O relógio de 24 horas é utilizado.
T	Exibe a hora formatada como hour:minute:second com dois dígitos para o hour, minute e second (por exemplo, 16:30:25). O relógio de 24 horas é utilizado.



## Caracteres de conversão de datas

### Definição



- Caractere de sufixo de conversão de formatação de data:

Caractere de sufixo de conversão	Descrição
A	Exibe o nome completo do dia da semana (por exemplo, Wednesday).
a	Exibe o nome abreviado com três caracteres do dia da semana (por exemplo, Wed).
B	Exibe o nome completo do mês (por exemplo, March).
b	Exibe o nome abreviado com três caracteres do mês (por exemplo, Mar).
d	Exibe o dia do mês com dois dígitos, preenchendo com zeros à esquerda conforme necessário (por exemplo, 03).
m	Exibe o mês com dois dígitos, preenchendo com zeros à esquerda conforme necessário (por exemplo, 07).
e	Exibe o dia do mês sem zeros à esquerda (por exemplo, 3).
Y	Exibe o ano com quatro dígitos (por exemplo, 2004).
y	Exibe os dois últimos dígitos do ano com zeros à esquerda conforme necessário (por exemplo, 04).
j	Exibe o dia do ano com três dígitos, preenchendo com zeros à esquerda conforme necessário (por exemplo, 016).



## Caracteres de conversão de datas

### Definição



- Caractere de sufixo de conversão de formatação de tempo:

Caractere de sufixo de conversão	Descrição
H	Exibe horas no relógio de 24 horas com um zero à esquerda conforme necessário (por exemplo, 16).
I	Exibe horas no relógio de 12 horas com um zero à esquerda conforme necessário (por exemplo, 04).
k	Exibe horas em relógio de 24 horas sem zeros à esquerda (por exemplo, 16).
l	Exibe horas em relógio de 12 horas sem zeros à esquerda (por exemplo, 4).
M	Exibe minutos com um zero à esquerda conforme necessário (por exemplo, 06).
S	Exibe segundos com um zero à esquerda conforme necessário (por exemplo, 05).
Z	Exibe a abreviação para o fuso horário (por exemplo, GMT-05:00, significa Eastern Standard Time, que está 5 horas atrás do Greenwich Mean Time).
P	Exibe marcador de manhã ou de tarde em letras minúsculas (por exemplo, pm).
p	Exibe marcador de manhã ou de tarde em letras maiúsculas (por exemplo, PM).

# ECM251 – Linguagens de Programação I

## Caracteres de conversão de datas

### Exemplo



```
1 import java.util.Calendar;
2
3 public class DateTimeTest
4 {   public static void main( String args[] )
5     {   // obtém a data e hora atual
6         Calendar dateTime = Calendar.getInstance();
7         // imprimindo com caracteres de conversão para composições de data/hora
8         System.out.printf("%tc\n", dateTime);
9         System.out.printf("%tF\n", dateTime);
10        System.out.printf("%tD\n", dateTime);
11        System.out.printf("%tr\n", dateTime);
12        System.out.printf("%tT\n", dateTime);
13        // imprimindo com caracteres de conversão para data
14        System.out.printf("%1$tA, %1$tB %1$td, %1$tY\n", dateTime);
15        System.out.printf("%1$TA, %1$TB %1$Td, %1$TY\n", dateTime);
16        System.out.printf("%1$ta, %1$tb %1$te, %1$ty\n", dateTime);
17        // imprimindo com caracteres de conversão para hora
18        System.out.printf("%1$tH:%1$tM:%1$tS\n", dateTime);
19        System.out.printf("%1$tZ %1$tI:%1$tM:%1$tS %tp", dateTime);
20    }
21 }
```

### Exercícios



1. Analisar, codificar, executar e apresentar ao professor todos os exercícios exemplos deste material; e
2. Utilizando os componentes vistos neste material e nos anteriores, em grupo, desenvolver uma aplicação onde o usuário só conseguirá acessá-la através de **login** e **senha** corretos, pré-cadastrados e armazenados em uma tabela de **login\_senha** no banco de dados MySQL;

### Exercícios



- Ao ser validado, o usuário terá acesso às suas notas e faltas do curso do IMT, apresentados em uma tabela, preenchida com valores também vindos do banco de dados;
- Qualquer dado alterado pelo usuário na tabela, deverá ser armazenado no banco de dados, que também registrará a data e a hora da última alteração realizada;

# ECM251 – Linguagens de Programação I

## Manipulação de Ícones, Senhas e Datas

### Exercícios



- Este exercício deverá ser desenvolvido ao longo da aula de exercícios e apresentado ao professor até o término da aula, para ser por ele avaliado proporcionalmente;
- Todas as funcionalidades que não forem entregues até o final da aula de exercícios deverão ser entregues no CanvasLMS até o início da próxima aula da matéria.

