Java LocalDateTime class

Java LocalDateTime class is an immutable date-time object that represents a date-time, with the default format as yyyy-MM-dd-HH-mm-ss.zzz. It inherits object class and implements the ChronoLocalDateTime interface.

Java LocalDateTime class declaration

Let's see the declaration of java.time.LocalDateTime class.

1. **public** **final** **class** LocalDateTime **extends** Object
2. **implements** Temporal, TemporalAdjuster, ChronoLocalDateTime<LocalDate>, Serializable

Methods of Java LocalDateTime

|  |  |
| --- | --- |
| **Method** | **Description** |
| String format(DateTimeFormatter formatter) | It is used to format this date-time using the specified formatter. |
| int get(TemporalField field) | It is used to get the value of the specified field from this date-time as an int. |
| LocalDateTime minusDays(long days) | It is used to return a copy of this LocalDateTime with the specified number of days subtracted. |
| static LocalDateTime now() | It is used to obtain the current date-time from the system clock in the default time-zone. |
| static LocalDateTime of(LocalDate date, LocalTime time) | It is used to obtain an instance of LocalDateTime from a date and time. |
| LocalDateTime plusDays(long days) | It is used to return a copy of this LocalDateTime with the specified number of days added. |
| boolean equals(Object obj) | It is used to check if this date-time is equal to another date-time. |

|  |  |
| --- | --- |
| **package** com.snow;  **import** java.time.\*; **import** java.time.format.DateTimeFormatter;  **public class** Main {   **public static void** main(String[] args) {  **try** {  LocalDateTime now = LocalDateTime.*now*();  System.***out***.println(**"Before Formatting: "** + now);  DateTimeFormatter format = DateTimeFormatter.*ofPattern*(**"dd-MM-yyyy HH:mm:ss"**);  String formatDateTime = now.format(format);  System.***out***.println(**"After Formatting: "** + formatDateTime);  } **catch** (Exception e){  System.***out***.println(**"error message : "** + e.getMessage());  }  } } | Before Formatting: 2018-03-23T01:07:57.656646700  After Formatting: 23-03-2018 01:07:57 |

|  |  |
| --- | --- |
| 1. **import** java.time.LocalDateTime; 2. **import** java.time.format.DateTimeFormatter; 3. **public** **class** LocalDateTimeExample2 { 4. **public** **static** **void** main(String[] args) { 5. LocalDateTime datetime1 = LocalDateTime.now(); 6. DateTimeFormatter format = DateTimeFormatter.ofPattern("dd-MM-yyyy HH:mm:ss"); 7. String formatDateTime = datetime1.format(format); 8. System.out.println(formatDateTime); 9. } 10. } | 14-01-2017 11:42:32 |
| Java LocalDateTime Example: now() | |
| 1. **import** java.time.LocalDateTime; 2. **import** java.time.format.DateTimeFormatter; 3. **public** **class** LocalDateTimeExample2 { 4. **public** **static** **void** main(String[] args) { 5. LocalDateTime datetime1 = LocalDateTime.now(); 6. DateTimeFormatter format = DateTimeFormatter.ofPattern("dd-MM-yyyy HH:mm:ss"); 7. String formatDateTime = datetime1.format(format); 8. System.out.println(formatDateTime); 9. } 10. } | 14-01-2017 11:42:32 |
| Java LocalDateTime Example: get() | |
| 1. **import** java.time.LocalDateTime; 2. **import** java.time.temporal.ChronoField; 3. **public** **class** LocalDateTimeExample3 { 4. **public** **static** **void** main(String[] args) { 5. LocalDateTime a = LocalDateTime.of(2017, 2, 13, 15, 56); 6. System.out.println(a.get(ChronoField.DAY\_OF\_WEEK)); 7. System.out.println(a.get(ChronoField.DAY\_OF\_YEAR)); 8. System.out.println(a.get(ChronoField.DAY\_OF\_MONTH)); 9. System.out.println(a.get(ChronoField.HOUR\_OF\_DAY)); 10. System.out.println(a.get(ChronoField.MINUTE\_OF\_DAY)); 11. } 12. } | 1  44  13  15  956 |

## Java LocalDateTime Example: minusDays()

|  |  |
| --- | --- |
| 1. **import** java.time.LocalDateTime; 2. **import** java.time.format.DateTimeFormatter; 3. **public** **class** LocalDateTimeExample4 { 4. **public** **static** **void** main(String[] args) { 5. LocalDateTime datetime1 = LocalDateTime.of(2017, 1, 14, 10, 34); 6. LocalDateTime datetime2 = datetime1.minusDays(100); 7. System.out.println("Before Formatting: " + datetime2); 8. DateTimeFormatter format = DateTimeFormatter.ofPattern("dd-MM-yyyy HH:mm"); 9. String formatDateTime = datetime2.format(format); 10. System.out.println("After Formatting: " + formatDateTime ); 11. } 12. } |  |

## Java LocalDateTime Example: plusDays()

|  |  |
| --- | --- |
| 1. **import** java.time.LocalDateTime; 2. **import** java.time.format.DateTimeFormatter; 3. **public** **class** LocalDateTimeExample5 { 4. **public** **static** **void** main(String[] args) { 5. LocalDateTime datetime1 = LocalDateTime.of(2017, 1, 14, 10, 34); 6. LocalDateTime datetime2 = datetime1.plusDays(120); 7. System.out.println("Before Formatting: " + datetime2); 8. DateTimeFormatter format = DateTimeFormatter.ofPattern("dd-MM-yyyy HH:mm"); 9. String formatDateTime = datetime2.format(format); 10. System.out.println("After Formatting: " + formatDateTime ); 11. } 12. } | Before Formatting: 2017-05-14T10:34  After Formatting: 14-05-2017 10:34 |