SENAI CIMATEC SISTEMA FIEB
Federação das Indústrias do Estado da Bahia

2016.2

Introdução a Programação Orientada a Objetos Utilizando PHP PARTE 03

Docente: Gil Jader – gil.jader@gmail.com

1. Relacionamento entre Classes

- Classes possuem relacionamentos entre elas (para comunicação)
  - Compartilham informações
  - Colaboram umas com as outras

- Principais tipos de relacionamentos
  - Herança
  - Associação
  - Agregação / Composição

1. Relacionamento entre Classes

- **Associação**
  - Associação é a relação mais comum entre dois objetos, de modo que um possui uma referência à posição de memória onde o outro se encontra, podendo visualizar seus atributos ou acionar uma de suas funcionalidades (métodos).

Tipos de Associação

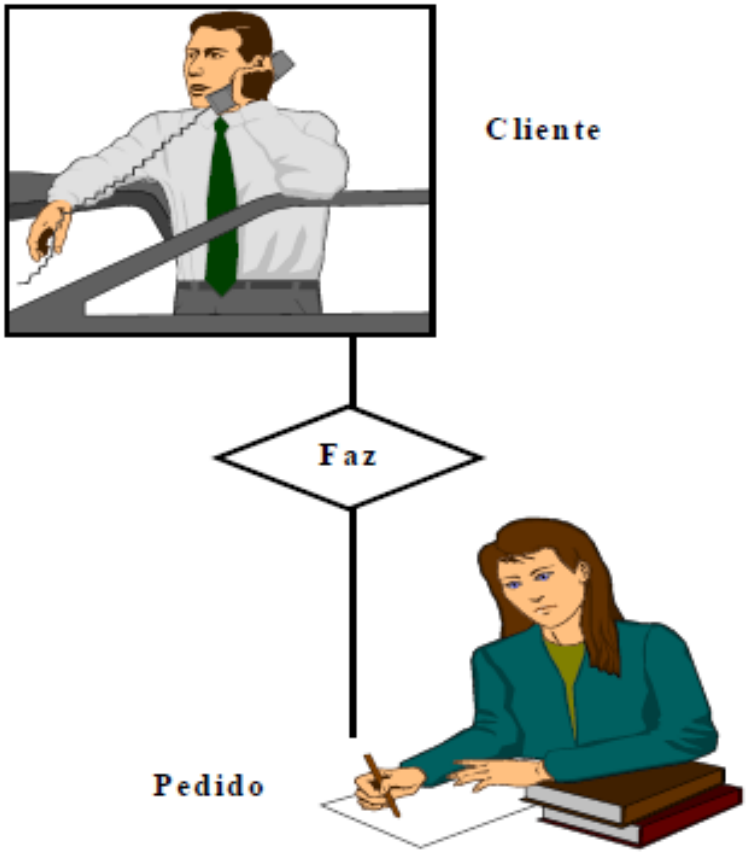
- **Unárias** - quando a associação ocorre entre objetos de uma mesma classe.
- **Binárias** - quando a associação ocorre entre dois objetos de classes distintas.
- **Múltiplas** - quando a associação ocorre entre mais de dois objetos de classes distintas.

1. Relacionamento entre Classes

- **Associação**
  - **Determina que as instâncias de uma classe estão de alguma forma ligadas às instâncias da outra classe**
  - Usada para agrupar objetos que ocorrem sob algumas circunstâncias similares ou um ponto específico no tempo
  - Esse relacionamento existe porque um objeto necessita de outros para cumprir certas responsabilidades

