UNIP
UNIVERSIDADE PAULISTA

UNICLUSTER

Interface

Carlos Arruda Baltazar
UNIP – Cidade Universitária

Interface

Podemos definir como interface o contrato entre a classe e o mundo exterior. Quando uma classe implementa uma interface, se compromete a fornecer o comportamento publicado por esta interface.

Interface

As classes ajudam a definir um objeto e seu comportamento e as interfaces que auxiliam na definição dessas classes. As interfaces são formadas pela declaração de um ou mais métodos, os quais obrigatoriamente não possuem corpo.

Interface

As operações específicas para cada um desses métodos são realizadas pela classe que implementa. De um modo geral, podemos dizer que as interfaces definem certas funcionalidades, as quais dependem das classes que implementam as interfaces para que os métodos existam.

Interface

Em uma mesma classe, podem ser implementadas uma ou mais interfaces, sendo que elas devem estar separadas por vírgulas. Se uma classe implementa uma determinada interface, todos os métodos declarados nessa interface implementada devem ser definidos na respectiva classe. Caso isso não ocorra, será gerado um erro de compilação. Os membros declarados em uma interface são implicitamente públicos. Neste ponto, devemos considerar que não é permitido declarar variáveis em uma interface.

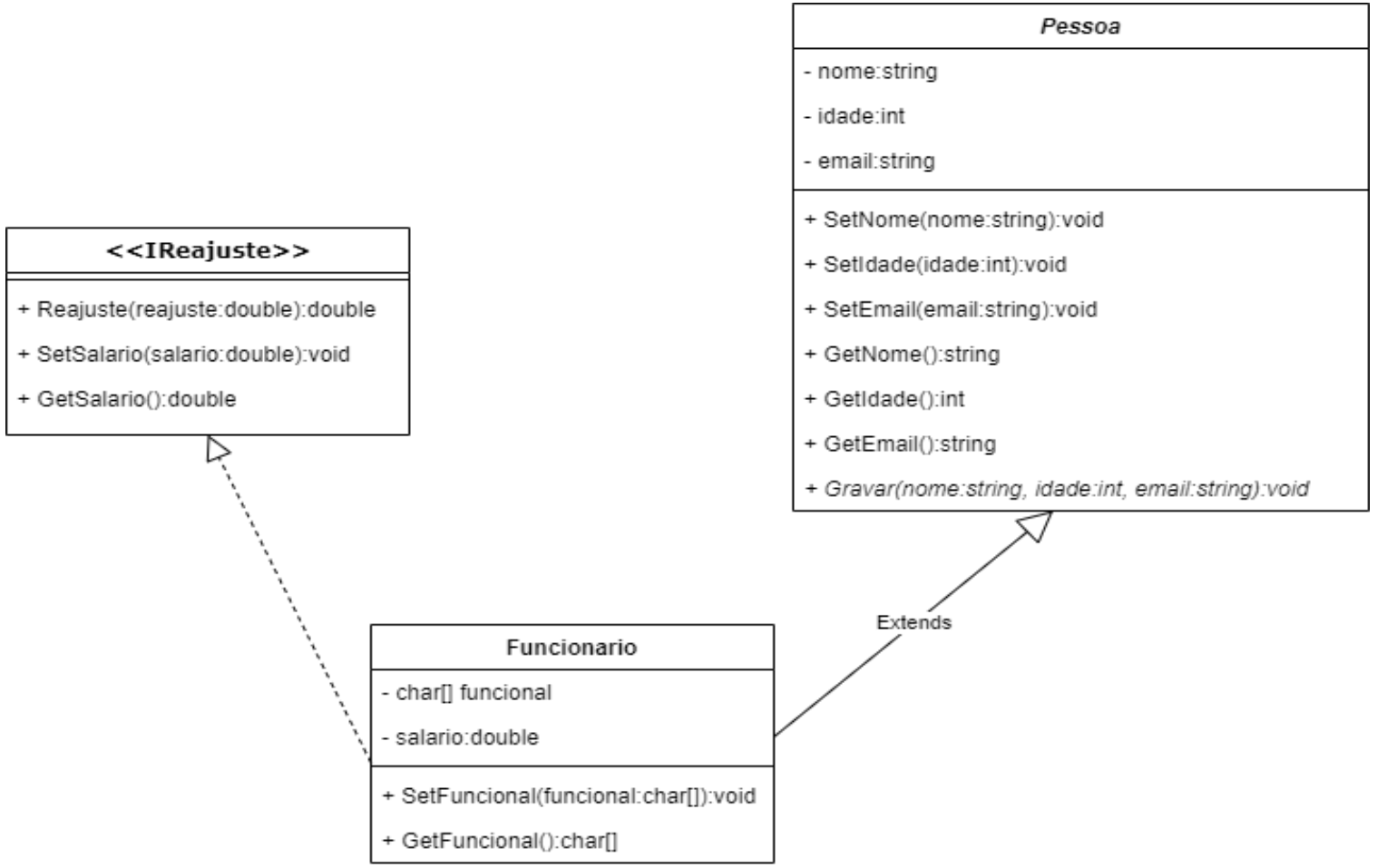
Interface

Como exemplo da utilização de interfaces, podemos citar quando trabalhamos com componentes gráficos na tela do computador, em que utilizamos botões, janelas e outros itens. Neste caso, temos várias interfaces com diversos métodos para o tratamento de eventos que podem ocorrer nesses componentes.

Interface

Dentro esses eventos, podemos ter os seguintes: fechar a janela, abrir a janela, clicar sobre um botão, etc. Os métodos que capturam esses eventos estão definidos em interfaces específicas, mas a ação a ser executada quando um destes eventos ocorrer é de responsabilidade da classe que implementou a interface. É por isso que os métodos declarados na interface não possuem corpo.

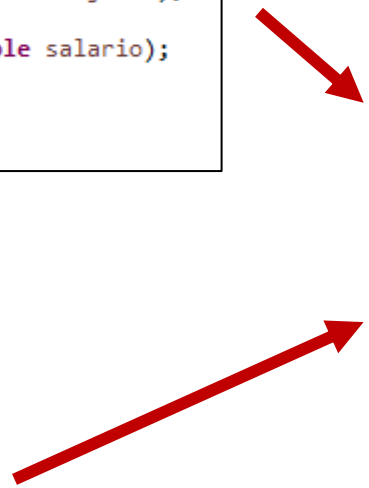
Interface


Pessoa
- nome:string
- idade:int
- email:string
+ SetNome(nome:string):void
+ SetIdade(idade:int):void
+ SetEmail(email:string):void
+ GetNome():string
+ GetIdade():int
+ GetEmail():string
+ Gravar(nome:string, idade:int, email:string):void
<<IReajuste>>
+ Reajuste(reajuste:double):double
+ SetSalario(salario:double):void
+ GetSalario():double
Funcionario
- char[] funcional
- salario:double
+ SetFuncional(funcional:char[]):void
+ GetFuncional():char[]
Extends

Interface


```java
package Pck_teste2;

public abstract class Pessoa
{
    private String nome;
    private int idade;
    private String email;

    public String getNome() {
        return nome;
    }
    public void setNome(String nome) {
        this.nome = nome;
    }
    public int getIdade() {
        return idade;
    }
    public void setIdade(int idade) {
        this.idade = idade;
    }
    public String getEmail() {
        return email;
    }
    public void setEmail(String email) {
        this.email = email;
    }

    public abstract void Gravar(String nome, int idade, String email);
}
```

```java
package Pck_teste2;

public interface IReajuste
{
    public double Reajuste(double reajuste);

    public void SetSalario(double salario);

    public double GetSalario();
}
```



```java
package Pck_teste2;

public class Funcionario extends Pessoa implements IReajuste
{
    private char[] funcional;
    private double salario;

    private char[] getFuncional() {
        return funcional;
    }

    private void setFuncional(char[] funcional) {
        this.funcional = funcional;
    }

    public void Gravar(String nome, int idade, String email)
    {
        super.setNome(nome);
        super.setIdade(idade);
        super.setEmail(email);
    }

    @Override
    public double Reajuste(double reajuste) {
        // TODO Auto-generated method stub
        return this.GetSalario() + reajuste;
    }

    @Override
    public void SetSalario(double salario) {
        // TODO Auto-generated method stub
        this.salario = salario;
    }

    @Override
    public double GetSalario() {
        // TODO Auto-generated method stub
        return this.salario;
    }
}
```

