**Thư viện xử lý ngày tháng trong Java**

**1. Calendar**

Trong Java, Calendar cung cấp một tập hợp các phương thức để tương tác với thời gian. Chẳng hạn, chúng ta có thể sử dụng Calendar để lấy thời gian hiện tại của hệ thống và nó cũng được dùng để thực hiện các phép toán giữa ngày và giờ. Dưới đây là ví dụ minh họa cách sử dụng thư viện Calendar trong Java.

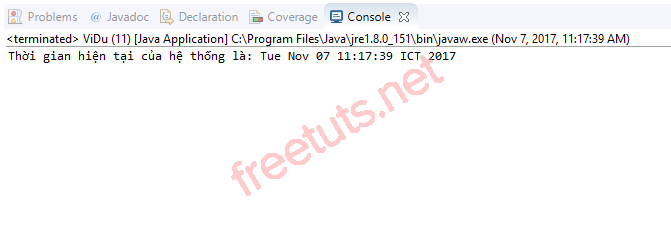
### **Lấy thời gian hiện tại của hệ thống**

Ví dụ dưới đây sẽ minh họa cách lấy thời gian hiện tại của hệ thống (bao gồm ngày tháng năm giờ phút giây) sử dụng Calendar:

*\***Ví dụ**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15 | public static void main(String[] args) {      // Tạo đối tượng Calendar mô tả thời điểm hiện tại      // với Locale (khu vực) và TimeZone (múi giờ)      // của máy tính đang chạy.      Calendar cal = Calendar.getInstance();        // import java.util.Date; để sử dụng Date      // hiển thị ngày tháng năm giờ phút giây hiện tại của hệ thống      // sử dụng phương thức getTime()      // phương thức này sẽ trả về 1 đối tượng Date      // lưu trữ thông tin ngày tháng năm giờ phút giây hiện tại của hệ thống      Date date = cal.getTime();      // hiển thị thời gian hiện tại      System.out.println("Thời gian hiện tại của hệ thống là: " + date);  } |

Kết quả sau khi biên dịch chương trình:

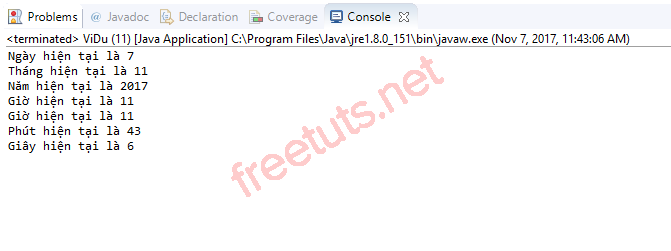


Ngoài ra, chúng ta có thể sử dụng Calendar để lấy riêng từng giá trị ngày, tháng, năm, giờ, phút, giây hiện tại của hệ thống như sau:

**Ví dụ**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35 | public static void main(String[] args) {      Calendar cal = Calendar.getInstance();        // hiển thị ngày hiện tai      // Calendar.DAY\_OF\_MONTH sẽ cho biết ngày hiện tại của tháng hiện tại      // hàm get() sẽ trả về giá trị ngày hiện tại      System.out.println("Ngày hiện tại là " + cal.get(Calendar.DAY\_OF\_MONTH));        // hiển thị tháng hiện tại      // sử dụng Calendar.MONTH      // vì tháng trong Java sẽ chạy từ 0 đến 11      // tức là nếu hiện giờ là tháng 7 thì chương trình sẽ hiển thị là tháng 6      // vì vậy để hiển thị đúng thì ta sẽ cộng thêm tháng đó cho 1      System.out.print("Tháng hiện tại là ");      System.out.println(cal.get(Calendar.MONTH)+1);        // hiển thị năm hiện tại      // Calendar.YEAR sẽ cho biết năm hiện tại      System.out.println("Năm hiện tại là " + cal.get(Calendar.YEAR));        // hiển thị giờ hiện tại      // sự khác nhau giữa HOUR và HOUR\_OF\_DAY là      // HOUR sẽ hiển thị theo khung giờ 12 giờ, ví dụ 11 giờ tối sẽ hiển thị là 11      // còn HOUR\_OF\_DAY sẽ hiển thị theo khung giờ 23 giờ, ví dụ 11 giờ tối sẽ hiển thị là 23      System.out.println("Giờ hiện tại là " + cal.get(Calendar.HOUR));      System.out.println("Giờ hiện tại là " + cal.get(Calendar.HOUR\_OF\_DAY));        // hiển thị phút hiện tại      // sử dụng Calendar.MINUTE      System.out.println("Phút hiện tại là " + cal.get(Calendar.MINUTE));        // hiển thị giây hiện tại      // sử dụng Calendar.SECOND      System.out.println("Giây hiện tại là " + cal.get(Calendar.SECOND));  } |

