Interação entre objetos

Instituto Metrópole Digital

Disciplina: IMD0040 - Linguagem de Programação II

Docente: Emerson Alencar

ArrayList

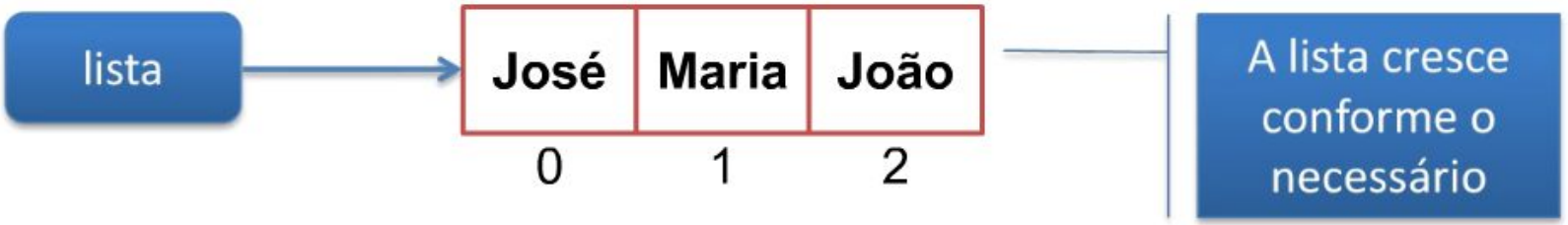
- É a implementação de listas mais utilizada;
- Trabalha internamente com um array

```
List l = new ArrayList();
```

ArrayList

- Usando o método add(), podemos adicionar elementos no fim da lista ou em uma posição qualquer;

```
List lista = new ArrayList();
lista.add("José");
lista.add("João");
lista.add(1, "Maria");
```



ArrayList

- O método size() retorna o tamanho da lista

```
int t = lista.size();
```

- O método get() retorna o elemento da posição especificada

```
Object item = lista.get(1);
```

ArrayList

- Todas as coleções são genéricas
- Trabalham apenas com tipos Object
- É preciso fazer casting da referência ao obter um elemento

```
String nome = (String) lista.get(1);
```

Percorrendo Listas

- Usando o iterator

```
Iterator iter = lista.iterator();

while(iter.hasNext()) {
    String nome = (String) iter.next();
    ...
}
```

- Usando o enhanced-for

```
for(Object obj : lista) {
    String nome = (String) obj;
    ...
}
```

Usando Generics com Listas

- Permite restringir os tipos de dados em coleções
- Vantagens
  - Evita casting, que pode ser feito de forma errada
  - Faz a verificação do tipo de dado em tempo de compilação

Usando Generics com Listas

Usando Generics com Listas

- Iterar sobre listas que usam generics é mais simples

```
Iterator<String> iter = lista.iterator();

while(iter.hasNext()) {
    String nome = iter.next();
    ...
}
```

```
for(String nome : lista) {
    ...
}
```

Ordenação de Listas

- A ordenação possibilita que os elementos fiquem posicionados de acordo com algum critério
- A classe Collections traz um método estático sort() para fazer ordenação de listas

```
Collections.sort(lista);
```

Ordenação de Listas

- A ordenação só funciona em um dos seguintes casos
  - Se os elementos da coleção implementarem a interface java.lang.Comparable
  - Se um java.util.Comparator for utilizado
- A utilização de uma dessas interfaces obriga o programador a implementar a regra de como os elementos serão ordenados

Distinção de Elementos

- Como especificar quais objetos são iguais?
  - equals()
- Métodos pertecem à classe Object
- A implementação da classe Object compara referências de memória

Exemplo de equals()

```
public class Linguagem {

  private String nome;
  private String descricao;

  @Override
  public boolean equals(Object obj) {
      if (this == obj)
          return true;
      if (obj == null)
          return false;
      if (getClass() != obj.getClass())
          return false;
      Linguagem other = (Linguagem) obj;
      if (nome == null) {
          if (other.nome != null)
              return false;
      } else if (!nome.equals(other.nome))
          return false;
      return true;
  }
}
```

Por hoje é só...