Exemplo

Suponha que temos um sistema de banco onde existem diferentes tipos de contas (conta corrente ou poupança; conta pessoa física ou jurídica). Apesar dos diferentes tipos, todas as contas devem executar operações como: depositar; sacar; transferir e gerar o saldo.

Exemplo

Neste exemplo, temos um conjunto de operações que são comuns a mais de um ponto no sistema. Entretanto, cada conta possui informações, validações, regras de negócio e campos próprios. Sendo assim, criar uma superclasse cadastro com a implementação dos métodos de forma que a implementação atender a todas as contas seria muito difícil pois cada conta possui políticas diferentes.

Exemplo

Por isso este problema não pode ser resolvido com o uso de herança. Neste caso devemos apenas declarar o que deve ser feito em uma interface que declare todas as operações comuns aos cadastros do sistema, sendo que cada cadastro deveria ter uma classe que implemente a interface de acordo com as informações, regras e validações específicas de cada cadastro.

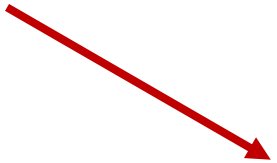
Exemplo

A solução para este problema seria a criação de uma interface para definir as operações que toda conta obrigatoriamente pode ter, desta forma, uma classe que implemente a interface se compromete a modelar seu comportamento em torno dessas operações obrigatórias.

Exemplo


```java
package Pck_Interface;

public class Conta
{
    private double saldo;
    private int nConta;
    private String nomeTitular;

    public double getSaldo() {
        return saldo;
    }

    public void setSaldo(double saldo) {
        this.saldo = saldo;
    }

    public int getnConta() {
        return nConta;
    }

    public void setnConta(int nConta) {
        this.nConta = nConta;
    }

    public String getNomeTitular() {
        return nomeTitular;
    }

    public void setNomeTitular(String nomeTitular) {
        this.nomeTitular = nomeTitular;
    }
}
```

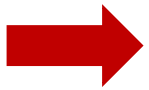
```java
package Pck_Interface;

public class ContaFisica extends Conta
{
    private char[] cpf;

    public char[] getCpf() {
        return cpf;
    }

    public void setCpf(char[] cpf) {
        this.cpf = cpf;
    }
}
```



```java
package Pck_Interface;

public class ContaJuridica extends Conta
{
    private char[] cnpj;

    public char[] getCnpj() {
        return cnpj;
    }

    public void setCnpj(char[] cnpj) {
        this.cnpj = cnpj;
    }
}
```

Exemplo

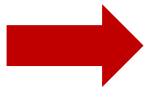

```java
package Pck_Interface;

public interface IConta
{
    public void Depositar(double valor);

    public double Sacar(double valor);

    public double Transferir(double valor);

    public void GetSaldo();
}
```



```java
package Pck_Interface;

public class ContaCorrenteJuridica extends ContaJuridica implements IConta
{
    private static double taxaOperacional = 0.45;
    private int nExtratos = 0;

    @Override
    public void Depositar(double valor)
    {
        // TODO Auto-generated method stub
        super.setSaldo(super.getSaldo() + valor);
    }

    @Override
    public double Sacar(double valor)
    {
        // TODO Auto-generated method stub
        super.setSaldo(super.getSaldo() - valor - this.taxaOperacional);
        return valor;
    }

    @Override
    public double Transferir(double valor)
    {
        // TODO Auto-generated method stub
        super.setSaldo(super.getSaldo() - valor - this.taxaOperacional);
        return valor;
    }

    @Override
    public void GetSaldo()
    {
        // TODO Auto-generated method stub
        if(this.nExtratos > 16)
        {
            System.out.println("Seu saldo é: " + this.getSaldo());
            super.setSaldo(super.getSaldo() - this.taxaOperacional);
            this.nExtratos ++;
        }
        else
        {
            System.out.println("Seu saldo é: " + this.getSaldo());
            this.nExtratos ++;
        }
    }

}
```

