Minicurso PHP 5

Vinícius Costa de Souza

viniciuscs@unisinos.br

Julho de 2005

Apresentações

- Nome
- Experiências com programação
- Conhecimentos de linguagens WEB / HTML
- Expectativas referentes ao curso

Programa

- Introdução
- Manipulação de dados
- Operadores
- Estruturas de controle
- Funções
- PHP e formulários HTML
- Manipulação de arquivos
- Sessões e Cookies
- PHP OO

Introdução

O que é PHP?

- *Hypertext Processor*
- linguagem de script para a Web
- *server side*
- embutida no HTML

Diferenças?

- PHP x HTML
- PHP x JavaScript
- PHP x CGI

Introdução

Por que utilizar PHP?

- linguagem mais utilizada na Web
- código-fonte aberto
- possui muitos recursos prontos (extensões)
- amplo suporte a bancos de dados
- portável
- estável
- rápido
- fácil de aprender

Introdução

Material na Web

- http://www.php.net
- http://zend.com
- http://phpfaqts.com
- http://phpbuilder.com
- www.phphub.com
- www.phppaterns.com
- www.phpclasses.org
- www.sourceforge.net
- http://pear.php.net
- http://phpbrasil.com

Introdução

Noções básicas de programação em PHP

- tags limitadoras de um programa PHP
- comentários (como e porquê?)
- comandos para saída na tela
- finalização das linhas de comando
- PHP dentro de código HTML
- HTML dentro do código PHP
- caractere identificador de variável

Introdução

Exercício

Criar uma página em PHP para exibição da data e hora atual.

Manipulação de dados

Numéricos

- inteiro (5)
- real (4.432)

Alfanuméricos

- aspas simples (')
- aspas duplas (")
- aspas invertidas (´)
- caractere de controle (\ )

Manipulação de dados

Variáveis

- não é necessário declarar as variáveis em PHP
- iniciam sempre com o caractere **$**
- não podem iniciar com números
- PHP é case-sensitive

Constantes

- armazenam dados que não são alterados
- referencia-se diretamente pelo nome (sem $)

```
define (<nome_constante>,<valor>);
```

Manipulação de dados

Arrays

- podem armazenar mais de um valor, pois possuem além de um nome identificador um índice que pode ser numérico ou textual
- o índice aparece entre colchetes [] e após o nome
- índices numéricos iniciam sempre em zero
- os valores atribuídos podem ser de tipos diferentes
- podem ser uni ou multidimensionais

Manipulação de dados

Exercício

Crie um array chamado *estado* para armazenar neste as capitais dos estados na região sul do Brasil, utilizando como índice as sigla dos estados.

Operadores

Aritméticos

+	ADIÇÃO
-	SUBTRAÇÃO
*	MULTIPLICAÇÃO
/	DIVISÃO
-oper	TROCA SINAL
++oper	PRÉ-INCREMENTO
--oper	PRÉ-DECREMENTO
oper++	PÓS-INCREMENTO
oper--	PÓS-DECREMENTO

Operadores

Condicionais

```
X == Y      →     X IGUAL A Y

X >= Y      →     X MAIOR OU IGUAL A Y

X <= Y      →     X MENOR OU IGUAL A Y

X != Y      →     X DIFERENTE DE Y

X <> Y      →     X DIFERENTE DE Y

X > Y       →     X MAIOR QUE Y

X < Y       →     X MENOR QUE Y
```

Operadores

Atribuição

```
$num = 5;

$num += 5;          →  $num = $num + 5;

$num -= 5;          →  $num = $num - 5;

$num *= 5;          →  $num = $num * 5;

$num /= 5;          →  $num = $num / 5;

$nome .= "aluno";   →  $nome = $nome."aluno";
```

Operadores

Lógicos

```
   !X          → verdadeiro se X for falso

X AND Y        → verdadeiro se X e Y forem verdadeiros

X OR Y         → verdadeiro se X ou Y forem verdadeiros

X XOR Y        → verdadeiro se apenas um for verdadeiro

X && Y         → verdadeiro se X e Y forem verdadeiros

X || Y         → verdadeiro se X ou Y forem verdadeiros
```

Operadores

Exercício

Quais são os valores das variáveis $a, $b, $c, $x, $y e $z após a execução do seguinte programa:

```
$a=2;
$b=4;
$c=6;
$x= --$c + $b;
$y= $b++ + $a;
$z= $a - $b--;
```

