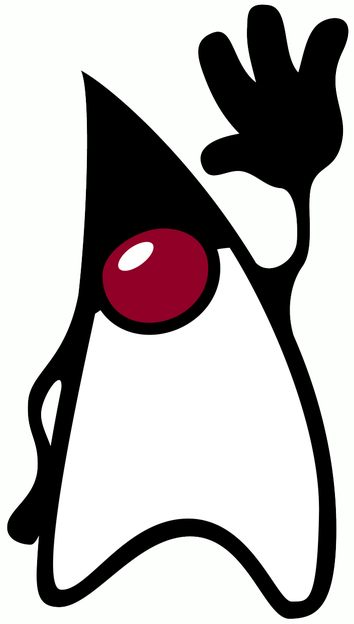
UNIVERSIDADE
FEDERAL DO CEARÁ
CAMPUS QUIXADÁ

Interfaces

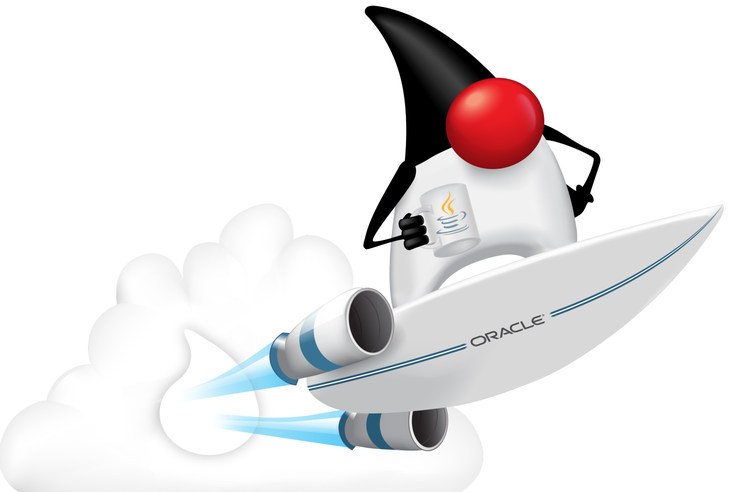
QXD0007 - Programação Orientada a Objetos



Prof. Bruno Góis Mateus (brunomateus@ufc.br)

