POR CRISTIANE PAGINE

JAVA

LIDANDO COM DATAS E HORAS EM JAVA

2025

1. Classe java.util.Date

A classe Date no pacote java.util representa uma data simples. Para utilizá-la, basta criar uma variável do tipo Date:

```
java.util.Date x;
```

Se você importar o pacote java.util.*, pode omitir o prefixo java.util, ficando assim:

```
Date x;
```

Criando um objeto de Date:

```
x = new Date();
```

Ao criar o objeto, a data e hora do sistema são automaticamente armazenadas no objeto x.
Para exibir a data e hora atuais:

```
Date x = new Date();
System.out.println(x);
```

2. Convertendo Datas com SimpleDateFormat

Transformar String em Date:

A classe SimpleDateFormat serve para formatar datas, convertendo uma string para o formato de data desejado.
Exemplo para pegar uma data digitada pelo usuário e convertê-la em um objeto Date:

```
import java.util.*;
import java.text.*;

 // Pegando dados de um formulário
String x = request.getParameter("dataUsuario");

// Definindo o formato
SimpleDateFormat sdf1 = new
SimpleDateFormat("dd/MM/yyyy"); Date dataUsuario =
sdf1.parse(x);
```

Transformar Date em String também é possivel:

```
//pegando dados de um formulário WEB
String x=request.getParameter("dataUsuario");

//você pode usar outras máscaras
SimpleDateFormat sdf1= new
SimpleDateFormat("dd/MM/yyyy");
Date y=new Date();
System.out.println(sdf1.format(y));
```

3. Calendário Gregoriano

A classe GregorianCalendar pertence ao pacote java.util e permite manipular datas no calendário gregoriano. Vamos explorar exemplos de uso.

Criando uma instância do calendário gregoriano:

```
import java.util.GregorianCalendar;
import java.util.Calendar;

public class ExemploGregorianCalendar {
    public static void main(String[] args) {
        GregorianCalendar dataAtual = new
GregorianCalendar();
        System.out.println("Data atual: " +
dataAtual.getTime());

        // Criar uma data específica
        GregorianCalendar dataEspecifica = new
GregorianCalendar(2024, Calendar.DECEMBER, 25);
        System.out.println("Data específica: " +
dataEspecifica.getTime());
    }
}
```

Alterando componentes da data:

```
public class AlterarData {
    public static void main(String[] args) {
        GregorianCalendar calendario = new
GregorianCalendar(2024, Calendar.JANUARY, 1);

        // Alterar o mês para Fevereiro
        calendario.set(Calendar.MONTH,
Calendar.FEBRUARY);
        System.out.println("Data alterada: " +
calendario.getTime());

        // Adicionar 10 dias
        calendario.add(Calendar.DAY_OF_MONTH, 10);
        System.out.println("Data após adicionar 10 dias:
" + calendario.getTime());
    }
}
```

Verificando Anos Bissextos:

```
public class VerificarAnoBissexto {
    public static void main(String[] args) {
        GregorianCalendar calendario = new
GregorianCalendar();

        int ano = 2024;
        boolean bissexto = calendario.isLeapYear(ano);
        System.out.println("O ano " + ano + (bissexto ?
" é bissexto." : " não é bissexto."));
    }
}
```

4. Classe LocalDate

A partir do Java 8, a API java.time oferece uma maneira mais moderna de trabalhar com datas.
Criando Instâncias de LocalDate:

```
import java.time.LocalDate;
public class ExemploLocalDate {
    public static void main(String[] args) {
        LocalDate hoje = LocalDate.now();
        System.out.println("Data atual: " + hoje);

        // Criar uma data específica
        LocalDate natal = LocalDate.of(2024, 12, 25);
        System.out.println("Natal de 2024: " + natal);
    }
}
```

Operações com datas utilizando LocalDate:

```
import java.time.LocalDate;
import java.time.temporal.ChronoUnit;

public class OperacoesComDatas {
    public static void main(String[] args) {
        LocalDate hoje = LocalDate.now();
        LocalDate futuro = hoje.plusDays(30);
        System.out.println("Data atual: " + hoje);
        System.out.println("Data daqui a 30 dias: " +
futuro);

        // Diferença entre duas datas
        LocalDate inicioDoAno = LocalDate.of(2024, 1,
1);
        long diasPassados =
ChronoUnit.DAYS.between(inicioDoAno, hoje);
        System.out.println("Dias desde o início do ano:
" + diasPassados);
    }
}
```