Herança e Sobrescrita

profº Mauricio Conceição Mario

Exemplo: superclasse ou classe-mãe → Conta_Bancária.class.php.

Conta_Bancária.class.php

```php
<?php
 class Conta_Bancária
{
    protected $agencia;
    protected $conta;
    protected $saldo;

    public function __construct($agencia, $conta, $saldo)
    {
        $this->agencia = $agencia;
        $this->conta   = $conta;
        if ($saldo >= 0) {
            $this->saldo   = $saldo;
        }
    }
    public function getInfo()
    {
        return "Agência: {$this->agencia}, Conta: {$this->conta}";
    }
    public function depositar($quantia)
    {
        if ($quantia > 0) {
            $this->saldo += $quantia;
        }
    }

    public function getSaldo()
    {
        return $this->saldo;
    }
}
?>
```

(Dall'Oglio, 2014)

Subclasse ou classe-filha: Conta_Corrente.php.

Conta_Bancária.class.php | Consignado.php | Conta_Poupanca.php | aplica_herança2.php | Conta_Corrente.php

```php
<?php
class Conta_Corrente extends Conta_Bancária
{
    protected $limite;

    public function __construct($agencia, $conta, $saldo, $limite)
    {
        parent::__construct($agencia, $conta, $saldo);
        $this->limite = $limite;
    }

    public function retirar($quantia)
    {
        if ( ($this->saldo + $this->limite) >= $quantia ) {
            $this->saldo -= $quantia; // retirada permitida
        }
        else {
            return false; // retirada não permitida
        }
        return true;
    }
}
?>
```

parent::__construct (...) → acesso ao método construtor da superclasse

(Dall'Oglio, 2014)

Subclasse ou classe-filha: Conta_Poupança.php.

Conta_Bancária.class.php | Consignado.php | Conta_Poupanca.php | aplica_herança2.php

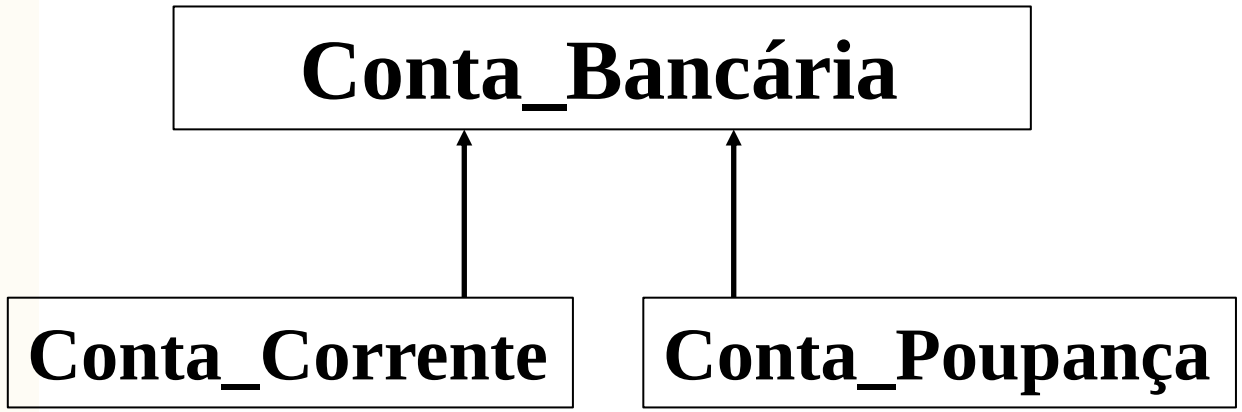
```php
<?php
 class Conta_Poupanca extends Conta_Bancária
{

    public function __construct($agencia, $conta, $saldo)
    {
        parent::__construct($agencia, $conta, $saldo);

    }

    function retirar($quantia)
    {
        if ($this->saldo >= $quantia) {
            $this->saldo -= $quantia;
        }
        else {
            return false; // retirada não permitida
        }
        return true; // retirada permitida
    }
}
?>
```

Diagrama de Classes → hierarquia:



(Dall'Oglio, 2014)

- Sobrescrita de métodos ou *Overriding*: o comportamento de um método da superclasse é modificado nas classes-filha, dando-lhe funcionalidades diferentes. Exemplo: acrescido o método contabilizar(..) nas classes da aplicação.

Conta_Bancária.class.php

```php
    public function contabilizar($saldo)
    { $this-> saldo += 60.55;
    print "Acesso pela classe Conta_Bancária <br>\n";
    }
}
?>
```

Conta_Corrente.php

```php
    public function contabilizar($saldo)
    { $this-> saldo -= 0.85;
    print "Acesso pela classe Conta_Corrente <br>\n";
    }
}
?>
```

Conta_Poupanca.php

```php
    public function contabilizar($saldo)
    { $this-> saldo += 110.43;
    print "Acesso pela classe Conta_Poupanca <br>\n";
    }
}
?>
```

aplicação: aplica_herança4.php.