Conteúdo

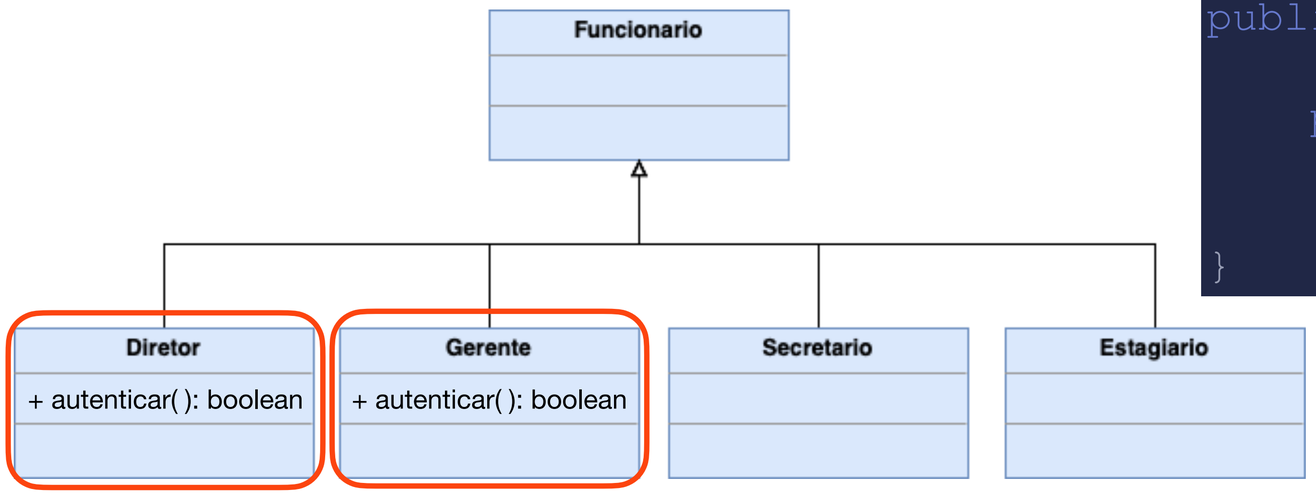
- Interfaces
- Mão na massa


ORACLE

Interfaces

Interfaces

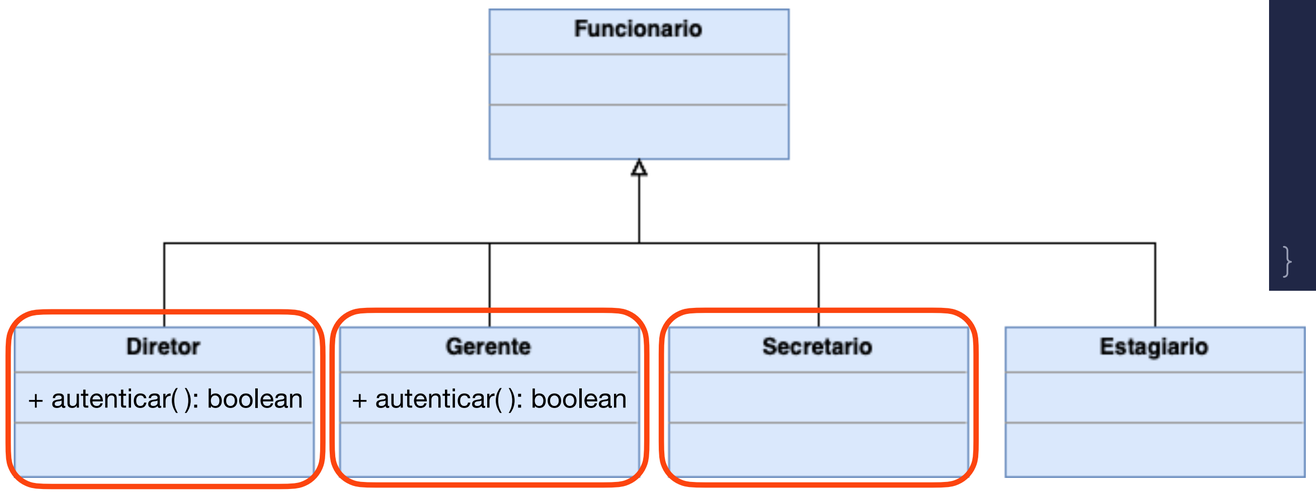
- Imagine um sistema de controle do banco que pode ser acessado por Gerentes e Diretores:




```
public class SistemaInterno {

    public void login(Funcionario funcionario){
        funcionario.autenticar()
    }
}
```

?