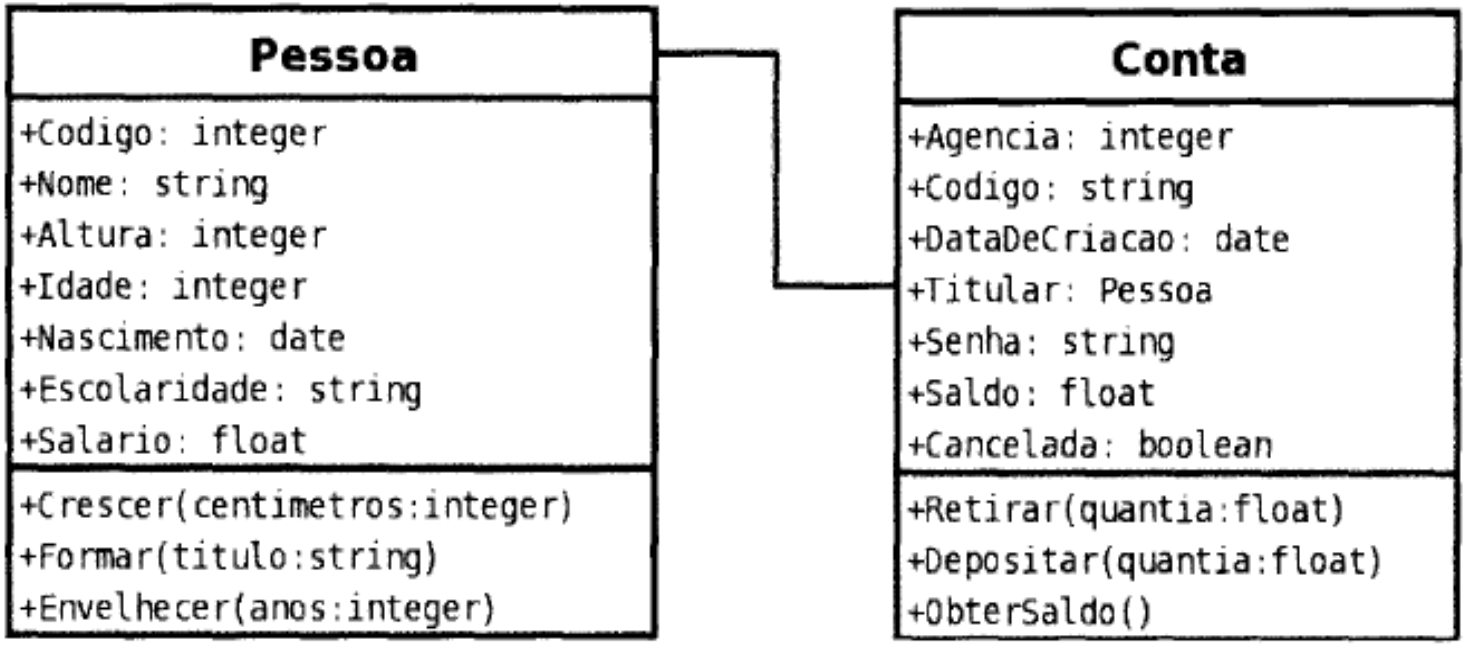
1. Relacionamento entre Classes

- **Associação**



1. Relacionamento entre Classes

- **Representação de Associação**

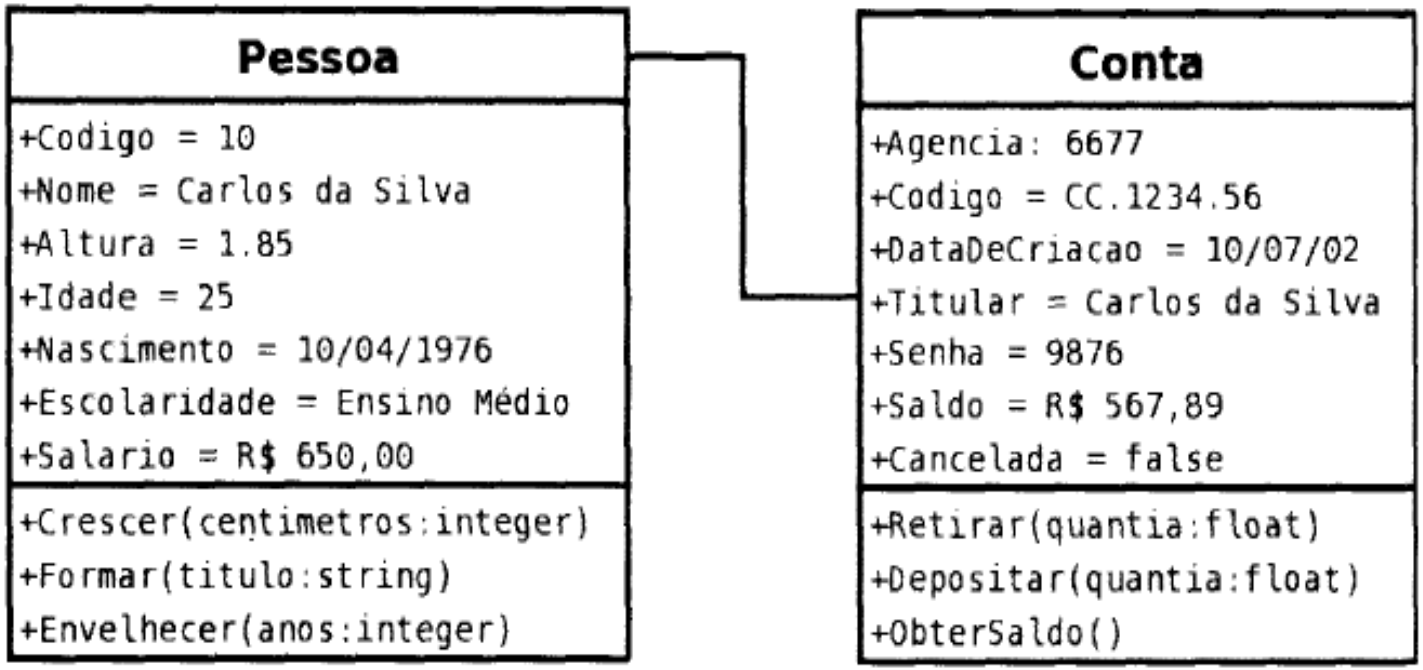


```php
Class Pessoa
{
    var $Codigo;
    var $Nome;
    var $Altura;
    var $Idade;
    var $Nascimento;
    var $Escolaridade;
    var $Salario;

    /* método Crescer
    * aumenta a altura em $centimetros
    */
    function Crescer($centimetros)
    {
        if ($centimetros > 0){
            $this->Altura += $centimetros;
        }
    }

    /* método Formar
    * altera a Escolaridade para $titulacao
    */
    function Formar($titulacao)
    {
            $this->Escolaridade = $titulacao;
    }

    /* método Envelhecer
    * aumenta a idade em $anos
    */
    function Envelhecer($anos)
```

```php
Class Conta
{
    var $Agencia;
    var $Codigo;
    var $DataDeCriacao;
    var $Titular;
    var $Senha;
    var $Saldo;
    var $Cancelada;

    /* método Retirar
    * diminui o saldo em $quantia
    */
    function Retirar($quantia)
    {
        if ($quantia > 0){
            $this->Saldo -= $quantia;
        }