Exemplo

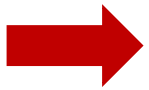
```java
package Pck_Interface;

public interface IConta
{
    public void Depositar(double valor);

    public double Sacar(double valor);

    public double Transferir(double valor);

    public void GetSaldo();
}
```



```java
package Pck_Interface;

public class ContaPoupancaFisica extends ContaFisica implements IConta
{
    private static double taxaOperacional = 0.11;

    @Override
    public void Depositar(double valor)
    {
        // TODO Auto-generated method stub
        super.setSaldo(super.getSaldo() + valor);
    }

    @Override
    public double Sacar(double valor)
    {
        // TODO Auto-generated method stub
        super.setSaldo(super.getSaldo() - valor);
        return valor;
    }

    @Override
    public double Transferir(double valor)
    {
        // TODO Auto-generated method stub
        super.setSaldo(super.getSaldo() - valor - this.taxaOperacional);
        return valor;
    }

    @Override
    public void GetSaldo()
    {
        // TODO Auto-generated method stub
        System.out.println("Seu saldo é: " + this.getSaldo());
    }

}
```

Exemplo


```java
package Pck_Interface;

public interface IConta
{
    public void Depositar(double valor);

    public double Sacar(double valor);

    public double Transferir(double valor);

    public void GetSaldo();
}
```