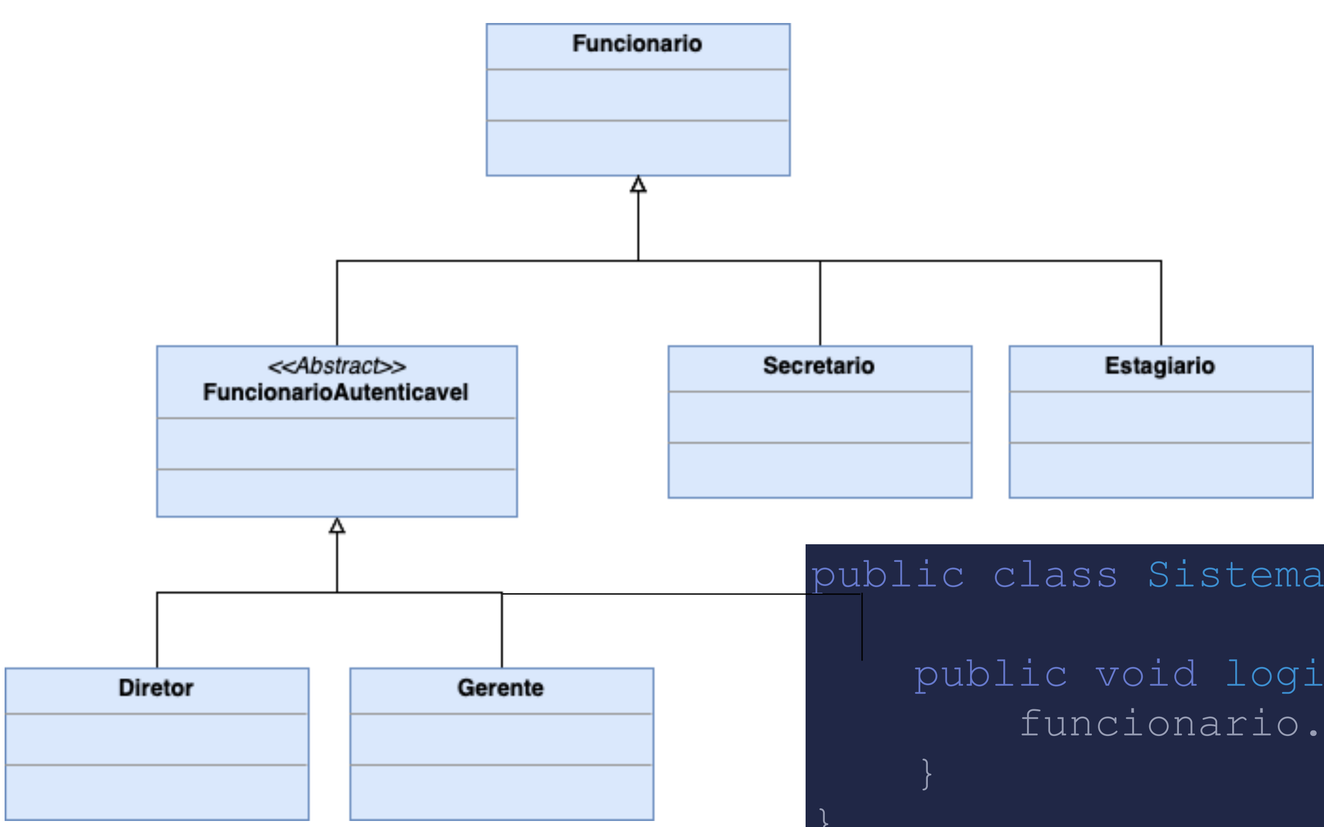
Interfaces

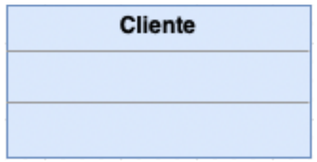



```
public class SistemaInterno {

    public void login(Diretor funcionario) {
        funcionario.autenticar();
    }

    public void login(Gerente funcionario) {
        funcionario.autentica();
    }
}
```

?

Interfaces





?

```
public class SistemaInterno {

    public void login(FuncionarioAutenticavel funcionario) {
        funcionario.autenticar();
    }
}
```

Interfaces

Classes vs Interfaces

Classes

- O que uma classe faz
  - As assinaturas dos métodos
- Como as classes realizam essas tarefas
  - O corpos dos métodos e seus atributos

Interfaces

- Contrato entre a classe e o mundo externo
  - Determinam o que, porém não indica o como
- Ao implementar uma interface, a classe se compromete a fornecer o comportamento publicado pela interface

Interfaces

Interface

- Uma interface descreve um aspecto de comportamento que muitas classes diferentes podem requerer
- O nome de uma interface é freqüentemente um adjetivo como Sortable, Comparable, Runnable

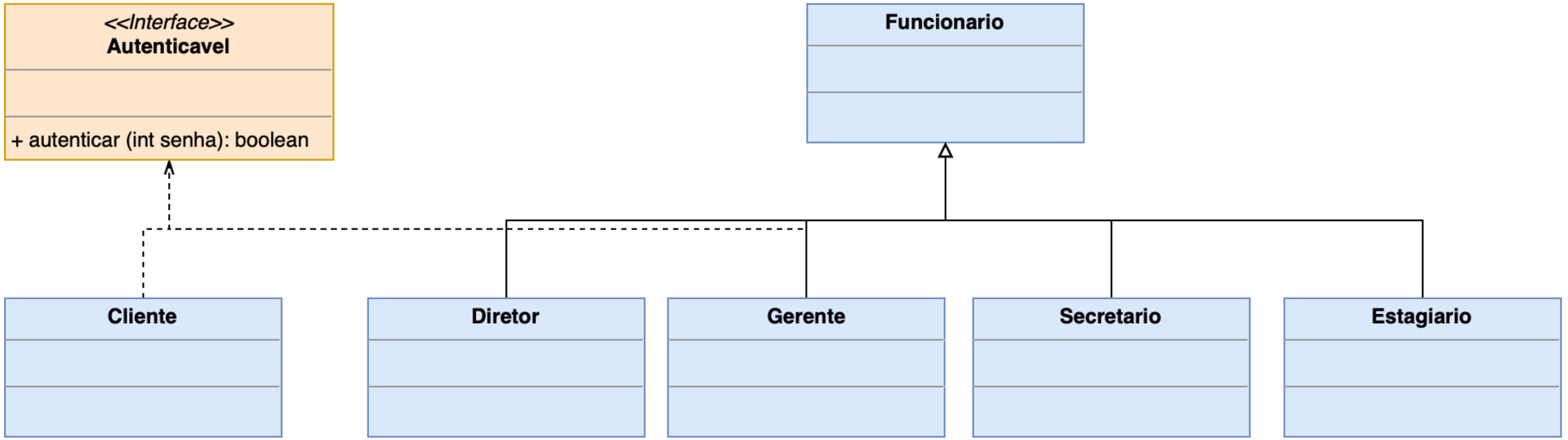
Interfaces

Interface Autenticável

- Contrato:
  - Quem quiser ser Autenticavel precisa saber fazer:
    - Autenticar dada uma senha, devolvendo um booleano

```
interface Autenticavel {
    boolean autentica(int senha);
}
```

Interfaces




```
public class SistemaInterno {

    public void login(Autenticavel funcionario) {
        …
        funcionario.autenticar(senha);
    }
}
```