    }

    /* método Depositar
    * acrescenta a $quantia em saldo
    */
    function Depositar($quantia)
    {
        if ($quantia > 0){
            $this->Saldo += $quantia;
        }
    }

    /* método ObterSaldo
    * retorna o Saldo Atual
    */
    function ObterSaldo()
```

1. Relacionamento entre Classes

- **Representação de Associação**



```php
# carrega as classes
include_once 'classes/Pessoa.class.php';
include_once 'classes/Conta.class.php';

# criação do objeto $fulano
$fulano = new Pessoa;
$fulano->Codigo = 10;
$fulano->Nome = "Fulano de Tal";
$fulano->Altura = 1.72;
$fulano->Idade = 34;
$fulano->Nascimento = '13/07/1981';
$fulano->Escolaridade = 'Doutorado';

echo "Manipulando o objeto: <b>$fulano->Codigo</b> <br><br>";

echo "$fulano->Nome tem Escolaridade em: <b> $fulano->Escolaridade</b> <br>";
$fulano->Formar('Doutorado em Ciências da Computação');
echo "$fulano->Nome tem Escolaridade em: <b> $fulano->Escolaridade </b> <br><br>";

echo "$fulano->Nome tem <b> $fulano->Idade </b> anos de idade <br>";
$fulano->Envelhecer(1);
echo "$fulano->Nome tem <b> $fulano->Idade </b> anos de idade <br><br><br>";
```