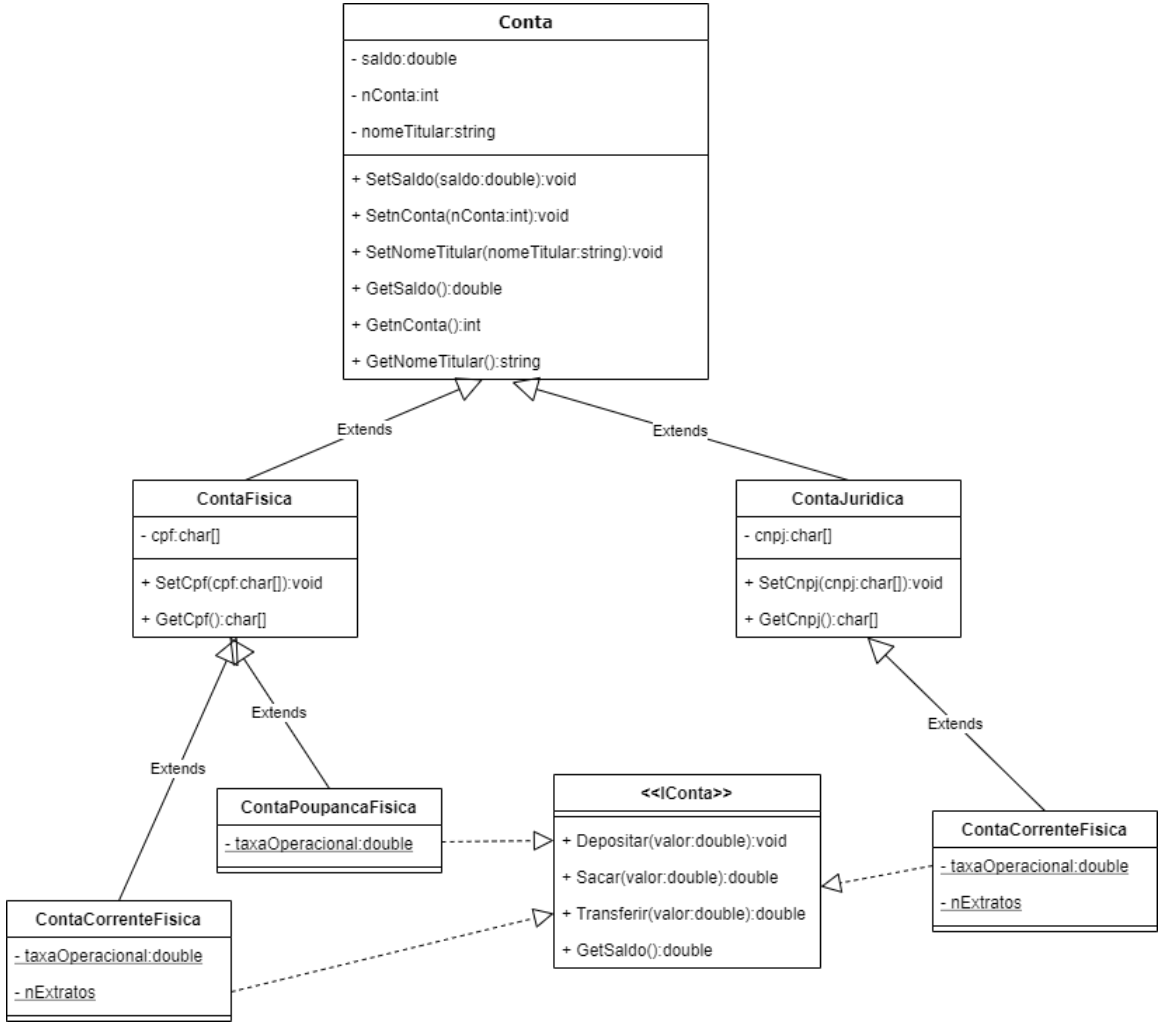


```java
package Pck_Interface;

public class ContaCorrenteFisica extends ContaFisica implements IConta
{
    private static double taxaOperacional = 0.22;
    private int nExtratos = 0;

    @Override
    public void Depositar(double valor)
    {
        // TODO Auto-generated method stub
        super.setSaldo(super.getSaldo() + valor);
    }

    @Override
    public double Sacar(double valor)
    {
        // TODO Auto-generated method stub
        super.setSaldo(super.getSaldo() - valor);
        return valor;
    }

    @Override
    public double Transferir(double valor)
    {
        // TODO Auto-generated method stub
        super.setSaldo(super.getSaldo() - valor - this.taxaOperacional);
        return valor;
    }

    @Override
    public void GetSaldo() {
        // TODO Auto-generated method stub
        if(this.nExtratos > 4)
        {
            System.out.println("Seu saldo é: " + this.getSaldo());
            super.setSaldo(super.getSaldo() - this.taxaOperacional);
            this.nExtratos ++;
        }
        else
        {
            System.out.println("Seu saldo é: " + this.getSaldo());
            this.nExtratos ++;
        }
    }
}
```

Exercício

A partir do código apresentado no exemplo, construa o respectivo diagrama de classes.

Exercício


Conta
- saldo:double
- nConta:int
- nomeTitular:string
+ SetSaldo(saldo:double):void
+ SetnConta(nConta:int):void
+ SetNomeTitular(nomeTitular:string):void
+ GetSaldo():double
+ GetnConta():int
+ GetNomeTitular():string
Extends
Extends
ContaFisica
- cpf:char[]
+ SetCpf(cpf:char[]):void
+ GetCpf():char[]
ContaJuridica
- cnpj:char[]
+ SetCnpj(cnpj:char[]):void
+ GetCnpj():char[]
Extends
Extends
Extends
ContaPoupancaFisica
- taxaOperacional:double
<<IConta>>
+ Depositar(valor:double):void
+ Sacar(valor:double):double
+ Transferir(valor:double):double
+ GetSaldo():double
ContaCorrenteFisica
- taxaOperacional:double
- nExtratos
ContaCorrenteFisica
- taxaOperacional:double
- nExtratos

OBRIGADO