

Programação Orientada a Objetos

ITE - 003

profº Mauricio Conceição Mario

Conceitos de Subtipagem

- Método `super(...)`: utilizado quando deseja-se executar um método na superclasse que está sendo sobreposto na subclasse ou quando deseja-se inicializar o método construtor na superclasse.

Conceitos de Subtipagem

- Identificador de um objeto: é uma variável que conterá o endereço deste. A linguagem Java considera que um identificador de objeto de determinada classe é compatível com instâncias da própria classe ou com qualquer instância de uma classe descendente. A compatibilidade de endereços identificadores de uma classe com instâncias de subclasses é denominada **Subtipagem**.

Conceitos de Subtipagem

Classe Interface_Tipagem

- //construção do objeto que representará a interface tipagem

```
Interface_Tipagem IT = new Interface_Tipagem() ;
```

- //objeto da superclasse Cidade

```
Cidade CD ;
```

- //uso do método locais_cidade da interface através do objeto da superclasse

```
CD = IT.locais_cidade() ;
```

- //no método da interface locais_cidade é definido de que classe é o objeto

```
IT.tipo_local(CD) ;
```

Conceitos de Subtipagem

Classe Interface_Tipagem

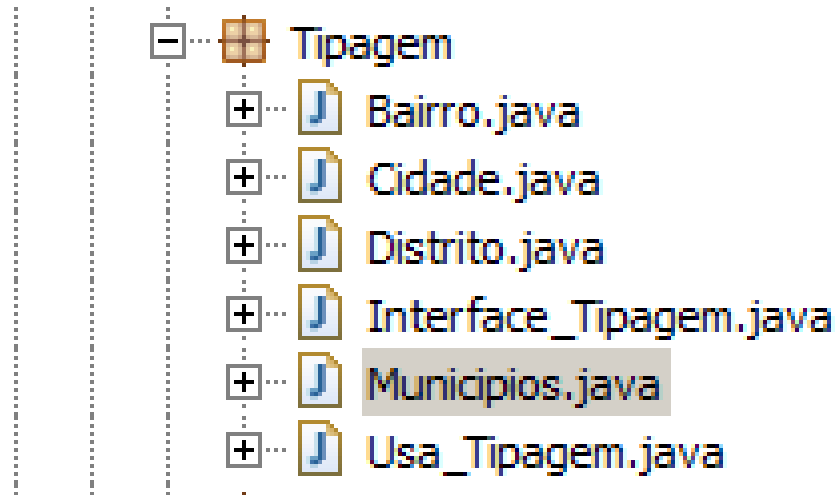
- //operador **instanceof**

if(objeto instanceof classe)

verifica se uma instância de objeto pertence ou não de uma determinada classe

Conceitos de Subtipagem

Pacote Tipagem



Conceitos de Subtipagem

Superclasse Municipios

```
package Tipagem;

public class Municipios {

    protected String nome_local;
    protected int habitantes;

    Municipios() {}

    Municipios(String nome_local, int habitantes) {

    }

}
```

Conceitos de Subtipagem

Classe Cidade

```
package Tipagem;

public class Cidade extends Municipios {

    protected String nome_local;
    protected int habitantes;

    Cidade (String nome_local, int habitantes){
        //inicialmente é chamado o construtor da superclasse
        super(nome_local, habitantes);
    }

    public void setNome(String nome_local){
        this.nome_local = nome_local;
    }

    public void setHabitantes(int habitantes){
        this.habitantes = habitantes;
    }

    public String getNome(){
        return nome_local + "_city";
    }

    public int getHabitantes(){
        return habitantes = habitantes*1000;
    }
}
```


