Java-Success.com

Industrial strength Java/JEE Career Companion to open more doors

Register | Login | Logout | Contact Us

search here … Go

Home | Java FAQs | 600+ Java Q&As | Career | Tutorials | Member | Why?

Can u Debug? | Java 8 ready? | Top X | Productivity Tools | Judging Experience?

Home › Interview › Core Java Interview Q&A › Collection and Data structures ›
Java 8: Different ways to sort a collection of objects in pre and post Java 8

Java 8: Different ways to sort a collection of objects in pre and post Java 8

Posted on November 8, 2014 by Arulkumaran Kumaraswamipillai — No Comments ↓

The object we are going to sort is a *Person*.

```
public class Person  {

 public enum Gender {FEMALE, MALE};

 private String name;
 private Integer age;
 private Gender gender;

 public Person(String name, Integer age, Gender 
  this.name = name;
  this.age = age;
  this.gender = gender;
 }

 //getter, setter, equals(...), and hashCode() m

 @Override
 public String toString() {
  return "Person [name=" + name + ", age=" + age
 }
}

```

Option 1: Writing your own **Comparator** implementation. This can be done as an anonymous inner class instead of a separate class.

```
import java.util.Comparator;

public class PersonComparator implements Compara

 @Override
 public int compare(Person o1, Person o2) {

  //by gender first
  int i1 = o1.getGender().compareTo(o2.getGender
  if (i1 != 0) return i1;

  //by name next
  int i2 =  o1.getName().compareTo(o2.getName())
  if (i2 != 0) return i2;

  //by age
  return o1.getAge().compareTo(o2.getAge());
 }
}

```

The test class

```
import java.util.ArrayList;
import java.util.List;

public class PersonTest {

 public static void main(String[] args) {

  List<Person> people = new ArrayList<>();
  people.add(new Person("John", 35, Person.Gende
  people.add(new Person("John", 32, Person.Gende
  people.add(new Person("Simone", 30, Person.Gen
  people.add(new Person("Shawn", 30, Person.Gend

  System.out.println("before sorting = "  + peop
  people.sort(new PersonComparator());
  System.out.println("after sorting = "  + peopl

 }
}

```

16 Technical Key Areas

Option 2: The Option 1 is not bad, but the the moment you need to handle null element values, the *PersonComparator* will have more code. One of the best practices in Java is "Don't reinvent the wheel". So, let's use the *BeanComparator*,*NullComparator*, and *ComparatorChain* from the **Apache commons library** commons-beanutils -> commons-beanutils-bean-collections that uses **reflection**. The example below also handles null values.

```
import java.util.ArrayList;
import java.util.List;

import org.apache.commons.beanutils.BeanComparat
import org.apache.commons.collections.comparator
import org.apache.commons.collections.comparator

public class PersonTest {

 public static void main(String[] args) {

  List<Person> people = new ArrayList<>();
  people.add(new Person("John", 35, Person.Gende
  people.add(new Person("John", 32, Person.Gende
  people.add(new Person("Simone", 30, Person.Gen
  people.add(new Person("Shawn", 30, Person.Gend
  people.add(new Person("Shawn", 30, null));

  System.out.println("before sorting = "  + peop

  //Apache commons-beanutils.commons-beanutils-b
  ComparatorChain comparatorChain = new Comparat
  //null is compared s lower
  comparatorChain.addComparator(new BeanComparat
  //null is compared as higher
  comparatorChain.addComparator(new BeanComparat
  comparatorChain.addComparator(new BeanComparat
  people.sort(comparatorChain);

  System.out.println("after sorting = "  + peopl

 }
}

```

Option 3: Using the **Google Gauva library** to sort the collection in **functional programming** style.

```
import java.util.ArrayList;
```

- ⊞ Low Latency (7)
- ⊞ Memory Managemen
- ⊞ Performance (13)
- ⊞ QoS (8)
- ⊞ Scalability (4)
- ⊞ SDLC (6)
- ⊞ Security (13)
- ⊞ Transaction Managem

80+ step by step Java Tutorials

open all | close all

- ⊞ Setting up Tutorial (6)
- ⊞ Tutorial - Diagnosis (2
- ⊞ Akka Tutorial (9)
- ⊞ Core Java Tutorials (2
- ⊞ Hadoop & Spark Tuto
- ⊞ JEE Tutorials (19)
- ⊞ Scala Tutorials (1)
- ⊞ Spring & HIbernate Tu
- ⊞ Tools Tutorials (19)
- ⊞ Other Tutorials (45)

100+ Java pre-interview coding tests

open all | close all

- ⊞ Can you write code?
- ⊞ ♦ Complete the given
- ⊞ Converting from A to I
- ⊞ Designing your classe
- ⊞ Java Data Structures
- ⊞ Passing the unit tests
- ⊞ What is wrong with th
- ⊞ Writing Code Home A
- ⊞ Written Test Core Jav

```
import java.util.Comparator;
import java.util.List;

import com.google.common.collect.ComparisonChain
import com.google.common.collect.Ordering;

public class PersonTest {

 public static void main(String[] args) {

  List<Person> people = new ArrayList<>();
  people.add(new Person("John", 35, Person.Gende
  people.add(new Person("John", 32, Person.Gende
  people.add(new Person("Simone", 30, Person.Gen
  people.add(new Person("Shawn", 30, Person.Gend
  people.add(new Person("Shawn", 30, null));

  System.out.println("before sorting = "  + peop

  //anonymous inner class using the Google Gauva
  people.sort(new Comparator<Person>() {

   @Override
   public int compare(Person o1, Person o2) {
    return ComparisonChain.start()
          .compare(o1.getGender(), o2.getGender(
          .compare(o1.getName(), o2.getName(), O
          .compare(o1.getAge(), o2.getAge(), Ord
          .result();
   }


  });

  System.out.println("after sorting = "  + peopl

 }
}

```

⊞ Written Test JEE (1)

How good are your .....?

open all | close all

⊞ Career Making Know-

⊞ Job Hunting & Resum

Option 4: If you are using **Java 8**, using the **functional programming** approach.