Continuação

```php
# criação do objeto $conta_fulano
$conta_fulano = new Conta;
$conta_fulano->Agencia = 6677;
$conta_fulano->Codigo = "CC.1234.56";
$conta_fulano->DataDeCriacao = "10/07/2002";
$conta_fulano->Titular = $fulano;
$conta_fulano->Senha = 9876;
$conta_fulano->Saldo = 567.89;
$conta_fulano->Cancelada = false;

echo "Manipulando a <b>conta</b> de: {$conta_fulano->Titular->Nome} <br><br>";
echo "O saldo atual é <b> R\$ {$conta_fulano->ObterSaldo()} </b> <br>";

$conta_fulano->Depositar(20);
echo "O saldo atual é <b> R\$ {$conta_fulano->ObterSaldo()} </b> <br>";

$conta_fulano->Retirar(87.89);
echo "O saldo atual é <b> R\$ {$conta_fulano->ObterSaldo()} </b> <br>";
```

```php
<?php
// Marido
class Marido
{
    // Propriedade
    public $nome;
    public $esposa;

    // Configura a propriedade
    function __construct( $nome = null, $esposa = null ) {
        $this->nome   = $nome;
        $this->esposa = $esposa;
    }
}

// Esposa
class Esposa
{
    // Propriedade
    public $nome;

    // Configura a propriedade
    function __construct( $nome = null ) {
        $this->nome = $nome;
    }
}

// Faz as instâncias
$esposa = new Esposa('Maria');
$marido = new Marido('Joãozinho', $esposa );

// Joãozinho e Maria
echo $marido->nome . ' e ' . $marido->esposa->nome;
?>
```

Associação – Outro Exemplo

1. Relacionamento entre Classes

- **Associação - Resumo**

É uma das relações mais comuns entre dois objetos, acontece quando um objeto "utiliza" outro, porém, sem que eles dependam um do outro. Em outras palavras, é como se eu tivesse duas classes distintas, e fizesse uso de uma delas dentro da outra, ou como parâmetro de outra.

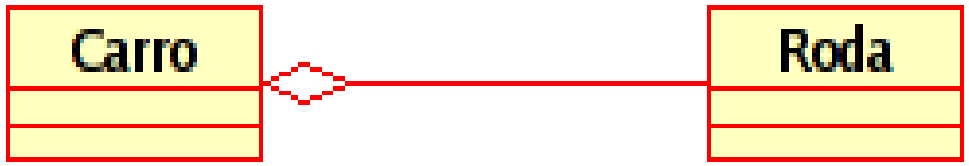
1. Relacionamento entre Classes

- **Agregação**

  - Tipo especial de associação
  - Associação Todo - Parte
    - Objeto - todo
    - Objeto - parte
  - Uma agregação diz como é que a classe que tem o papel do 'todo' é composta (ou tem) as outras classes, que tem o papel das partes.
  - Para as agregações, a classe que atua como o 'todo' tem sempre uma multiplicidade de um.