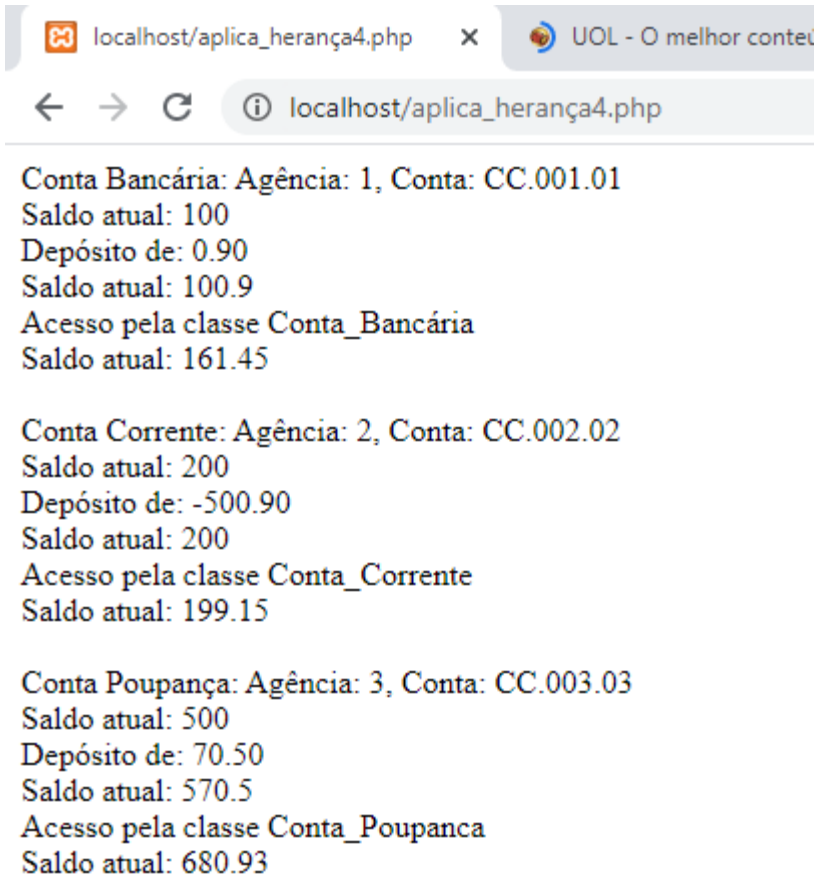
```php
<?php
require_once 'Conta_Bancária.class.php';
require_once 'Conta_Poupanca.php';
require_once 'Conta_Corrente.php';

/* criação de objetos tipo Conta_Corrente e Conta_Poupança
+ Conta_Bancária*/
$cb = new Conta_Bancária(000001, "CC.001.01", 100);
$cc = new Conta_Corrente(000002, "CC.002.02", 200, 200);
$cp = new Conta_Poupanca(000003, "CC.003.03", 500);

    print "Conta Bancária: {$cb->getInfo()} <br>\n";
    print "    Saldo atual: {$cb->getSaldo()} <br>\n";
    $cb->depositar(0.90);
    print "    Depósito de: 0.90 <br>\n";
    print "    Saldo atual: {$cb->getSaldo()} <br>\n";
    $cb->contabilizar(0.66);
    print "    Saldo atual: {$cb->getSaldo()} <br>\n";

    print "<br>\n Conta Corrente: {$cc->getInfo()} <br>\n";
    print "    Saldo atual: {$cc->getSaldo()} <br>\n";
    $cc->depositar(-500.90);
    print "    Depósito de: -500.90 <br>\n";
    print "    Saldo atual: {$cc->getSaldo()} <br>\n";
    $cc->contabilizar(-1.66);
    print "    Saldo atual: {$cc->getSaldo()} <br>\n";

    print "<br>\n Conta Poupança: {$cp->getInfo()} <br>\n";
    print "    Saldo atual: {$cp->getSaldo()} <br>\n";
    $cp->depositar(70.50);
    print "    Depósito de: 70.50 <br>\n";
    print "    Saldo atual: {$cp->getSaldo()} <br>\n";
    $cp->contabilizar(3.36);
    print "    Saldo atual: {$cp->getSaldo()} <br>\n";
?>
```



Exercício proposto:

13. Criar uma classe “Rendimentos.php”, filha de “Conta_Poupança.php”, e com o atributo “juros”. Criar nesta classe uma versão do método sobrescrito “contabilizar”, onde através do atributo “juros” o “saldo” possa ser atualizado. Adaptar também a aplicação para que se possa acessar propriedades de Rendimentos e das classes que estão acessíveis a ela através da hierarquia de Herança.

Referências Bibliograficas

- *php* Programando com Orientação a Objetos
  Pablo Dall'Oglio - editora Novatec - 2014

- Desenvolvimento web com PHP e MySQL
  Evaldo Junior Bento – editora Casa do Código - 2018