La Classe LocalTime en Java

[sommaire]

1. ## Objectifs

- Comprendre les bases de la classe LocalTime.
- Apprendre à créer et manipuler des objets LocalTime.
- Utiliser les méthodes pratiques pour les opérations courantes sur les heures.
- Apprendre à formater et à analyser les heures.

- ## Introduction à LocalTime

  - La classe LocalTime, introduite dans Java 8, fait partie de l'API java.time et est utilisée pour représenter une heure sans date et sans fuseau horaire. Comme les autres classes de l'API de date et d'heure, LocalTime est immuable et thread-safe.
    LocalTime est utilisé pour représenter une heure sans information de date. Cela inclut les heures, les minutes, les secondes et les nanosecondes.

- ## Création de LocalTime

  - Vous pouvez créer une instance de LocalTime de plusieurs façons :

1. **Utilisation de la méthode `now()`**

- Commençons par utiliser une méthode très utile, la **méthode now()** :

```
import java.time.LocalTime;
public class LocalTimeExample {
    public static void main(String[] args) {
        LocalTime now = LocalTime.now();
        System.out.println("Current Time: " + now);
    }
}
```

- **Utilisation de la méthode `of()`**

apcpedagogie.com

```
import java.time.LocalTime;

public class LocalTimeExample {
    public static void main(String[] args) {
        LocalTime specificTime = LocalTime.of(14, 30, 15);
        System.out.println("Specific Time: " + specificTime);
    }
}
```

- **Utilisation de la méthode `parse()`**

```
import java.time.LocalTime;

public class LocalTimeExample {
    public static void main(String[] args) {
        LocalTime parsedTime = LocalTime.parse("14:30:15");
        System.out.println("Parsed Time: " + parsedTime);
    }
}
```

Manipulation des Dates avec LocalTime

- LocalTime fournit diverses méthodes pour manipuler les heures, comme ajouter ou soustraire des heures, des minutes, des secondes, etc.

1. **Ajouter ou Soustraire des Heures, Minutes, Secondes**

```
import java.time.LocalTime;

public class LocalTimeExample {
    public static void main(String[] args) {
        LocalTime now = LocalTime.now();
        LocalTime plusHours = now.plusHours(2);
        LocalTime minusMinutes = now.minusMinutes(30);
        LocalTime plusSeconds = now.plusSeconds(45);
```

apcpedagogie.com

```
        System.out.println("Current Time: " + now);
        System.out.println("Plus 2 Hours: " + plusHours);
        System.out.println("Minus 30 Minutes: " + minusMinutes);
        System.out.println("Plus 45 Seconds: " + plusSeconds);
    }
}
```

- ## Comparaison des Heures

  - Vous pouvez comparer des heures en utilisant des méthodes telles que isBefore(), isAfter(), et equals().

```
import java.time.LocalTime;

public class LocalTimeExample {
    public static void main(String[] args) {
        LocalTime time1 = LocalTime.of(14, 30);
        LocalTime time2 = LocalTime.of(16, 45);

        System.out.println("Time1 is before Time2: " + time1.isBefore(time2));
        System.out.println("Time1 is after Time2: " + time1.isAfter(time2));
        System.out.println("Time1 is equal to Time2: " + time1.equals(time2));
    }
}
```

- ## Formatage et Analyse des Heures

  - LocalTime peut être formaté et analysé à l'aide de la classe DateTimeFormatter.

1. **Formatage de LocalTime**

```
import java.time.LocalTime;
import java.time.format.DateTimeFormatter;

public class LocalTimeExample {
    public static void main(String[] args) {
        LocalTime now = LocalTime.now();
```

apcpedagogie.com

```
        DateTimeFormatter formatter = DateTimeFormatter.ofPattern("HH:mm:ss");
        String formattedTime = now.format(formatter);

        System.out.println("Formatted Time: " + formattedTime);
    }
}
```

- **Analyse de LocalTime**

```
import java.time.LocalTime;
import java.time.format.DateTimeFormatter;

public class LocalTimeExample {
    public static void main(String[] args) {
        String timeString = "14:30:15";
        DateTimeFormatter formatter = DateTimeFormatter.ofPattern("HH:mm:ss");
        LocalTime parsedTime = LocalTime.parse(timeString, formatter);

        System.out.println("Parsed Time: " + parsedTime);
    }
}
```

Exemples Pratiques

- Calculer la Durée entre Deux Heures

```
import java.time.Duration;
import java.time.LocalTime;

public class LocalTimeExample {
    public static void main(String[] args) {
        LocalTime startTime = LocalTime.of(9, 0);
        LocalTime endTime = LocalTime.of(17, 0);
        Duration duration = Duration.between(startTime, endTime);

        System.out.println("Duration: " + duration.toHours() + " hours");
    }
}
```

apcpedagogie.com

- Déterminer si une Heure est dans une Plage Spécifique

```
import java.time.LocalTime;

public class LocalTimeExample {
    public static void main(String[] args) {
        LocalTime currentTime = LocalTime.now();
        LocalTime startTime = LocalTime.of(9, 0);
        LocalTime endTime = LocalTime.of(17, 0);

        boolean isWithinRange = !currentTime.isBefore(startTime) &&
!currentTime.isAfter(endTime);
        System.out.println("Current Time is within range: " + isWithinRange);
    }
}
```

Cours JAVA

Source : https://apcpedagogie.com/la-classe-localtime-en-java · téléchargé le 04/11/2026 · © APC Pédagogie · apcpedagogie.com

apcpedagogie.com