Estruturas de controle

Condicional (if ... else)

```
$nota = ($N1 + 2*$N2)/3;
if ( $nota > 9.5 )
   $resultado = "Aprovado com distinção";
elseif ( ($nota >= 8) and ($nota <= 9.5) )
   $resultado = "Aprovado plenamente";
elseif  ( ($nota >= 6) and ($nota <= 7.9) )
   $resultado = "Aprovado";
else
   $resultado = "Reprovado";
echo "Você  foi $resultado em seu TCC";
```

Estruturas de controle

Condicional (switch)

```
switch  ($opcao){
    case "s":
        echo "Você escolheu a opção SIM";
        break;
    case "n":
        echo "Você escolheu a opção NÃO";
        break;
    default:
        echo "A opção digitada é inválida";
        break;
}
```

Estruturas de controle

Repetição (while)

```
$cont = 1;
while ( $cont < 10 ){
   echo "O valor atual do contador é $cont <br>";
   $cont++;
}
```

Repetição (do ... while)

```
   $cont = 0;
   do {
       $cont++;
       echo "O valor atual do contador é $cont <br>";
   } while ($cont < 10);
```

Estruturas de controle

Repetição (for)

```
for ($cont=100; $cont >= 0; $cont--):

    echo "O valor da variável \$cont é $cont <br>";

endfor;
```

Repetição (foreach)

```
$vetor = array ("um"=>1, "dois"=>2, "três"=>3);
foreach ($vetor as $chave => $valor){
    echo "O valor de \$vetor[$chave] é $valor <br>";
}
```

Estruturas de controle

Exercício

Crie um array chamado **curso** que armazene as seguintes informações sobre esse curso:

- nome
- data
- carga horária
- local

Utilize **strings** para os índices do array.
Utilize **foreach** para imprimir o array neste formato:

Nome do curso: PHP 5
Informação armazenada em $curso[nome]

Funções

- torna os programas mais organizados e modulares
- uma função pode ou não receber argumentos em sua chamada (por valor ou por referência →&)
- o comando **return** é opcional e serve para que a função retorne um valor
- sempre que o PHP encontra uma chamada para uma função, a execução do programa é interrompida e o fluxo de execução passa para o início da função

Funções

```
function lista_aprovados($alunos){
    for ($i=0; $i < sizeof($alunos); $i++){
       $media = ($alunos[$i]["n1"] + 2*$alunos[$i]["n2"])/3;
       if ($media >= 6){
           $aprovados[] = $alunos[$i]["nome"];
       }
    }
    return $aprovados;
}
```

Funções

Pré-definidas

array - cria um array

sort - ordena um array

sizeof - obtém o número de elementos de um array

strchr - encontra a primeira ocorrência de um caractere

strlen - obtém o tamanho de uma string

substr - retorna uma parte da string

split - subdivide uma string em várias strings

str_replace - substitui as ocorrências de uma string

date - formata data e hora

Funções

include e require

- funções que permitem reaproveitar funções ou arquivos, utilizando-os em diversas páginas do site.
- tem por objetivo incluir um arquivo dentro de outro.

```
include "nome_arquivo.inc.php";

require "nome_arquivo.inc.php";
```

Funções

Exercício

Criar uma função que recebe um array chamado alunos que contenha o nome, nota 1 e nota 2 dos seguintes alunos.

