

UNIVERSIDADE FEDERAL DO RIO GRANDE DO NORTE
INSTITUTO METRÓPOLE DIGITAL

AULA 09

LINGUAGEM DE PROGRAMAÇÃO 2
JAVA



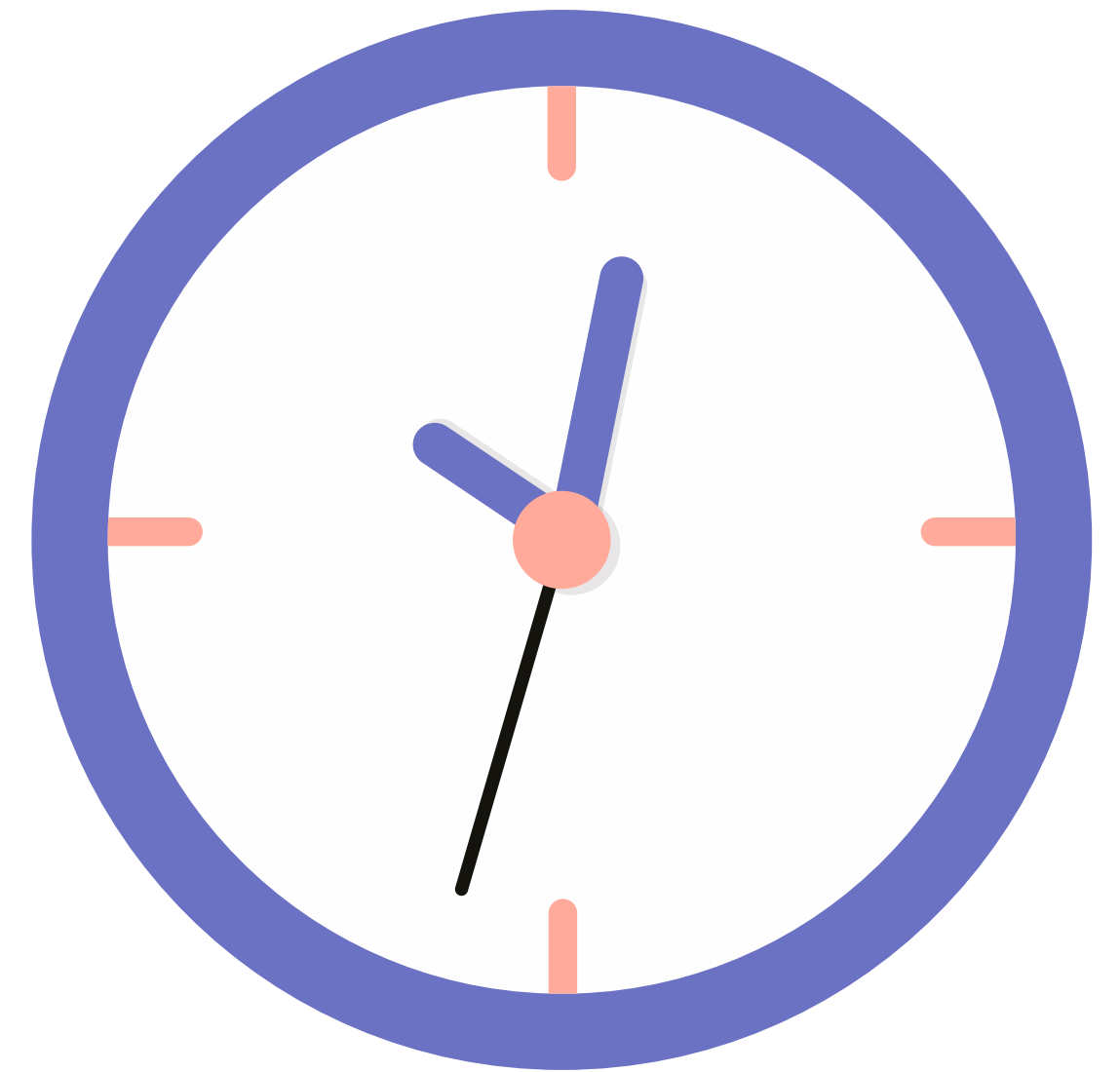
PROF. JANIHERYSON FELIPE

CONTEÚDO DESSA AULA

- **APRENDER COMO TRABALHAR COM DATAS NO JAVA;**
- **APRENDER COMO TRABALHAR COM HORAS NO JAVA;**
- **APRENDER A CONVERTER STRINGS EM DATAS E VICE-VERSA;**
- **DISCUSSÕES E DÚVIDAS GERAIS.**

TRABALHANDO COM TEMPO

A maior parte dos programas que desenvolvemos no dia a dia deve ser capazes de manipular **datas** e **horas**. Seja para setar uma sessão de usuário, ou para executar uma determinada ação ou interação com o usuário. Exemplo do uso desse recursos são os jogos, as agendas virtuais, marcadores de tempo, sistemas de backup, etc.



