HowToDoInJava

Java Stream – Get Object with Max Date From a List

Last Updated: March 14, 2022 By: Lokesh Gupta Java 8 Java Stream Basics

Learn to **get an object with the latest date (max date)** from a Stream of custom objects. We will use a custom Comparator for comparing the *Date* values stored in the custom objects.

This example uses `Employee` class. We will create a program to get the youngest employee in a list of employees.

1. Custom Comparator for Comparing Objects by Date

The `LocalDate()` implements the Comparable interface so it automatically supports the correct comparison logic for comparing the two `LocalDate` objects. We do not need to write our own comparison logic for this date comparison.

We need to write a custom `Comparator` that can compare the custom objects and compare their `LocalDate` value.

The given `Comparator` compares two given `Employee` objects by their age i.e. Date of birth.

Custom Object Comparator

```
Comparator<Employee> employeeAgeComparator = Comparator
                  .comparing(Employee::getDateOfBirth);
```

> Use `Comparator.reversed()` method, we need to find the Min Date from the Stream.

2. Getting Object with Max date using Stream.max()

Now we know what to compare, let us create a program to create a stream of `Employee` Objects and then pass the custom `Comparator` to the `Stream.max()` method.

Find Youngest Employee from a List

```
import java.time.LocalDate;
import java.util.ArrayList;
import java.util.Comparator;
import java.util.List;

public class MaxDateExample
{
    public static void main(final String[] args)
    {
        Comparator<Employee> employeeAgeComparator = Comparator
                                .comparing(Employee::getDateOfBirth);

        Employee youngestEmployee = getEmployeeList().stream()
                                    .max(employeeAgeComparator)
                                    .get();

        System.out.println(youngestEmployee); //Prints Employee 'D'
    }

    private static List<Employee> getEmployeeList()
    {
        List<Employee> empList = new ArrayList<>();
        empList.add(new Employee(1, "A", LocalDate.of(1991, 1, 1), 30
        empList.add(new Employee(2, "B", LocalDate.of(1976, 7, 9), 30
        empList.add(new Employee(3, "C", LocalDate.of(1992, 8, 1), 50
        empList.add(new Employee(4, "D", LocalDate.of(2001, 3, 11), 5
        return empList;
    }
}
```

Program Output:

```
Employee [id=4, name=D, dateOfBirth=2001-03-11, salary=50000.0]
```

In this way, we can get a custom object from a `List` of objects while comparing the date values from one of its fields.

Happy Learning !!

Sourcecode on Github

Was this post helpful?

Let us know if you liked the post. That's the only way we can improve.

Yes

No

Recommended Reading:

1. Python max() and min() – finding max and min in list or array
2. Finding Max and Min from List using Streams
3. Java Stream max()

4. Collecting Stream Items into List in Java
5. Java Stream reuse – traverse stream multiple times?
6. Java Regex to check Min/Max Length of Input Text
7. Finding Max and Min in Arrays
8. Hibernate count, min, max, sum, avg Functions
9. Object level lock vs Class level lock in Java

0. Serialize Java Object to XML – XMLEncoder and XMLDecoder Example

Leave a Comment

Name *

Email *

Website

□ Add me to your newsletter and keep me updated whenever you publish new blog posts

Post Comment

Search …

HowToDoInJava

A blog about Java and related technologies, the best practices, algorithms, and interview questions.

Meta Links

- About Me
- Contact Us
- Privacy policy
- Advertise
- Guest Posts

Blogs

REST API Tutorial