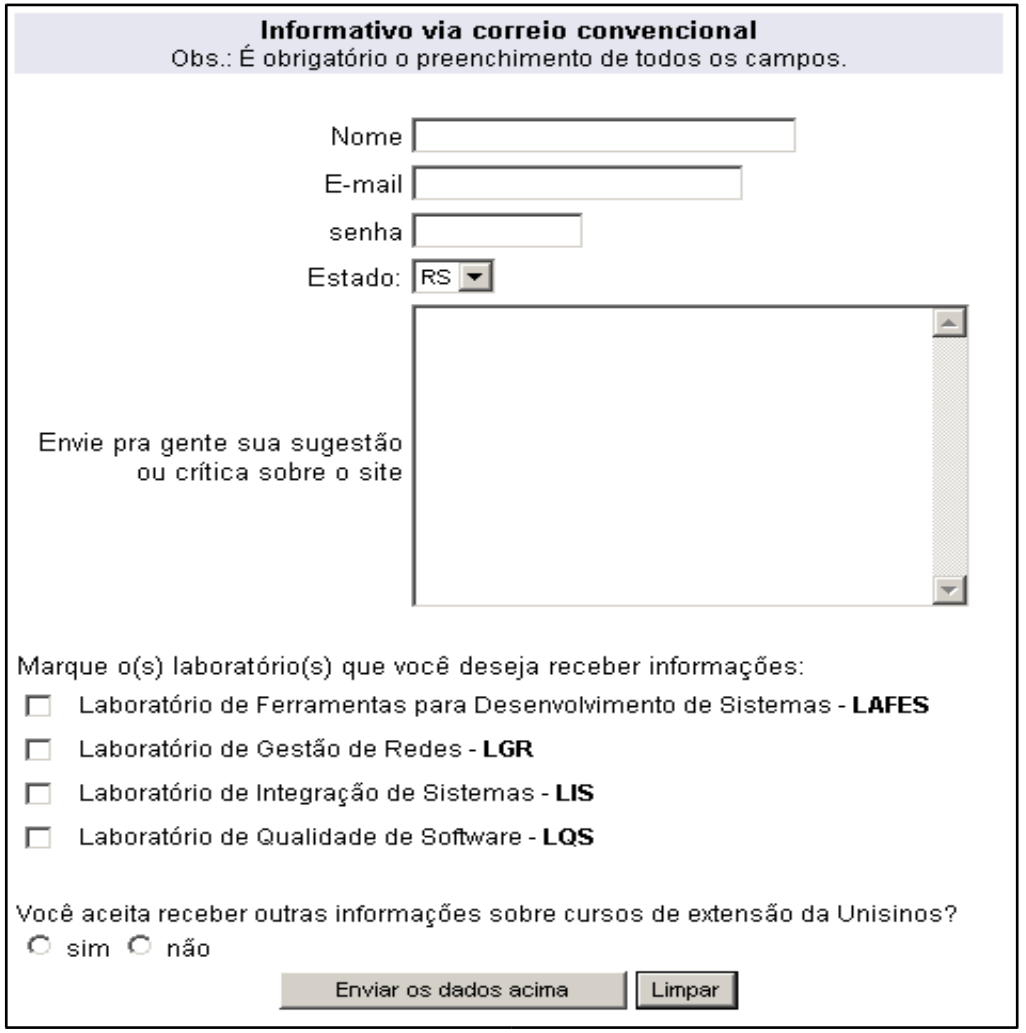
Nome	N1	N2
Aline dos Santos	6,5	2,9
Bianca da Silva	7,8	8,6
Carlos Pedroso	5,6	5,9
Eduardo Romero	3,7	8,6
Fabiane Almeida	9,1	6,7

A função deve calcular a média (N1 + 2*N2)/3 e retornar um outro array chamado aprovados que deve armazenar o nome e a média dos alunos aprovados. O programa para imprimir a listagem dos aprovados, em ordem alfabética, com suas médias finais.

PHP e Formulários HTML

- criados em HTML
- possuem no mínimo:

5. um campo para entrada de dados
6. um botão para enviar os dados
7. endereço de destino para os dados enviados



PHP e Formulários HTML

```
<FORM action="cadastro.php" method="post">
    <INPUT type="text" name="nome" size="30" maxlength="150">
    <INPUT type="password" name="senha" size="10" maxlength="150">
    <SELECT name="select">
        <option value="RS">RS</option>
        <option value="SC">SC</option>
        <option value="PR">PR</option>
    </SELECT>
    <TEXTAREA name="COMENTARIO" cols="30" rows="10" wrap="physical">
    </TEXTAREA>
    <INPUT type="checkbox" name="fds" value="FDS">
    <INPUT type="checkbox" name="lgr" value="LGR">
    <INPUT type="radio" name="noticias" value="sim">
    <INPUT type="radio" name="noticias" value="nao">
    <INPUT TYPE="submit" value="Enviar os dados acima">
    <INPUT TYPE="reset" value="Limpar">
</FORM>
```

PHP e Formulários HTML

Métodos de envio

- GET
  - método padrão, através do qual os dados são enviados com o nome da página que receberá os dados
  - www.site.com.br/programa.php?nome=vinicius&idade=27
  - desvantagens ?
- POST
  - envia os dados por meio do corpo da mensagem enviada ao servidor
  - www.site.com.br/programa.php
  - vantagens?

