



#R4E

Software Developer

Paradigmas de Programação

Aula 05 - ArrayList

Vitor Santos



Conteúdo



- Java ArrayList
- Java ArrayList vs Array Tradicional
- Sintaxe do ArrayList
- Adicionar, Editar e Remover Elementos
- ArrayList Size
- Iterar um ArrayList
- Ordenar um ArrayList

Java ArrayList

- A classe ArrayList é um array redimensionável, que pode ser encontrado no pacote java.util.

Java ArrayList vs Array

- A diferença entre um array *built-in* e um ArrayList em Java, é que o tamanho de um array não pode ser modificado (se quisermos adicionar ou remover elementos de/para um array, deve ser criado um novo). Enquanto os elementos podem ser adicionados e removidos de um ArrayList sempre que quisermos. A sintaxe também é um pouco diferente:

```
// import the ArrayList class
import java.util.ArrayList;
// Create an ArrayList object
ArrayList<TipoDados> nome_array = new ArrayList<TipoDados>();
```

Exemplo de Declaração

```
// import the ArrayList class  
import java.util.ArrayList;
```

```
// Create an ArrayList object  
ArrayList<String> cars = new ArrayList<String>();
```

```
ArrayList<Pessoa> agenda = new ArrayList<Pessoa>();
```

```
ArrayList<Atleta> competicao = new ArrayList<Atleta>();
```

Adicionar Elementos

- A classe ArrayList tem muitos métodos úteis. Por exemplo, para adicionar elementos ao ArrayList, usamos o método add():

```
import java.util.ArrayList;

public class Main {public static void main(String[] args){
    ArrayList<String> cars = new ArrayList<String>();
    cars.add("Volvo");
    cars.add("BMW");
    cars.add("Ford");
    cars.add("Mazda");
    System.out.println(cars);
}
```

Aceder um Elemento

- Para aceder um elemento no ArrayList, usamos o método `get()` e consulte o número do índice:
`cars.get(0);`

Editar um Elemento

- Para modificar um elemento, usamos o método `set()` e indicamos o número do índice:
`cars.set(0, "Opel");`

Remover um Elemento

- Para remover um elemento, usamos o método `remove()` e indicamos o número do índice:
`cars.remove(0);`

Remover todos os Elementos

- Para remover todos os elementos do ArrayList, usamos o método `clear()`:
`cars.clear();`

ArrayList Size

- Para descobrir quantos elementos um ArrayList tem, usamos o método size:
`cars.size();`

Iterar um ArrayList

- Percorremos os elementos de um ArrayList com um loop for e usamos o método size() para especificar quantas vezes o loop deve ser executado:

```
public class Main {  
    public static void main(String[] args){  
        ArrayList<String> cars = new ArrayList<String>();  
        cars.add("Volvo");  
        cars.add("BMW");  
        cars.add("Ford");  
        cars.add("Mazda");  
        for (int i = 0; i < cars.size(); i++) {  
            System.out.println(cars.get(i));  
        }  
    }  
}
```

Iterar um ArrayList

- Também podemos percorrer um ArrayList com o loop for-each:

```
public class Main {  
    public static void main(String[] args){  
        ArrayList<String> cars = new ArrayList<String>();  
        cars.add("Volvo");  
        cars.add("BMW");  
        cars.add("Ford");  
        cars.add("Mazda");  
        for (String carroAtual : cars) {  
            System.out.println(carroAtual);  
        }  
    }  
}
```

Ordenar um ArrayList

- Outra classe útil no pacote java.util é a classe Collections, que inclui o método sort() para ordenar listas em ordem alfabética ou numérica:

```
public class Main {  
    public static void main(String[] args){  
        ArrayList<String> cars = new ArrayList<String>();  
        cars.add("Volvo");  
        cars.add("BMW");  
        cars.add("Ford");  
        cars.add("Mazda");  
        Collections.sort(cars); // Sort cars  
        for (String carroAtual : cars) {  
            System.out.println(carroAtual);  
        }  
    }  
}
```

Ordenar um ArrayList

Outra classe útil no pacote `java.util` é a classe `Collections`, que inclui o método `sort()` para ordenar listas em ordem alfabética ou numérica:

```
public class Main {  
    public static void main(String[] args){  
        ArrayList<Integer> myNumbers = new ArrayList<Integer>();  
        myNumbers.add(33);  
        myNumbers.add(15);  
        myNumbers.add(20);  
        myNumbers.add(34);  
        myNumbers.add(8);  
        myNumbers.add(12);  
        Collections.sort(myNumbers); // Sort myNumbers  
        for (int i : myNumbers) {  
            System.out.println(i);  
        }  
    }  
}
```

Lista Métodos ArrayList

Java ArrayList add()

inserts the element to the arraylist

Java ArrayList addAll()

adds all elements of a collection to arraylist

Java ArrayList clear()

removes all the elements from arraylist

Java ArrayList clone()

makes a copy of the array list

Java ArrayList contains()

checks if the element is present in the arraylist

Java ArrayList get()

returns the element present in the specified index

Java ArrayList indexOf()

returns the position of the specified element

Java ArrayList removeAll()

removes multiple elements from the arraylist

Java ArrayList remove()

removes the single element from the arraylist

Java ArrayList size()

returns the length of an arraylist

Java ArrayList isEmpty()

checks if the arraylist is empty

Java ArrayList subList()

returns a portion of the arraylist

Java ArrayList set()

replace the single element from an arraylist

Java ArrayList sort()

sorts the arraylist according to specified order

Java ArrayList toArray()

converts an arraylist to an array

Java ArrayList toString()

converts the arraylist into a String

Java ArrayList ensureCapacity()

set the size of an arraylist

Java ArrayList lastIndexOf()

returns position of last occurrence of the element

Java ArrayList retainAll()

retains only the common elements

Java ArrayList containsAll()

checks if a collection is a subset of arraylist

Java ArrayList trimToSize()

trims the capacity of arraylist equal to the size

Java ArrayList removeRange()

removes a portion of the arraylist

Java ArrayList replaceAll()

replace all elements from the arraylist

Java ArrayList removeIf()

removes element that satisfy the condition

Java ArrayList forEach()

performs an action to all elements of arraylist

Java ArrayList iterator()

returns an iterate to loop through the ArrayList



#R4E

Software Developer

Paradigmas de Programação

Aula 05 - ArrayList

Vitor Santos

