# **Java 8 Date/Time API - Java LocalDate class**

### **Methods of Java LocalDate**

|  |  |
| --- | --- |
| **Method** | **Description** |
| LocalDateTime atTime(int hour, int minute) | It is used to combine this date with a time to create a LocalDateTime. |
| int compareTo(ChronoLocalDate other) | It is used to compares this date to another date. |
| boolean equals(Object obj) | It is used to check if this date is equal to another date. |
| String format(DateTimeFormatter formatter) | It is used to format this date using the specified formatter. |
| int get(TemporalField field) | It is used to get the value of the specified field from this date as an int. |
| boolean isLeapYear() | It is used to check if the year is a leap year, according to the ISO proleptic calendar system rules. |
| LocalDate minusDays(long daysToSubtract) | It is used to return a copy of this LocalDate with the specified number of days subtracted. |
| LocalDate minusMonths(long monthsToSubtract) | It is used to return a copy of this LocalDate with the specified number of months subtracted. |
| static LocalDate now() | It is used to obtain the current date from the system clock in the default time-zone. |
| LocalDate plusDays(long daysToAdd) | It is used to return a copy of this LocalDate with the specified number of days added. |
| LocalDate plusMonths(long monthsToAdd) | It is used to return a copy of this LocalDate with the specified number of months added. |

## *Java LocalDate Example*

|  |  |
| --- | --- |
| **package** com.snow; **import** java.time.\*; **public class** Main {   **public static void** main(String[] args) {  **try** {  LocalDate date = LocalDate.*now*();  System.***out***.println(**"current date : "** + date);  System.***out***.println(**"yesterday : "** + date.minusDays(1));  System.***out***.println(**"tomorrow : "** + date.plusDays(1));  System.***out***.println(**"day after tomorrow : "** + date.plusDays(2));  } **catch** (Exception e){  System.***out***.println(**"error message : "** + e.getMessage());  }  } } | current date : 2018-03-22yesterday : 2018-03-21tomorrow : 2018-03-23day after tomorrow : 2018-03-24 |

|  |  |
| --- | --- |
| **package** com.snow;  **import** java.time.\*; **public class** Main {   **public static void** main(String[] args) {  **try** {  LocalDate localDate = LocalDate.*of*(2017, 4, 15);  System.***out***.println(**"date is : "** + localDate);   *// .isLeapYear()* **boolean** x = localDate.isLeapYear();  System.***out***.println(localDate + **" is "** + x + **" leapyear"**);   *//current date* LocalDate ld = LocalDate.*now*(); *//try 1* LocalDate ld1 = LocalDate.*now*(); *//try 2* System.***out***.println(**"current date "** + ld);   *// compareTo(ChronoLocalDate other)* **int** check\_val = ld.compareTo(localDate);  System.***out***.println(**"value of check\_val "** + check\_val); *// 1 for not match* **int** check\_val\_1 = ld.compareTo(ld1);  System.***out***.println(**"value of check\_val\_1 "** + check\_val\_1); *// 0 for match   //boolean equals(Object obj)* System.***out***.println(**"is ld and localdate same? "** + ld.equals(localDate));  System.***out***.println(**"is ld and ld1 same? "** + ld.equals(ld1));   } **catch** (Exception e){  System.***out***.println(**"error message : "** + e.getMessage());  }  } } | date is : 2017-04-15  2017-04-15 is false leapyear  current date 2018-03-23  value of check\_val 1  value of check\_val\_1 0  is ld and localdate same? false  is ld and ld1 same? true |

|  |  |
| --- | --- |
| **package** com.snow;  **import** java.time.\*; **public class** Main {   **public static void** main(String[] args) {  **try** {  LocalDate date = LocalDate.*of*(2017, 1, 13);  LocalDateTime datetime = date.atTime(1,50,9);  System.***out***.println(datetime);   } **catch** (Exception e){  System.***out***.println(**"error message : "** + e.getMessage());  }  } } | 2017-01-13T01:50:09 |