BIBLIOTECAS DE TEMPO DO JAVA

Java possui algumas bibliotecas de manipulação de datas e horas

```
import java.time.LocalDate;  
import java.time.LocalTime;  
import java.time.LocalDateTime;  
import java.util.Date; //Deprecated
```

```
Date data = new Date();  
data.getSeconds();
```



BIBLIOTECAS DE TEMPO DO JAVA

Java possui algumas bibliotecas de manipulação de datas e horas

```
import java.time.LocalDate;  
import java.time.LocalDateTime;  
import java.time.LocalTime;
```



TRABALHANDO COM DATAS LOCALDATE

Para trabalharmos com datas, devemos utilizar a classe **LocalDate**. Esta classe representa uma data, sem hora, minuto, segundo, etc. O método estático **now()** retorna a data atual.

```
var dataAtual = LocalDate.now();  
System.out.println(dataAtual);
```



2024-04-15



TRABALHANDO COM DATAS LOCALDATE

- `var dataEspecifica = LocalDate.of(2019, 10, 10);` Cria uma data especifica;
- `System.out.println(dataAtual.getDayOfMonth());` Recupera o mês;
- `System.out.println(dataAtual.getMonth());` Recupera o dia;
- `System.out.println(dataAtual.getYear());` Recupera o ano;
- `System.out.println(dataAtual.getDayOfYear());` Retorna o dia no ano (ex: 87)
- `dataAtual.plusDays(10);` Soma 10 dias;
- `dataAtual.plusMonths(2);` Soma dois meses;
- `dataAtual.plusYears(2);` Soma dois anos;



TRABALHANDO COM DATAS LOCALDATE

Para trabalharmos com horas, semelhantemente as datas, devemos utilizar a classe **LocalTime**. Esta classe representa uma hora, , minuto, segundo, etc. O método estático **now()** retorna a hora atual.

```
var horaAtual = LocalTime.now();  
System.out.println(horaAtual);
```



20:21:02.908441



TRABALHANDO COM DATAS LOCALDATE

- `var horaEspecifica = LocalTime.of(10, 10, 10);` Cria uma hora especifica;
- `System.out.println(dataAtual.getDayOfMonth());` Recupera o mês;
- `System.out.println(dataAtual.getMonth());` Recupera o dia;
- `System.out.println(dataAtual.getYear());` Recupera o ano;
- `System.out.println(dataAtual.getDayOfYear());` Retorna o dia no ano (ex: 87)
- `dataAtual.plusDays(10);` Soma 10 dias;
- `dataAtual.plusMonths(2);` Soma dois meses;
- `dataAtual.plusYears(2);` Soma dois anos;



TRABALHANDO COM DATAS LOCALDATETIME

Esse é o mais completo de todos os métodos, pois trabalha com datas e horas simultaneamente. O método estático **now()** retorna a data e a hora atual.

```
var dateTime = LocalDateTime.now();  
System.out.println(dateTime);
```



```
2024-04-16T13:25:58.589112800
```



TRABALHANDO COM DATAS LOCALDATE

- `var atual = LocalDateTime.of(2024, 1, 28, 10, 59, 10);`
- `System.out.println(atual.getYear());` //Retorna o ano
- `System.out.println(atual.getMonth());` //Retorna o mês
- `System.out.println(atual.getDayOfMonth());` //Retorna o dia do mês
- `System.out.println(atual.getMonthValue());` //Retorna o mês numérico
- `System.out.println(atual.getDayOfYear());` //Retorna o dia da ano
- `System.out.println(atual.getDayOfWeek());` //Retorna o dia do semana















TRABALHANDO COM DATAS LOCALDATE

- `System.out.println(atual.getHour());` //Retorna a hora
- `System.out.println(atual.getMinute());` //Retorna os minutos
- `System.out.println(atual.getSecond());` //Retorna os segundos
- `System.out.println(atual.getNano());` //Retorna o nanossegundos



TRABALHANDO COM DATAS LOCALDATE

 plus (TemporalAmount amountToAdd)	LocalDateTime
 plus (long amountToAdd, TemporalUnit un...)	LocalDate...
 plusDays (long days)	LocalDateTime
 plusHours (long hours)	LocalDateTime
 plusMinutes (long minutes)	LocalDateTime
 plusMonths (long months)	LocalDateTime
 plusNanos (long nanos)	LocalDateTime
 plusSeconds (long seconds)	LocalDateTime
 plusWeeks (long weeks)	LocalDateTime
 plusYears (long years)	LocalDateTime
 parse (CharSequence text)	LocalDateTime
 parse (CharSequence text, DateTimeFormatter ...)	Loc...



TRABALHANDO COM DATAS LOCALDATE

- `System.out.println(atual.getHour());` //Retorna a hora
- `System.out.println(atual.getMinute());` //Retorna os minutos
- `System.out.println(atual.getSecond());` //Retorna os segundos
- `System.out.println(atual.getNano());` //Retorna o nanossegundos



CLASSE CALENDAR

A classe Calendar tambem permite trabalhar com datas e horas, no entanto, por simplicidade, vamos utilizar as classes ja mencionadas anteriormente.

```
Calendar datCalendar = Calendar.getInstance();  
  
System.out.println(datCalendar.getTime());  
System.out.println(Calendar.DAY_OF_WEEK);  
System.out.println(Calendar.YEAR);  
System.out.println(Calendar.MONTH);
```



FORMATANDO DATAS

```
var dataHoje = LocalDate.now();  
  
var dataFormatada = DateTimeFormatter.ofPattern(pattern: "dd/MM/yyyy");  
  
System.out.println(dataHoje.format(dataFormatada));
```