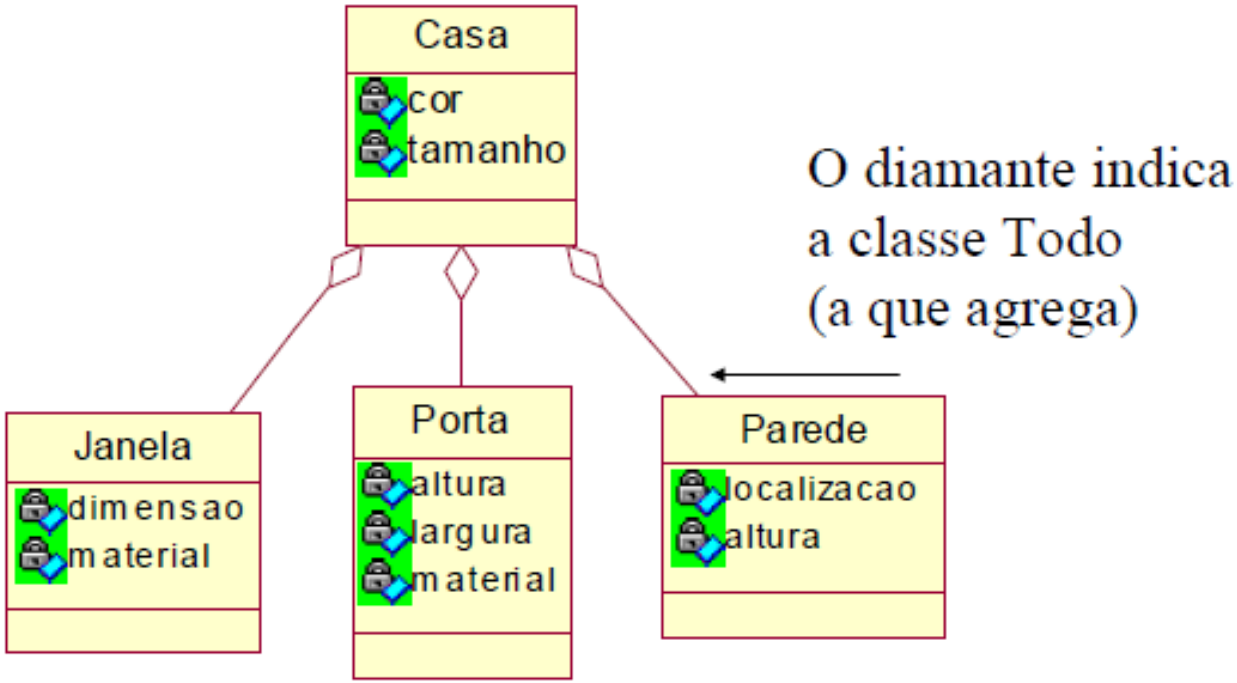
1. Relacionamento entre Classes

- **Representação da Agregação**
  - Agregação é um tipo de relação entre objetos como todo/parte. Na agregação, um objeto agrega outro objeto, ou seja, torna um objeto externo parte de si mesmo pela utilização de seus métodos.
  - Nesta relação, um objeto poderá agregar **uma ou muitas instâncias** de um outro objeto. Para agregar muitas instâncias. Para agregar muitas instâncias, a forma mais simples é utilizando **arrays**.
  - Na UML, as agregações são representadas por uma associação com um losango do lado do 'todo'.



1. Relacionamento entre Classes

- **Representação da Agregação**



Exemplo

```php
<?php
// Cria a classe que gera as propriedades do produto
class Produtos
{
    // Propriedades
    public $nome;
    public $valor;

    // Configura as propriedades
    function __construct ( $nome = null, $valor = null ) {
        $this->nome  = $nome;
        $this->valor = $valor;
    }
}

// Cria o carrinho de compras
class CarrinhoCompras
{
    // Pega as propriedades do produto
    public $produtos;

    // Configura as propriedades do produto no array $this->produtos
    public function adiciona( Produtos $produto ) {
        $this->produtos[] = $produto;
    }

    // Exibe todos os produtos
    public function exibe() {
        foreach ( $this->produtos as $produto ) {
            echo $produto->nome . '<br>';
        }
    }
}
```

1. Relacionamento entre Classes

- **Agregação – Continuação do exemplo**

```php
// Cria duas instâncias da classe Produtos
$produto1 = new Produtos('PlayStation');
$produto2 = new Produtos('Xbox');

// Cria uma instância da classe CarrinhoCompras
$carrinho = new CarrinhoCompras();

// Adiciona os produtos ao carrinho
$carrinho->adiciona( $produto1 );
$carrinho->adiciona( $produto2 );

// Exibe os dados na tela
$carrinho->exibe();
```

1. Relacionamento entre Classes

- **Agregação - Resumo**

Acontece quando um objeto precisa de outro objeto para completar a sua ação (Todo/Parte), ou seja, faz a agregação de um objeto externo e o utiliza como parte de si própria.