Kết quả sau khi biên dịch chương trình:



### **Thay đổi thời gian của hệ thống**

Trong Java, chúng ta có thể sử dụng phương thức set() để thay đổi thời gian như sau:

**Cú pháp**

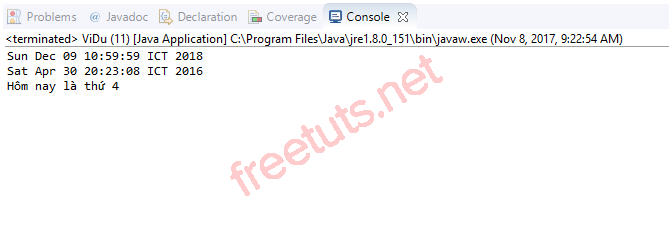
|  |  |
| --- | --- |
| 1  2 | Calendar cal = Calendar.getInstance();  cal.set(int field, int value); |

, trong đó field là tên giá trị cần thay đổi (ví dụ Calendar.MONTH là tháng) và value là giá trị mới của field.

**Ví dụ**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25 | public static void main(String[] args) {      Calendar cal = Calendar.getInstance();        // thay đổi ngày hiện tại thành ngày 9      // tháng hiện tại thành tháng 12      // năm hiện tại thành năm 2018      // giờ hiện tại thành 10      // phút hiện tại thành 59      // và giây hiện tại thành 59      // sử dụng phương thức set()      cal.set(Calendar.DAY\_OF\_MONTH, 9);      cal.set(Calendar.MONTH, Calendar.DECEMBER);      cal.set(Calendar.YEAR, 2018);      cal.set(Calendar.HOUR\_OF\_DAY, 10);      cal.set(Calendar.MINUTE, 59);      cal.set(Calendar.SECOND, 59);        // hiển thi thời gian hiện tại sau khi thay đổi      System.out.println(cal.getTime());        // hoặc chúng ta có thể thay đổi tất cả như sau      cal.set(2016, Calendar.APRIL, 30, 20, 23, 8);        System.out.println(cal.getTime());  } |

Kết quả sau khi biên dịch chương trình:



### **Cộng, trừ thời gian của hệ thống**

Trong Java, chúng ta có thể thực hiện phép cộng và trừ thời gian của hệ thống bằng cách sử dụng phương thức add() như sau:

**Cú pháp**

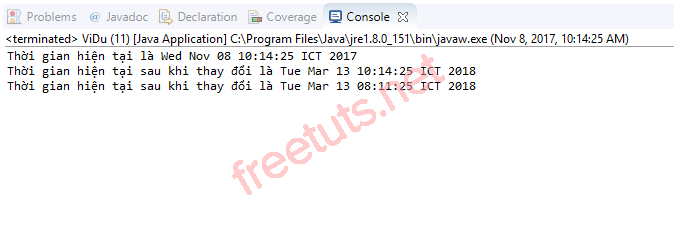
|  |  |
| --- | --- |
| 1  2 | Calendar cal = Calendar.getInstance();  cal.add(int field, int amount); |

, trong đó field là tên giá trị cần thay đổi và amount là số lượng được thêm vào hay trừ đi. Nếu amount > 0 thì sẽ tăng giá trị field lên ứng với giá trị amount, còn amount < 0 thì sẽ giảm giá trị của field ứng với giá trị amount đó.

**Ví dụ**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16 | public static void main(String[] args) {      Calendar cal = Calendar.getInstance();      System.out.println("Thời gian hiện tại là " + cal.getTime());        // cộng thời gian      // sử dụng phương thức add()      cal.add(Calendar.MONTH, 4); // thêm 4 tháng      cal.add(Calendar.DAY\_OF\_MONTH, 5);  // và 5 ngày      System.out.println("Thời gian hiện tại sau khi thay đổi là " + cal.getTime());        // trừ thời gian      // sử dụng phương thức add()      cal.add(Calendar.HOUR\_OF\_DAY, -2);  // giảm giờ hiện tại đi 2 tiếng      cal.add(Calendar.MINUTE, -3);   // giảm phút hiện tại đi 3 phút      System.out.println("Thời gian hiện tại sau khi thay đổi là " + cal.getTime());  } |

Kết quả sau khi biên dịch chương trình:



### **Phương thức roll()**

Chúng ta có thể hình dung về ứng dụng của phương thức roll() như sau: Giả sử hôm nay là tháng 12 năm 2017, nếu chúng ta tăng tháng của ngày hiện tại lên 1 thì kết quả sẽ là tháng 1 năm 2018. Trong trường hợp chúng ta muốn tăng tháng nhưng vẫn giữ nguyên năm thì Java cung cấp cho chúng ta phương thức roll() với cú pháp như sau:

**Cú pháp**

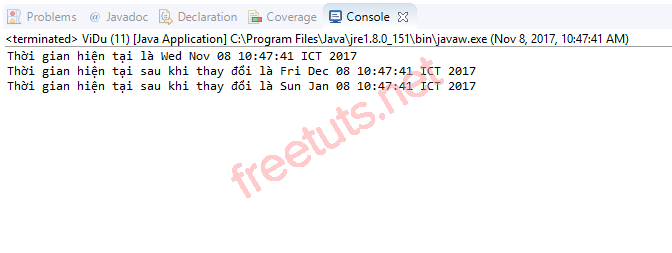
|  |  |
| --- | --- |
| 1  2 | Calendar cal = Calendar.getInstance();  cal.roll(int field, int amount); |

, trong đó field là tên giá trị cần thay đổi và amount là số lượng được thêm vào hay trừ đi. Nếu amount > 0 thì sẽ tăng giá trị field lên ứng với giá trị amount, còn amount < 0 thì sẽ giảm giá trị của field ứng với giá trị amount đó.

**Ví dụ**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13 | public static void main(String[] args) {      Calendar cal = Calendar.getInstance();      System.out.println("Thời gian hiện tại là " + cal.getTime());        // thay đổi tháng thành 12      cal.set(Calendar.MONTH, Calendar.DECEMBER);      System.out.println("Thời gian hiện tại sau khi thay đổi là " + cal.getTime());        // tăng tháng sử dụng phương thức roll()      // lúc này tháng sẽ là tháng 1 và năm vẫn giữ nguyên      cal.roll(Calendar.MONTH, 1);      System.out.println("Thời gian hiện tại sau khi thay đổi là " + cal.getTime());  } |

Kết quả sau khi biên dịch chương trình:



**2. SimpleDateFormat**

Trong Java, SimpleDateFormat được dùng để định dạng cách hiển thị ngày tháng. Để sử dụng được SimpleDateFormat, chúng ta cần phải import gói thư viện java.text.SimpleDateFormat của Java. Cú pháp import như sau:

**Cú pháp**

|  |  |
| --- | --- |
| 1  2  3  4 | import java.text.SimpleDateFormat;  public class TênClass {      // ...  } |

Dưới đây là ví dụ minh họa cách sử dụng SimpleDateFormat:

**Ví dụ**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37 | public static void main(String[] args) {      Calendar cal = Calendar.getInstance();      Date date = cal.getTime();        System.out.println("Thời gian hiện tại là " + date);        // khai báo 1 đối tượng SimpleDateFormat      // chi tiết về khai báo đối tượng chúng ta sẽ học trong chương "Lập trình hướng đối tượng"      // đối tượng này sẽ định dạng ngày theo cấu trúc "dd/MM/yyyy"      // trong đó "dd" là ngày, "MM" là tháng và "yyyy" là năm      // nếu ngày và tháng nào có 1 chữ số (ví dụ 1, 2, 3,...)      // thì sẽ được tự động thêm vào số 0 đằng trước      SimpleDateFormat sdf = new SimpleDateFormat("dd/MM/yyyy");        // hiển thị date theo cấu trúc mà chúng ta đã khai báo (dd/MM/yyyy)      // sử dụng phương thức format()      System.out.println("Thời gian hiện tại sau khi định dạng là " + sdf.format(date));        // định dạng ngày hiện tại theo cấu trúc "d/M/yyyy"      // trong đó "d" là ngày, "M" là tháng và "yyyy" là năm      // với kiểu định dạng này thì ngày và tháng nào có 1 chữ số (ví dụ 1, 2, 3,...)      // thì sẽ không được thêm vào số 0 đằng trước      SimpleDateFormat sdf1 = new SimpleDateFormat("d/M/yyyy");      System.out.println("Thời gian hiện tại sau khi định dạng là " + sdf1.format(date));        // định dạng ngày giờ hiện tại theo cấu trúc "dd/MM/yyyy HH:mm:ss"      // trong đó "HH" là giờ, "mm" là phút và "ss" là giây      // "HH" là định dạng 24 giờ, và "hh" là định dạng 12 giờ      SimpleDateFormat sdf2 = new SimpleDateFormat("dd/MM/yyyy HH:mm:ss");      System.out.println("Ngày giờ hiện tại sau khi định dạng là " + sdf2.format(date));        // hiển thị ngày giờ theo định dạng 12 giờ      // với định dạng 12 giờ thì chúng ta sẽ thêm vào "aaa"      // trong đó "aaa" đại diện cho AM/PM      SimpleDateFormat sdf3 = new SimpleDateFormat("dd/MM/yyyy hh:mm:ss aaa");      System.out.println("Ngày giờ hiện tại sau khi định dạng là " + sdf3.format(date));  } |

Kết quả sau khi biên dịch chương trình:

