# Java Avançado

Inner Classes

SoftBlue
www.softblue.com.br

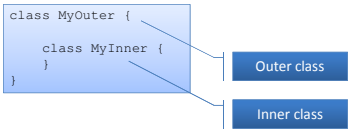


## Tópicos Abordados

- Introdução
- Tipos de inner classes
  - "Regular"
  - Method-Local
  - Anonymous
  - Static

## Introdução

- **Inner Classes** são classes declaradas dentro de outras classes




- Quando compiladas, geram bytecodes diferentes
  - MyOuter.class
  - MyOuter$MyInner.class

## 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

```
public class MyOuter {

    private int n = 10;

    private class MyInner {

        public void imprimirN() {
            System.out.println(n);
        }
    }
}
```

A inner class enxerga o atributo *n*

## 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 MyIner();
    }
}
```

A instanciação é feita como se fosse com qualquer outra classe

## 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

```
public class MyOuter {

    public void m() {
        class MyInner {
            public void imprimirMensagem() {
                System.out.println("Mensagem!");
            }
        }

        MyInner i = new MyInner();
        i.imprimirMensagem();
    }
}
```

Inner class

## Method-Local Inner Class

- A inner class pode acessar variáveis locais do método, desde que estas sejam **final**

```
public class MyOuter {

    public void m() {
        final String msg = "Mensagem!";

        class MyInner {
            public void imprimirMensagem() {
                System.out.println(msg);
            }
        }

        MyInner i = new MyInner();
        i.imprimirMensagem();
    }
}
```

## 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

## Exemplo de Anonymous Inner Class

```
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();
    }
}
```

Classe anônima sendo uma subclasse de *Porta*

O método invocado é o método sobrescrito

## Exemplo de Anonymous Inner Class

- Usando um java.util.Comparator
  - Interface que possui o método *compare()*

```
Comparator<String> comparator = new Comparator<String>() {
    public int compare(String o1, String o2) {
        return o1.compareTo(o2) * -1;
    }
};

TreeSet<String> s = new TreeSet<String>(comparator);
```

A classe anônima implementa a interface *Comparator*

A referência ao objeto é usada no construtor do *TreeSet*

## Exemplo de Anonymous Inner Class

- O exemplo pode ser simplificado ainda mais

```
TreeSet<String> s = new TreeSet<String>(new Comparator<String>() {
    public int compare(String o1, String o2) {
        return o1.compareTo(o2) * -1;
    }
});
```

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

## Static Inner Class

- 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**

## Exemplo de Static Inner Class

```
public class MyOuter {

    static class MyInner {
        public void imprimir() {
            System.out.println("Mensagem!");
        }
    }
}
```

```
MyOuter.MyInner i = new MyOuter.MyInner();
i.imprimir();
```

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

## Colocando em Prática...

Agora que você já aprendeu a teoria, acesse as vídeo-aulas práticas e pratique os assuntos abordados neste módulo!

Clique aqui para acessar as vídeo-aulas práticas