1. Relacionamento entre Classes

- **Composição**
  - Outra variação do tipo associação

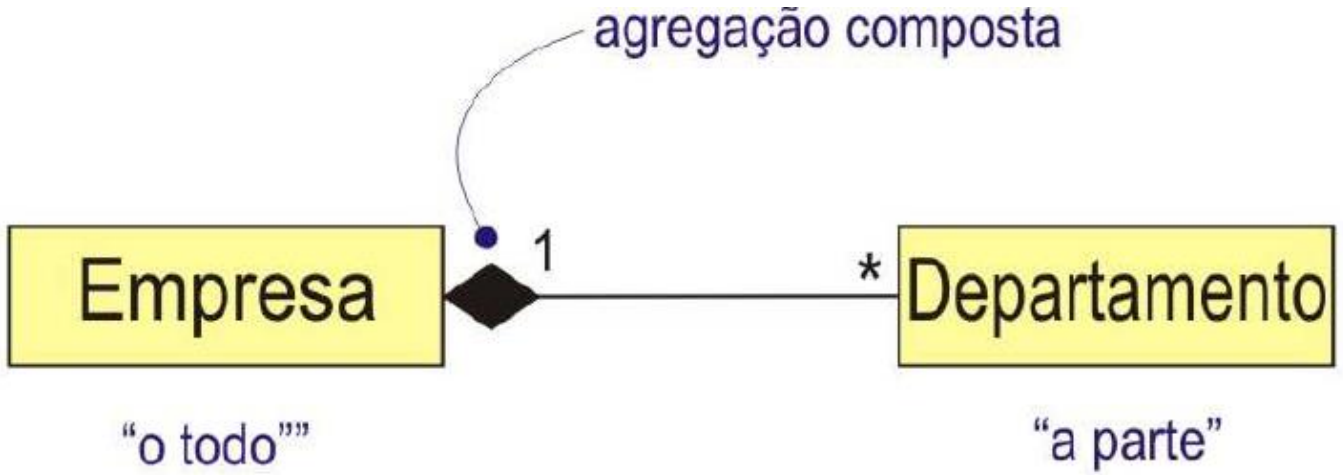
Na agregação, ao destruirmos o objeto "todo", as "partes" permanecem na memória por terem sido criadas fora do escopo da classe "todo".

- Na composição, o objeto-pai ou "todo" é **responsável** pela **criação** e **destruição** de suas partes. O objeto-pai "possui" as instâncias de suas partes.
- Representa um vínculo **mais forte** entre objetos - todo e objetos - parte
- O **todo não existe** (ou não faz sentido) sem as partes
- **As partes não existem sem o todo**

1. Relacionamento entre Classes

- **Representação da Composição**

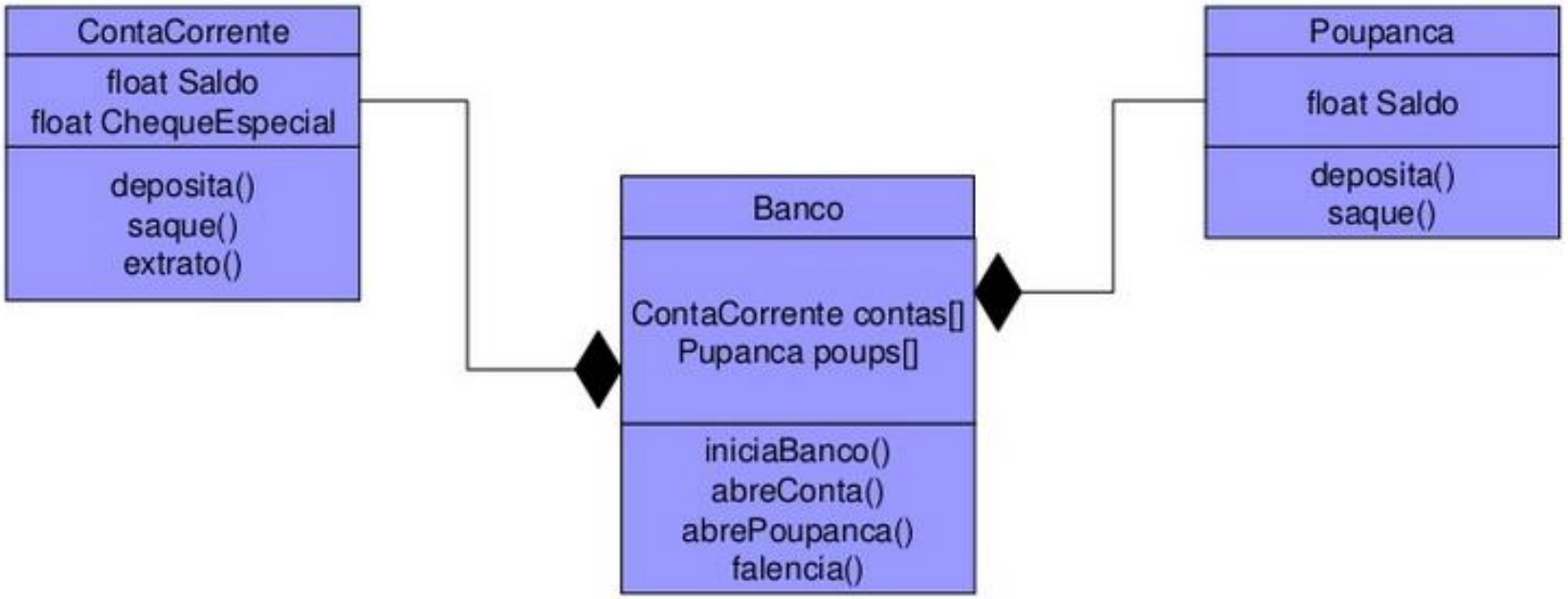
  Na UML, as Composições são representadas por um losango a cheio do lado do 'todo'.



