



**INSTITUTO FEDERAL DE
EDUCAÇÃO, CIÊNCIA E TECNOLOGIA**
RIO GRANDE DO SUL
Câmpus Feliz

ArrayList

Prof. Moser Fagundes

Programação II
Técnico em Informática

Classe ArrayList

- O Java fornece várias estruturas de dados predefinidas, chamadas **coleções**, usadas para armazenar grupos de objetos relacionados.
- Como exemplo, temos a classe **ArrayList**.

Classe ArrayList

- Os arrays que estudamos até agora **não alteram automaticamente o seu tamanho** em tempo de execução para acomodar elementos adicionais.
- A classe de coleção **ArrayList<T>** fornece uma solução conveniente para este problema. Nesta classe, o **T** é um espaço reservado que indica o **tipo de elemento** que vamos ter no **ArrayList**.
- Exemplos:

```
ArrayList<String> lista1;  
ArrayList<Integer> numeros;  
ArrayList<Pessoas> grupo;
```

Classe ArrayList

Métodos da classe **ArrayList**:

- **add** para adicionar um elemento
- **clear** para remover todos elementos
- **get** para retornar um elemento específico
- **remove** para remover elementos
- **size** para obter o número de elementos
- **contains** para verificar se um objeto está no arraylist
- **isEmpty** verifica se o arraylist está vazio
- **addAll** adiciona uma coleção no arraylist
- **removeAll** remove uma coleção do arraylist

add, size e get

```
package exemplo;

import java.util.ArrayList;

public class ArrayListCollection {

    public static void main(String[] args) {

        ArrayList<String> itens = new ArrayList<String>();

        // Adiciona red na ultima posicao
        itens.add("red");

        // Adiciona yellow na primeira posicao
        itens.add(0,"yellow");

        // Adiciona blue na segunda posicao
        itens.add(1,"blue");

        // Percorre todos itens, imprimindo eles na tela
        for (int i = 0; i < itens.size(); i++)
            System.out.println(itens.get(i));
    }
}
```

clear

Remove todos
os elementos

```
package exemplo;

import java.util.ArrayList;

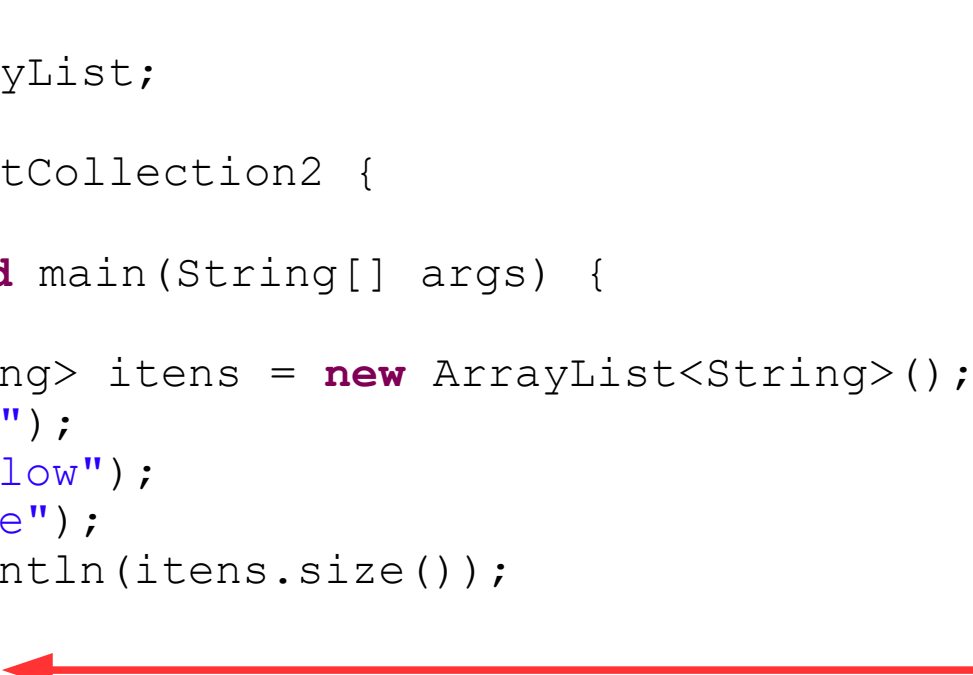
public class ArrayListCollection2 {

    public static void main(String[] args) {

        ArrayList<String> itens = new ArrayList<String>();
        itens.add("red");
        itens.add("yellow");
        itens.add("blue");
        System.out.println(itens.size());

        itens.clear();
        System.out.println(itens.size());

    }
}
```



remove

```
package exemplo;

import java.util.ArrayList;

public class ArrayListCollection3 {

    public static void main(String[] args) {

        ArrayList<String> itens = new ArrayList<String>();
        itens.add("red");
        itens.add("yellow");
        itens.add("blue");
        itens.add("white");

        itens.remove(1);
        itens.remove("blue");

        for (int i = 0; i < itens.size(); i++)
            System.out.println(itens.get(i));
    }
}
```

Podemos remover um elemento tanto pelo seu índice como pelo seu objeto.



addAll

```
package exemplo;

import java.util.ArrayList;

public class ArrayListCollection {

    public static void main(String[] args) {

        ArrayList<Integer> x = new ArrayList<Integer>();
        x.add(1);
        x.add(2);
        x.add(3);
        x.add(4);

        ArrayList<Integer> x2 = new ArrayList<Integer>();
        x2.add(5);
        x2.add(6);

        x.addAll(x2);

        for (Integer i : x) {
            System.out.print(i + " ");
        }
    }
}
```


removeAll

```
package exemplo;

import java.util.ArrayList;

public class ArrayListCollection {

    public static void main(String[] args) {

        ArrayList<Integer> x = new ArrayList<Integer>();
        x.add(1);
        x.add(2);
        x.add(3);
        x.add(4);

        ArrayList<Integer> x2 = new ArrayList<Integer>();
        x2.add(2);
        x2.add(3);

        x.removeAll(x2);

        for (Integer i : x) {
            System.out.print(i + " ");
        }
    }
}
```

Exemplos com outras classes

```
ArrayList<Aluno> turma;  
ArrayList<Carro> garagem;
```

```
// Implementados no Eclipse
```

Comparativo

arrays (vetores)

Capacidade fixa.

Acessado com **z[i];**

Construtor: `new double[30];`
`new Pessoa[20];` etc.

Obtemos seu tamanho com **z.length;**

Não possui métodos.

Mais rápido e menos flexível.

ArrayList

Capacidade aumenta a medida que novos elementos são adicionados.

Acessado com **z.get(i);**

Construtor: `new ArrayList<Double>();`
`new ArrayList<Pessoa>();`

Obtemos seu tamanho com **z.size();**

Muitos métodos: `z.add();`
`z.get();` etc.

Mais lento e mais flexível.

Exercícios

Lista no Moodle!