Curso de Java

aula 06

Prof. Anderson Henrique

TECH
soluções em ti

Formatação de números

- Aprenderemos a utilizar a classe NumberFormat( ), esta classe é abstrata, o que significa que não precisamos instanciar novo objeto para utilizar os métodos oferecidos nesta classe.
- O primeiro método é um formatador genérico de um número

Ex.: double saldo = 123_456.789;

NumberFormat f = NumberFormat.getInstance( );

f.format(saldo);

- Este formatador é para números do tipo inteiro

```
NumberFormat fi = getIntegerInstance( );
fi.format(saldo);
```

- Este formatador é para números percentuais

```
NumberFormat fp = getPercentInstance( );
fp.format(0.25);
```

- Este formatador é para números monetários

```
NumberFormat fm = getCurrencyInstance( );
fm.format(saldo);
```

- Podemos definir a quantidade de casas decimais

```
f.setMaximumFractionDigits(1);
```

Internacionalização de números

Ex.: fus = NumberFormat.getCurrencyInstance(Locale.US);

Ex.: ffr = NumberFormat.getCurrencyInstance(Locale.FRANCE);

- Podemos converter números, realizar o parse de um número

Ex.: f = NumberFormat.getCurrencyInstance( );

System.out.print(f.parse(“R$ 1.100,25”));

Manipulação de Data e Hora

- As datas são representadas por objetos do tipo Date e manipuladas por objetos do tipo Calendar
- Todo o tempo em Java é representado em milissegundos com o tipo long, e esse tempo é medido a partir de 01 de Janeiro de 1970
- O nosso computador tem a informação de qual é o nosso tempo atual, e podemos recuperá-lo em milissegundos através do método System.currentTimeMillis( );

- Para criarmos uma data, instanciamos um objeto do tipo Date( )

Ex.: Date agora = new Date( );

System.out.print(agora);

Ex.: Date data = new Date(1_000_000_000_000L);

System.out.print(data);

data.getTime( ); //recupera o tempo em milissegundos

data.setTime(1_000_000_000_000L);

data.compareTo(agora);

- Para manipular datas você deve trabalhar com a classe Calendar

Ex.: Calendar c = Calendar.getInstance( );

c.set(1980,Calendar.FEBRUARY, 12);

System.out.println(c.getTime); //recuperamos objeto do tipo date

c.get(Calendar.YEAR) // ano

c.get(Calendar.MONTH) //mes

c.get(Calendar.DAY_OF_MONTH) //dia do mês

c.set(Calendar.YEAR, 1972) // altera o ano

c.set(Calendar.MONTH, Calendar.MARCH) // altera o mês

c.set(Calendar.DAY_OF_MONTH, 25) // altera o dia do mês

- Podemos limpar os campos, utilizando o método clear

Ex.:

```
c.clear(Calendar.MINUTE);
c.clear(Calendar.SECOND);
```

- Podemos aumentar ou diminuir unidades de tempo, add

```
c.add(Calendar.DAY_OF_MONTH, 1);
c.add(Calendar.YEAR, 1);
c.add(Calendar.DAY_OF_MONTH, -1);
c.add(Calendar.YEAR, -1);
```

- Podemos aumentar ou diminuir unidades de tempo, sem alterar unidades superiores

Ex.:

```
c.roll(Calendar.DAY_OF_MONTH, 20);
c.roll(Calendar.DAY_OF_MONTH, -20);
```

Ex.: Saudação (bom dia, boa tarde, boa noite)

```
Calendar c1 = Calendar.getInstance( );
int hora = c1.get(Calendar.HOUR_OF_DAY);
```

```
If(hora < 12){
        System.out.println(“Bom dia”);
}else if(hora > 12 && hora < 18){
        System.out.println(“Boa tarde”);
}else{
        System.out.println(“Boa noite”);
}
```

Formatação de Data e Hora

- Existem duas classes específicas para a formatação de Data e Hora, DateFormat e SimpleDateFormat

Ex.: Calendar c = Calendar.getInstance( );

c.set(1980, Calendar.FEBRUARY, 12);

Date data = c.getTime( ); //recupera o objeto do tipo date

DateFormat fd = DateFormat.getDateInstance( ); //formatar datas

System.out.println(fd.format(data)) //retorna string

DateFormat fh = DateFormat.getTimeInstance( ); //formatar horário

System.out.println(fh.format(data));

DateFormat fdh = DateFormat.getDateTimeInstance( ); //formatar data e horário

System.out.println(fdh.format(data));

Estilos diferentes para Data e Hora

fd = DateFormat.getDateInstance(DateFormat.FULL);
System.out.println(fd.format(data)); //formato completo

fd = DateFormat.getDateInstance(DateFormat.LONG);
System.out.println(fd.format(data)); //formato longo

```
fd = DateFormat.getDateInstance( DateFormat.MEDIUM);
System.out.println(fd.format(data)); //formato médio

d = DateFormat.getDateInstance( DateFormat.SHORT);
System.out.println(fd.format(data)); //formato curto
```

- Podemos utilizar a classe de formatação para converter uma String em uma data

```
Date data2 = f.parse(“12/02/1980”); //lança uma exceção

System.out.println(f.format(data2));
```

- Utiliza no construtor o formato de uma data

Ex.: SimpleDateFormat sdf = new SimpleDateFormat(“dd/MM/yyyy”);

System.out.println(sdf.format(data));

System.out.println(sdf.parse(data));

Internacionalização de Data e Hora

- É possível que durante a carreira de um programador, ele tenha que construir sistemas utilizados em outros países, de pessoas que falam o idioma diferente.
- Vamos internacionalizar, adaptá-los para outros países e regiões

Ex.:

```
Calendar c = Calendar.getInstance( );
c.set(2018, Calendar.JULY, 29);
Date data = c.getTime( );

// ISO 639 – Língua, ISO 3166 – país
Locale padrao = Locale.getDefault( );
Locale brasil = new Locale(“pt”, “BR” );

DateFormat f = DateFormat.getDateInstance(DateFormat.FULL);
System.out.println(f.format(data));
```

```
DateFormat f = DateFormat.getDateInstance(DateFormat.FULL, brasil);
System.out.println(f.format(data));

Locale usa = Locale.US;
Locale jp    = Locale.JAPAN;
Locale ita   = Locale.ITALY;
```

Prosseguiremos no próximo slide...

Professor: Anderson Henrique

Programador nas Linguagens Java e PHP