2. Diagrama de Classes

Relacionamentos entre classes

- **Resumo sobre Composição**



Nota : No caso desta composição uma vez que o **Objeto** banco for destruído todas os objetos **Poupanca** e **ContaCorrente** deverão ser destruídos também.

1. Relacionamento entre Classes

- **Exemplo em Java sobre Composição**

```
public class Poupanca {
        float Saldo;

        void saque() {
                Saldo -= 10.0f;
                System.out.println("Novo Saldo →" + Saldo);
        }
        void deposito() {
                Saldo += 10.0f;
                System.out.println("Novo Saldo →" + Saldo);
        }    }
public class ContaCorrente {
        float Saldo;

        void saque() {
                Saldo -= 100.0f;
                System.out.println("Novo Saldo →" + Saldo);
        }
        void saque() {
                Saldo -= 100.0f;
                System.out.println("Novo Saldo →" + Saldo);
        }    }
```

```
public class Banco {
        Poupanca[] pops;
        ContaCorrente[] cc;
        int numConta, numPoupanca;
        void iniciaBanco() {
                pops = new Poupanca[100];
                cc = new ContaCorrente[100];
                numConta = 1;
                numPoupanca = 1;
        }
        void abreConta() {
                cc[ numConta ] = new ContaCorrente();
                numConta++;
        }
        void abrePoupanca() {
                pops[ numConta ] = new Poupanca();
                numPoupanca++;
        }
        void falencia() {
                for (int i = 0; i< 100; i++) {
                   pops[ i ] = null;
                   cc[ i ] = null;
                }
        }    }
```

```php
<?php
// Cria uma classe que configura o nome da pessoa
class Pessoa
{
    // Uma função que apenas adiciona "Nome: " no valor
    public function configura ( $nome ) {
        // Retorna
        return "Nome: " . $nome;
    }
}

// Cria uma classe para exibir dados
class Exibe
{
    // Configura as propriedades
    // $pessoa será a instância da classe Pessoa
    public $pessoa;

    // Este será apenas um nome
    public $nome;

    // Configura as propriedades
    function __construct( $nome = null ){
        // Faz a instância da class Pessoa
        $this->pessoa = new Pessoa();

        // Configura o valor do nome
        $this->nome = $nome;
    }

    public function exibe() {
        // Utiliza um método da classe Pessoa para exibir o nome enviado
        echo $this->pessoa->configura( $this->nome );
    }
}
```

- **Exemplo em PHP sobre Composição**

1. Relacionamento entre Classes

- **Exemplo em PHP sobre Composição**

```php
// Faz a instância da classe Exibe (todo)
$exibe = new Exibe('Gil Jader');

// Retorna: 'Nome: Gil Jader'
$exibe->exibe();
```

REFERÊNCIAS

- Dall'Oglio, Pablo. Php 5 - PHP: Programando com Orientação a Objetos. 3ª Edição, São Paulo, Novatec, 2015.
- WALLACE, Soares. Php 5 - Conceitos, Programação e Integração com Banco de Dados. 7ª Edição, São Paulo, Erica, 2013.
- http://www.php.net/manual/pt_BR/index.php