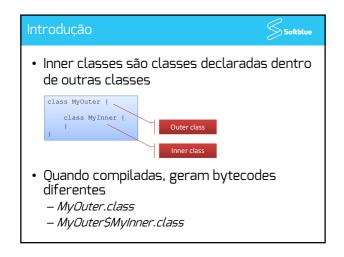


Tópicos Abordados	Softblue
 Introdução Tipos de inner classes Regular Method-Local Anonymous Static 	



Tipos de inner classes

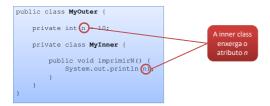


- As inner classes podem ser divididas em 4 tipos
 - Regular inner class
 - Method-local inner class
 - Anonymous inner class
 - Static inner class

Regular inner class



- Declarada como membro de uma classe
- Possui acesso aos elementos da classe dentro da qual está inserida



Instanciando uma inner class



- Uma instância de uma inner class não pode existir sem estar associada a uma instância de uma outer class
- Normalmente é a outer class que instancia a inner class

```
public class MyOuter {
    private class MyInner {
        //...
}

public void criarInner() {
        MyInner i = new MyInner();
}

A instanciação é feita como se fosse com qualquer outra classed public void criarInner() {
        MyInner i = new MyInner();
}
```

Obtendo a referência da outer class



- O operador *this* referencia o próprio objeto
- Dentro de uma inner class, *this* referencia a instância da inner class
- E para referenciar a outer class?

```
public class MyOuter {
    private class MyInner {
        public void m() {
            System.out.println("Ref inner: " + this);
            System.out.println("Ref outer: " + MyOuter.this);
        }
    }
}
```

Method-local inner class

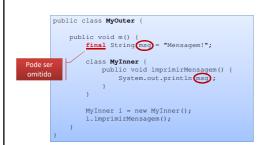


- Declarada dentro de um método
- Apenas o método enxerga a classe

Mothed local inner class



 A inner class pode acessar variáveis locais do método, desde que estas sejam final ou não tenham seu valor alterado



Anonymous inner class



- Não possui nome
- Classes anônimas são sempre subclasses de uma classe ou implementação de uma interface
- Sobrescrevem ou implementam métodos da superclasse ou interface

public class Porta { public void abrir() { System.out.println("abrir"); } } public class Casa { private Porta p = new Porta() { public void abrir() { System.out.println("porta anônima"); }; public void m() { p.abrir(); } } O método invocado é o método sobrescrito sobrescrito

• Usando um java.util.Comparator - Interface que possui o método compare() Comparator<String> comparator = new Comparator<String>() { public int compare(String ol, String ol) { return ol.compareTo(ol) * -1; }; TreeSet<String> s = new TreeSet<String> (comparator); A classe anônima implementa a interface Comparator

1



• O exemplo pode ser simplificado ainda mais

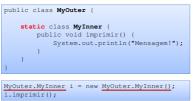
```
TreeSet<String> s = new TreeSet<String>(new Comparator<String>() {
   public int compare(String ol, String o2) {
      return ol.compareTo(o2) * -1;
   }
}
```

• Neste exemplo a classe anônima é passada diretamente como parâmetro para o construtor do *TreeSet*



- Não é realmente uma inner class porque não tem um relacionamento especial com a outer class
- Ela é apenas uma classe declarada dentro de outra classe
- Basta declarar a classe como static





• Uma static inner class famosa no Java é a Map. Entry, cujos objetos são retornados quando o método *entrySet()* é invocado