Conceitos de Subtipagem

Classe Bairro

```
package Tipagem;

public class Bairro extends Cidade {

    //método construtor de Bairro
    Bairro (String nome_local, int habitantes){

        //ativação do método construtor da superclasse
        super(nome_local, habitantes);
    }

    public void setNomeBairro(String nome_local){
        this.nome_local = nome_local + "_area";
    }

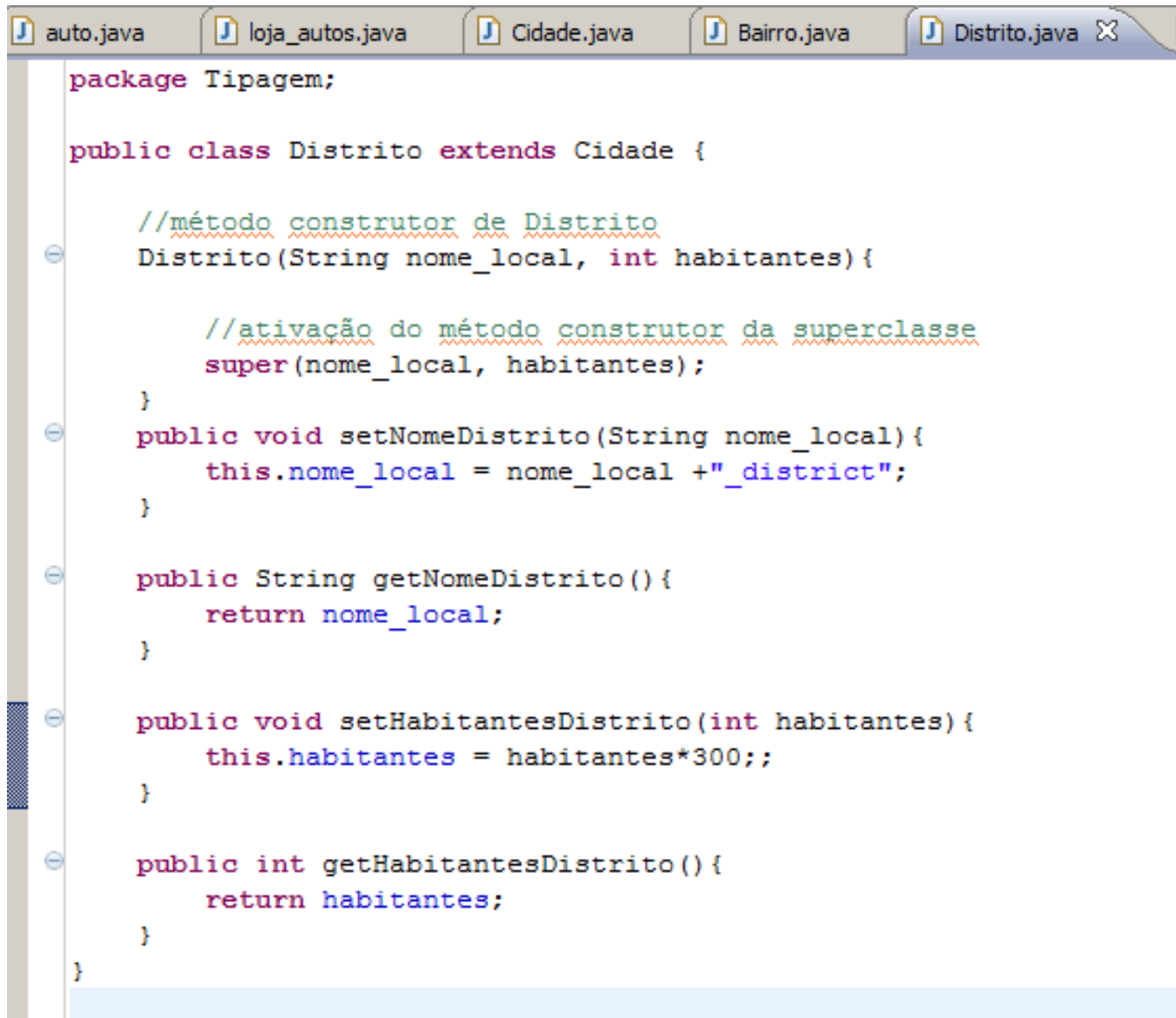
    public String getNomeBairro(){
        return nome_local;
    }

    public void setHabitantesBairro(int habitantes){
        this.habitantes = habitantes*200;;
    }

    public int getHabitantesBairro(){
        return habitantes;
    }
}
```

