javatpoint.com

Home **Core Java** **Servlet** **JSP** **Struts2** **Mail API** **Hibernate** **Spring** **Android** **Quiz** **Projects** **Interview Q** **Comment** **Forum**

Basics of Java
OOPs Concepts
String Handling
Exception Handling
Nested Classes
Multithreading
Synchronization
I/O
Serialization
Networking
AWT
Event Handling
Swing
LayoutManager
Applet
Reflection API
Collection
JDBC
Java New Features
RMI
Internationalization
Internationalization
ResourceBundle class
I18N with Date
I18N with Time
I18N with Number
I18N with Currency

<<prev   next>>

Internationalizing Time (I18N with Time)

The display format of the time differs from one region to another, so we need to internationalize the time. For internationalizing the time, the DateFormat class provides some useful methods. The getTimeInstance() method of the DateFormat class returns the instance of the DateFormat class for the specified style and locale. Syntax of the getTimeInstance() method is given below:

Internationalizing Time
Example of Internationalizing Time

```
public static DateFormat getTimeInstance(int style, Locale locale)
```

Example of Internationalizing Time

In this example, we are displaying the current time for the specified locale. The format() method of the DateFormat class receives date object and returns the formatted and localized time as a string. Notice that the object of Date class prints date and time both.

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

public class InternationalizingTime {

static void printTime(Locale locale){
DateFormat formatter=DateFormat.getTimeInstance(DateFormat.DEFAULT,locale);
Date currentDate=new Date();
String time=formatter.format(currentDate);
System.out.println(time+" in locale "+locale);
}

public static void main(String[] args) {
printTime(Locale.UK);
printTime(Locale.US);
printTime(Locale.FRANCE);
}
}
```

download this example

```
Output:16:22:49 in locale en_GB
4:22:49 PM in locale en_US
16:22:49 in locale fr_FR
```

<<prev   next>>