Theory: What are collections

9 minutes 0 / 5 problems solved

§1. When arrays are not enough

The Java language supports arrays to store multiple values or objects of the same type together. An array is initialized with a predefined size during creation. The size cannot be changed in the future, and that imposes some limitations on their use for solving business problems. If we want to store more data, we need to create a new larger array and then copy the data in this array manually. This can be inefficient for programs that process a lot of data.

§2. Different collections

Fortunately, there is a set of containers called **collections** for grouping elements into a single unit. They are used to store, retrieve, manipulate, and communicate aggregated data.

Collections are more sophisticated and flexible than arrays. First of all, they are **resizable**: you can add any number of elements to a collection. A collection will automatically handle the deletion of an element from any position. The second point is collections provide a rich set of **methods** that are already implemented for you.

There are several types of collections with different internal storage structure. You can choose a collection type best matching your problem so that your most frequent operations will be convenient and efficient.

> Actually, collections are representations of different data structures and abstract data types from Computer Science. It is good to understand the relationship between them and collections in Java. This will help you in programming interviews, and in working to select an appropriate collection.

§3. Features of collections

There are several specific features of collections in Java:

1. They are represented by different classes from the Java Standard Library.
2. All modern collections are **generic types** while old collections are **non-generic**. We will only focus on new collections. As regular generics, they can store any reference types including classes defined by you (like `Person` or something else).
3. Collections can be **mutable** (possible to add and remove elements) and **immutable** (impossible to do that).

In addition to standard collections, there are a number of external libraries with collections. One of such libraries is Guava Collections which was developed by Google. It can be used if standard collections are not enough for solving your problems.

§4. The simplest collection example

There is an example of a simple collection called `ArrayList`. To use it, make the following import:

```
1 java.util.ArrayList;
```

It works in a similar way to a regular array, but you do not have to manually resize it to add and remove elements.

Current topic:

What are collections

Topic depends on:

✓ Data structures

✗ Generic programming

Topic is required for:

ArrayList

The Collections Framework overview

Collections and thread-safety

Table of contents:

↑ What are collections

§1. When arrays are not enough

§2. Different collections

§3. Features of collections

§4. The simplest collection example

§5. Conclusion

Feedback & Comments

```
ArrayList<String> list = new ArrayList<>();

list.add("first");
list.add("second");
list.add("third");

System.out.println(list); // [first, second, third]

System.out.println(list.get(0)); // first

System.out.println(list.get(1)); // second

System.out.println(list.get(2)); // third



list.remove("first");



System.out.println(list); // [second, third]



System.out.println(list.size()); // 2
```

Note, in this example we used the `get` method to access an element by its index. Unlike arrays, collections do not have the `[]` operator.

We hope this is enough for the first acquaintance with the collections. In further topics, you will learn different kinds of collections in more detail. Now the main thing to understand is that using collections is not more difficult than using a regular array.

All modern collections are generic, so you can specify any reference type as a generic parameter and store it in a collection. But there is one restriction, collections cannot store primitive values at all (`int`, `long`, `char`, `double` and so on). You should use one of the wrapper classes (`Integer`, `Long`, `Character`, `Double` or another one) instead.

§5. Conclusion

Sometimes arrays are not flexible enough to store and manipulate your data. For that, Java provides collections — mostly generic classes from the Java Standard Library or external libraries, either mutable or immutable, that are more adjusted to store objects for specific complicated purposes. It'll take you more than one topic to learn even about basic types of collections that represent different data structures and serve different purposes. Stay tuned!

Report a typo

399 users liked this theory. **1** didn't like it. **What about you?**

Start practicing

This content was created over 2 years ago and updated 10 days ago. Share your feedback below in comments to help us improve it!

Comments (9) Hints (0) Useful links (1) Show discussion