Conceitos de Subtipagem

Classe Distrito



```
package Tipagem;

public class Distrito extends Cidade {

    //método construtor de Distrito
    Distrito(String nome_local, int habitantes){

        //ativação do método construtor da superclasse
        super(nome_local, habitantes);
    }

    public void setNomeDistrito(String nome_local){
        this.nome_local = nome_local + "_district";
    }

    public String getNomeDistrito(){
        return nome_local;
    }

    public void setHabitantesDistrito(int habitantes){
        this.habitantes = habitantes*300;;
    }

    public int getHabitantesDistrito(){
        return habitantes;
    }

}
```

Conceitos de Subtipagem

Classe Interface_Tipagem

```
package Tipagem;

import javax.swing.*.*;

public class Interface_Tipagem {
    String p;
    int q;
    Cidade r = null;
    /*método do tipo da superclasse */
    public Cidade locais_cidade() {

        p = JOptionPane.showInputDialog("DIGITAR NOME DO LOCAL");
        q = Integer.parseInt(JOptionPane.showInputDialog("DIGITAR A QUANTIDADE DE HABITANTES"));
        String local = JOptionPane.showInputDialog("cidade = c n/ bairro = b n/ distrito = d /n ");
        char tipo = local.charAt(0);

        if (tipo == 'c')
            return new Cidade(p,q);

        if (tipo == 'b')
            return r = new Bairro(p,q);

        else if (tipo == 'd')
            return r = new Distrito(p,q);

        else
            return null;
    }
}
```

Conceitos de Subtipagem

Classe Interface_Tipagem

```
public void tipo_local(Cidade f){

    //o operador instanceof permite definir se uma instância é ou não de determinada classe
    if (f instanceof Bairro){
        ((Bairro)f).setNomeBairro(p); ((Bairro)f).setHabitantesBairro(q);
        System.out.println("local é bairro" + "\n nome do bairro = " + ((Bairro)f).getNomeBairro() +
            "\n número de habitantes = " + ((Bairro)f).getHabitantesBairro());
    }

    else if (f instanceof Distrito){
        ((Distrito)f).setNomeDistrito(p); ((Distrito)f).setHabitantesDistrito(q);
        System.out.println("local é distrito" + "\n nome do distrito = " + ((Distrito)f).getNomeDistrito() +
            "\n número de habitantes = " + ((Distrito)f).getHabitantesDistrito());
    }
    else{
        f.setNome(p); f.setHabitantes(q);
        System.out.println("local é cidade" + "\n nome da cidade = " + ((Cidade)f).getNome() +
            "\n número de habitantes = " + ((Cidade)f).getHabitantes());
    }

    //System.out.println("localidade indefinida");
}
```

Conceitos de Subtipagem

Classe Usa_Tipagem

```
package Tipagem;

public class Usa_Tipagem {

    public static void main (String[] parametros){

        //construção do objeto que representará a interface tipagem
        Interface_Tipagem IT = new Interface_Tipagem();

        //objeto da superclasse Cidade
        Cidade CD;

        //uso do método locais_cidade da interface através do objeto da superclasse
        CD = IT.locais_cidade();

        //no método da interface locais_cidade é definido de que classe é o objeto

        IT.tipo_local(CD);
        /*O identificador de objeto pode receber endereços de instâncias de sua classe
        * ou de qualquer classe descendente.
        *O objeto ou identificador CD recebe o endereço de instância das classe Cidade,
        *Bairro e Distrito. Este identificador passa a identificar (conter o endereço,
        *referência) de uma instância de subclasse de Cidade.
        *A COMPATIBILIDADE DE ENDEREÇOS DE IDENTIFICADORES DE UMA CLASSE COM INSTÂNCIAS
        *DE SUBCLASSES É DENOMINADA DE SUBTIPAGEM (Boratti, 2007). */
    }
}
```

Resultados

Entrada [X]

? DIGITAR NOME DO LOCAL

Santos

OK Cancelar

Entrada [X]