### Bibliografia Básica



- MILETTO, Evandro M.; BERTAGNOLLI, Silvia de Castro. Desenvolvimento de software II: introdução ao desenvolvimento web com HTML, CSS, javascript e PHP (Tekne). Porto Alegre: Bookman, 2014. E-book. Referência Minha Biblioteca:  
<https://integrada.minhabiblioteca.com.br/#/books/9788582601969>
- WINDER, Russel; GRAHAM, Roberts. Desenvolvendo Software em Java, 3ª edição. Rio de Janeiro: LTC, 2009. E-book. Referência Minha Biblioteca:  
<https://integrada.minhabiblioteca.com.br/#/books/978-85-216-1994-9>
- DEITEL, Paul; DEITEL, Harvey. Java: how to program early objects. Hoboken, N. J: Pearson, c2018. 1234 p. ISBN 9780134743356.

*Continua...*



### Bibliografia Básica (continuação)



- HORSTMANN, Cay S; CORNELL, Gary. Core Java. SCHAFRANSKI, Carlos (Trad.), FURMANKIEWICZ, Edson (Trad.). 8. ed. São Paulo: Pearson, 2010. v. 1. 383 p. ISBN 9788576053576.
- LIANG, Y. Daniel. Introduction to Java: programming and data structures comprehensive version. 11. ed. New York: Pearson, c2015. 1210 p. ISBN 9780134670942.
- TURINI, Rodrigo. Desbravando Java e orientação a objetos: um guia para o iniciante da linguagem. São Paulo: Casa do Código, [2017]. 222 p. (Caelum).



### Bibliografia Complementar



- HORSTMANN, Cay. Conceitos de Computação com Java. Porto Alegre: Bookman, 2009. E-book. Referência Minha Biblioteca:  
<https://integrada.minhabiblioteca.com.br/#/books/9788577804078>
- MACHADO, Rodrigo P.; FRANCO, Márcia H. I.; BERTAGNOLLI, Silvia de Castro. Desenvolvimento de software III: programação de sistemas web orientada a objetos em java (Tekne). Porto Alegre: Bookman, 2016. E-book. Referência Minha Biblioteca:  
<https://integrada.minhabiblioteca.com.br/#/books/9788582603710>
- BARRY, Paul. Use a cabeça! Python. Rio de Janeiro: Alta Books, 2012. 458 p.  
ISBN 9788576087434.

*Continua...*

# ECM251 – Linguagens de Programação I

## Aula 14 – L1/1, L2/1 e L3/1

### Bibliografia Complementar (continuação)



- LECHETA, Ricardo R. Web Services RESTful: aprenda a criar Web Services RESTful em Java na nuvem do Google. São Paulo: Novatec, c2015. 431 p.  
ISBN 9788575224540.
- SILVA, Maurício Samy. JQuery: a biblioteca do programador. 3. ed. rev. e ampl. São Paulo: Novatec, 2014. 544 p.  
ISBN 9788575223871.
- SUMMERFIELD, Mark. Programação em Python 3: uma introdução completa à linguagem Python. Rio de Janeiro: Alta Books, 2012. 506 p.  
ISBN 9788576083849.

*Continua...*

# ECM251 – Linguagens de Programação I

## Aula 14 – L1/1, L2/1 e L3/1

### Bibliografia Complementar (continuação)



- YING, Bai. Practical database programming with Java. New Jersey: John Wiley & Sons, c2011. 918 p.
- ZAKAS, Nicholas C. The principles of object-oriented JavaScript. San Francisco, CA: No Starch Press, c2014. 97 p. ISBN 9781593275402.

# ECM251 – Linguagens de Programação I

## Aula 14 – L1/1, L2/1 e L3/1

FIM

## Manipulação de Ícones, Senhas e Datas (L1/2, L2/2 e L3/2)

**2025**

# ECM251 – Linguagens de Programação I

## Aula 14 – L1/2, L2/2 e L3/2

### Horário

Terça-feira: 2 x 2 aulas/semana

- L1/1 (07h40min-09h20min): *Prof. Calvetti*;
- L1/2 (09h30min-11h10min): *Prof. Calvetti*;
- L2/1 (07h40min-09h20min): *Prof. Menezes*;
- L2/2 (11h20min-13h00min): *Prof. Calvetti*;
- L3/1 (09h30min-11h10min): *Prof. Evandro*;
- L3/2 (11h20min-13h00min): *Prof. Evandro*.

# ECM251 – Linguagens de Programação I

## *JTable, JComboBox, JMenu e Diálogos*

### Exercícios



- Terminar e apresentar ao professor para avaliação, os exercícios propostos na aula de teoria, deste material.



### Bibliografia (apoio)



- LOPES, ANITA. GARCIA, GUTO. Introdução à Programação: 500 algoritmos resolvidos. Rio de Janeiro: Elsevier, 2002.
- DEITEL, P. DEITEL, H. Java: como programar. 8 Ed. São Paulo: Prentice-Hall (Pearson), 2010;
- BARNES, David J.; KÖLLING, Michael. Programação orientada a objetos com Java: uma introdução prática usando o BlueJ . 4. ed. São Paulo: Pearson Prentice Hall, 2009.

# ECM251 – Linguagens de Programação I

## Aula 14 – L1/2, L2/2 e L3/2

FIM