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

public class PersonTest {

 public static void main(String[] args) {

  List<Person> people = new ArrayList<>();
  people.add(new Person("John", 35, Person.Gende
  people.add(new Person("John", 32, Person.Gende
  people.add(new Person("Simone", 30, Person.Gen
  people.add(new Person("Shawn", 30, Person.Gend
  people.add(new Person("Shawn", 30, null));

  System.out.println("before sorting = "  + peop

  //java 8 approach fro multi-fields
  Comparator<Person> multiFieldComparator =
            Comparator.comparing(Person::getGend
                      .thenComparing(Person::get
                      .thenComparing(Person::get

  people.sort(multiFieldComparator);
  System.out.println("after sorting = "  + peopl

 }
}

```

Option 5: If you are using **Java 8**, using **parallel processing**. Very similar to option 4, but processed in parallel with minor changes.

```
import java.util.ArrayList;
import java.util.Comparator;
import java.util.List;
import java.util.stream.Collectors;

public class PersonTest {

 public static void main(String[] args) {

  List<Person> people = new ArrayList<>();
  people.add(new Person("John", 35, Person.Gende
  people.add(new Person("John", 32, Person.Gende
  people.add(new Person("Simone", 30, Person.Gen
  people.add(new Person("Shawn", 30, Person.Gend
  people.add(new Person("Shawn", 30, null));

  System.out.println("before sorting = "  + peop

  Comparator<Person> multiFieldComparator =
          Comparator.comparing(Person::getGender
                    .thenComparing(Person::getNa
                    .thenComparing(Person::getAg

     //parallel() processing using Fork/Join
  List<Object> sortedPeople = people.stream()
                                          .paral
                                          .sorte
                                          .colle

  System.out.println("after sorting = "  + sorte

 }
}

```

Output:

before sorting = [Person [name=John, age=35, gender=MALE], Person [name=John, age=32, gender=MALE], Person [name=Simone, age=30, gender=FEMALE], Person [name=Shawn, age=30,

```
gender=MALE], Person [name=Shawn, age=30,
gender=null]]

after sorting = [Person [name=Shawn, age=30, gender=null],
Person [name=Simone, age=30, gender=FEMALE], Person
[name=John, age=32, gender=MALE], Person [name=John,
age=35, gender=MALE], Person [name=Shawn, age=30,
gender=MALE]]
```

Popular Posts

- ♦ 11 Spring boot interview questions & answers
  823 views
- ♦ Q11-Q23: Top 50+ Core on Java OOP Interview Questions & Answers
  765 views
- 18 Java scenarios based interview Questions and Answers
  399 views
- 001A: ♦ 7+ Java integration styles & patterns interview questions & answers
  388 views
- 01b: ♦ 13 Spring basics Q8 – Q13 interview questions & answers
  295 views
- ♦ 7 Java debugging interview questions & answers
  293 views
- 01: ♦ 15 Ice breaker questions asked 90% of the time in Java job interviews with hints
  285 views
- ♦ 10 ERD (Entity-Relationship Diagrams) Interview Questions and Answers
  279 views
- ♦ Q24-Q36: Top 50+ Core on Java classes, interfaces and generics interview questions & answers
  239 views
- 001B: ♦ Java architecture & design concepts interview questions & answers
  201 views

Bio | Latest Posts

Arulkumaran Kumaraswamipillai

Mechanical Eng to freelance Java developer in 3 yrs. Contracting since 2003, and attended 150+ Java job interviews, and often got 4 - 7 job offers to choose from. It pays to prepare. So, published Java interview Q&A books via Amazon.com in 2005, and sold 35,000+ copies. Books are outdated and replaced with this subscription based site.**945+** paid members. join my LinkedIn Group. **Reviews**

About Arulkumaran Kumaraswamipillai

Mechanical Eng to freelance Java developer in 3 yrs. Contracting since 2003, and attended 150+ Java job interviews, and often got 4 - 7 job offers to choose from. It pays to prepare. So, published Java interview Q&A books via Amazon.com in 2005, and sold 35,000+ copies. Books are outdated and replaced with this subscription based site.**945+** paid members. join my LinkedIn Group. **Reviews**

‹ Top 6 Java 8 features you can start using now

Java 7: Top 8 new features with examples ›

Posted in Collection and Data structures, Java 8, member-paid

Leave a Reply

Logged in as geethika. Log out?

Comment

Post Comment

Empowers you to open more doors, and fast-track

Technical Know Hows

☀ Java generics in no time ☀ Top 6 tips to transforming your thinking from OOP to FP ☀ How does a HashMap internally work? What is a hashing function?
☀ 10+ Java String class interview Q&As ☀ Java auto un/boxing benefits & caveats ☀ Top 11 slacknesses that can come back and bite you as an experienced Java developer or architect

Non-Technical Know Hows

☀ 6 Aspects that can motivate you to fast-track your career & go places ☀ Are you reinventing yourself as a Java developer? ☀ 8 tips to safeguard your Java career against offshoring ☀ My top 5 career mistakes

Prepare to succeed

☀ Turn readers of your Java CV go from "Blah blah" to "Wow"? ☀ How to prepare for Java job interviews? ☀ 16 Technical Key Areas ☀ How to choose from multiple Java job offers?

Select Category ▼

© Disclaimer