? DIGITAR A QUANTIDADE DE HABITANTES

400

OK Cancelar

Entrada [X]

? cidade = c n/ bairro = b n/ distrito = d /n

d

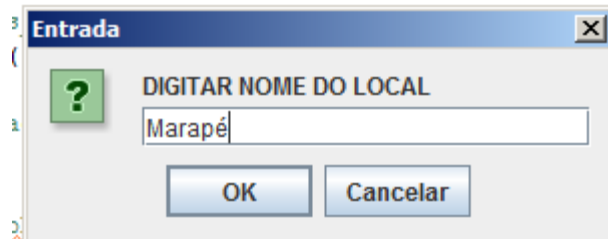
OK Cancelar

Problems Javadoc Declaration Console

<terminated> Usa_Tipagem [Java Application] C:\Program Files\Java\jre

```
local é cidade
nome da cidade = Santos_city
número de habitantes = 400000
```

Resultados

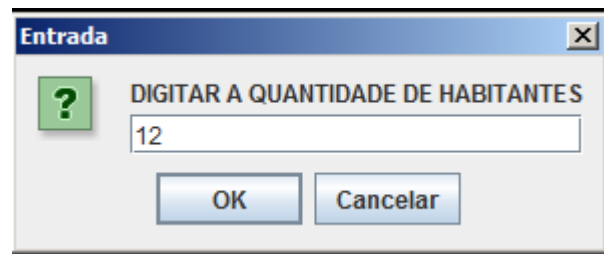


Entrada

DIGITAR NOME DO LOCAL

Marapé

OK Cancelar

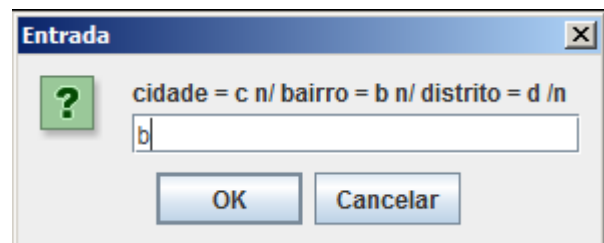


Entrada

DIGITAR A QUANTIDADE DE HABITANTES

12

OK Cancelar

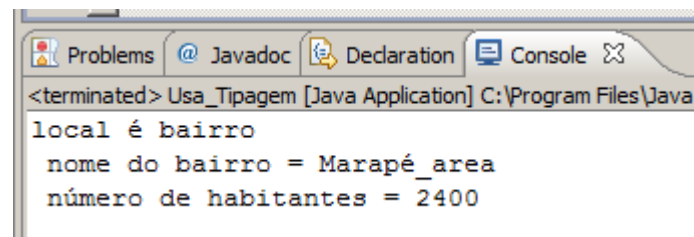


Entrada

cidade = c n/ bairro = b n/ distrito = d /n

b

OK Cancelar



Problems Javadoc Declaration Console

```
<terminated> Usa_Tipagem [Java Application] C:\Program Files\Java  
local é bairro  
nome do bairro = Marapé_area  
número de habitantes = 2400
```

Resultados

Entrada [X]

? DIGITAR NOME DO LOCAL

Vicente de Carvalho

OK Cancelar

Entrada [X]

? DIGITAR A QUANTIDADE DE HABITANTES

100

OK Cancelar

Entrada [X]

? cidade = c n/ bairro = b n/ distrito = d /n

d

OK Cancelar

Problems Javadoc Declaration Console

```
<terminated> Usa_Tipagem [Java Application] C:\Program Files\Java\jre1.8.0_20\  
local é distrito  
nome do distrito = Vicente de Carvalho_district  
número de habitantes = 30000
```


Exercício 19: Construir o diagrama de classes do pacote Tipagem.

Exercício 20: Compilar e executar as classes do pacote Tipagem. Verificar os resultados.

Referências Bibliográficas

- Programação Orientada a Objetos em Java
Isaias Camilo Boratti – Editora Visual Books ISBN 85-7502-199-0.
Florianópolis, 2007