Interfaces

Interfaces

- Pode definir um série métodos, mas eles “nunca” podem conter a implementação
  - São abstratos e públicos por padrão
  - Os métodos são implementados pelos subtipos
  - Não têm construtores
- Expõem somente o que o objeto deve fazer e não determina como o objeto tem que fazer
  - A classe que implementa a interface define o como

Interfaces

Sem herança

Múltiplos supertipos

- Uma classe pode implementar mais de uma interface
- A classe que implementa uma interface deve definir (não necessariamente implementar!) os métodos da interface

```
public abstract class Shape implements Drawable, Nomeavel, AreaCalculavel{

  public Shape(String color, Point point) {

  }
}
}
  }
```

Interfaces

Reusabilidade

- Evita duplicação de código através da definição de um tipo genérico, tendo como subtipos várias classes não relacionadas
- Pode agrupar objetos de várias classes definidas independentemente
  - Sem compartilhar código via herança (menor acoplamento)
  - Implementações totalmente diferentes

Interfaces

```
public class HelloRunnable implements Runnable {

  public void run() {
    System.out.println("Hello from a thread!");
  }

  public static void main(String args[]) {
    (new Thread(new HelloRunnable())).start();
  }

}
```

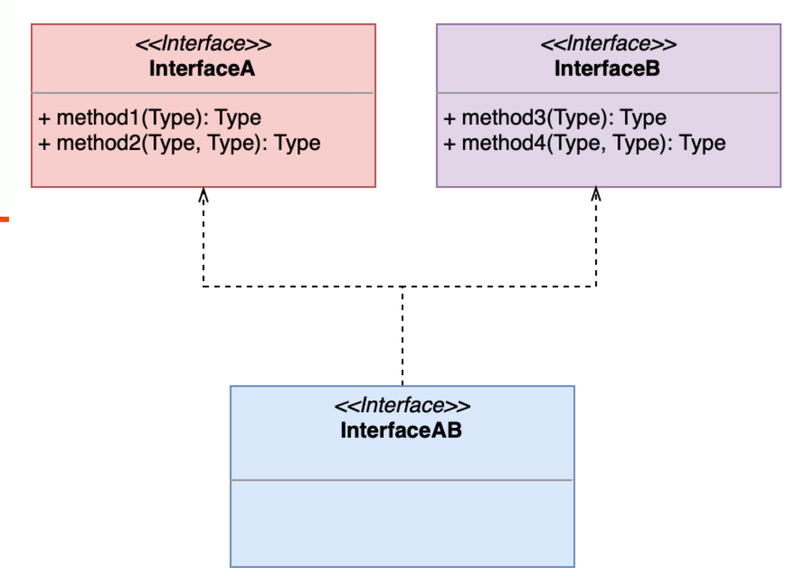
```
public class HelloThread extends Thread {

  public void run() {
    System.out.println("Hello from a thread!");
  }

  public static void main(String args[]) {
      (new HelloThread()).start();
  }

}
```

run()
If this thread was constructed using a separate `Runnable` run object, then that `Runnable` object's `run` method is called; otherwise, this method does nothing and returns.

Interfaces

Herança Múltiplas

- Uma interface pode herdar de várias interfaces



Interfaces

Classes abstratas vs Interfaces

	Interfaces	Classes Abstrastas
Modificadores de acesso	public	Todos
Podem ter atributos?	Não	Sim
Podem ter atributos de classe?	Sim (public e final)	Sim
Podem conter métodos abstratos?	Sim	Sim
Podem conter métodos concretos	Não	Sim
Podem conter métodos default?	Sim	Não
Podem conter métodos estáticos?	Sim	Sim
Podem ser instanciadas?	Não	Não
Podem ter construtores?	Não	Sim
Herança	Pode herdar de várias interfaces	Pode herdar de uma única classe
Implementação (*implements*)	Várias interfaces	Não

Interfaces

Classes vs Interfaces

Classes

- Agrupam objetos com implementações compartilhadas
- Definem novas classes através de herança de interface e implementação
- Uma hierarquia de classes compartilha implementação

Interfaces

- Agrupa objetos com implementações diferentes
- Define novas interfaces através de herança (múltipla) de interfaces
- Uma hierarquia de interfaces compartilha obrigrações
- Permitem que objetos tenham vários tipos

Interfaces

Mais informações

- Métodos default
  - https://www.amitph.com/introduction-to-default-methods-in-java-8/
  - https://docs.oracle.com/javase/tutorial/java/landl/defaultmethods.html
- Discussão sobre classes abstratas vs interfaces
  - https://stackoverflow.com/questions/8531292/why-to-use-interfaces-multiple-inheritance-vs-interfaces-benefits-of-interface
  - https://betterprogramming.pub/choosing-between-interface-and-abstract-class-7a078551b914

Mãos na massa

Por hoje é só