PHP e Formulários HTML

Tratando os dados recebidos

- **como variáveis** - apenas acrescenta-se o símbolo $ antes do nome definido no form

- **como array do PHP** – arrays definidos pelo PHP que armazenam as informações enviadas pelo

  - GET → **$_GET**
  - **P**OST → **$_POST**

  Neste caso, as chaves dos arrays são os nomes dos campos do formulário HTML

PHP e Formulários HTML

Exercício

Crie 3 arquivos (cadastro1.php, cadastro2.php e fim_cadastro.php)

- No arquivo **cadastro1.php** crie um form com os seguintes campos, que devem ser enviados para o arquivo cadastro2.php:
  - Nome (text - não pode estar vazio e deve ter sobrenome)
  - Sexo (radio)
- No arquivo **cadastro2.php** crie um form com os campos abaixo, que devem ser remetidos para o arquivo fim_cadastro.php:
  - Senha (password com mínimo de 5 caracteres)
  - Comentário: (textarea de 30 colunas e 6 linhas)
- O arquivo **fim_cadastro.php** deve receber e imprimir na tela todos os dados do cadastro (form1 e form2)

Manipulação de arquivos

Quando necessitamos armazenar poucos dados, podemos fazer isso em arquivos texto para termos um acesso mais rápido as informações.

Através do PHP podemos:

- abrir um arquivo → fopen
- ler um arquivo → fread
- escrever em um arquivo → fwrite
- fechar um arquivo → fclose

Manipulação de arquivos

- **modos possíveis para a função fopen**

  r → abre somente para leitura (ponteiro no início)

  r+ → abre para leitura e escrita (ponteiro no início)

  w → abre somente para escrita (cria ou zera)

  w+ → abre para leitura e escrita (cria ou zera)

  a → abre somente para escrita (ponteiro no final)

  a+ → abre para leitura e escrita (ponteiro no final)

Manipulação de arquivos

- **Exemplos**

Leitura

```
$arq = fopen ("nome_arquivo.txt", "r");
while (!feof ($arq)) {
      $linha = fread($arq,1024);
      echo $linha;
}
fclose ($arq);
```

Escrita

```
$arquivo = fopen("cadastro.txt","a+");
fwrite ($arquivo,$nome."##".$email."\n");
fclose($arquivo);
```

Manipulação de arquivos

Exercício

Modificar o arquivo fim_cadastro.php para que os dados do cadastro sejam armazenados em arquivo (cadastros.txt).

Após, faça testes simulando cadastros e consultando o arquivo para verificar se os dados estão sendo armazenados.

Atenção com a permissão para escrita no arquivo

Sessões e Cookies

Sessão

- trata-se de um período de tempo enquanto uma pessoa particular navega por determinado site.

- através das sessões, podemos registrar variáveis, as quais estarão disponíveis em todas as páginas enquanto o usuário estiver navegando pelo site.

Sessões e Cookies

Sessão

```
session_start();

session_register("nome_usuario");

session_register("login_usuario");

session_register("senha");

$_SESSION['nome_usuario'] = $nome_banco;

$_SESSION['login_usuario'] = $login_banco;

$_SESSION['senha'] = $senha_banco;
```

Sessões e Cookies

Cookies

- pequenos fragmentos de informação retidos na máquina do cliente, quer na memória do Browser, quer em um arquivo gravado no HD.
- cada cookie contém um par nome/valor.
- ***configurar*** um cookie significa associar um par valor/nome e armazenar no lado cliente.
- ***obter*** ou ***ler*** um cookie significa utilizar o nome para recuperar o valor
- vantagens e desvantagens

Sessões e Cookies

Cookies

- Sintaxe

```
setcookie(nome,valor,validade,caminho,dominio,seguro);
```

- Exemplo

```
setcookie("user","joao",time()+86400);
```

- Três dicas importantes

Sessões e Cookies

Exercício

Modificar os arquivos cadastro1.php, cadastro2.php e fim_cadastro.php para que os dados dos formulários sejam gravados em sessão (cadastro 1 e 2) e recuperados da sessão (fim_cadastro).

PHP OO

Introdução

Classe - tipo de dado definido com atributos e métodos

Objeto - uma instância de uma classe

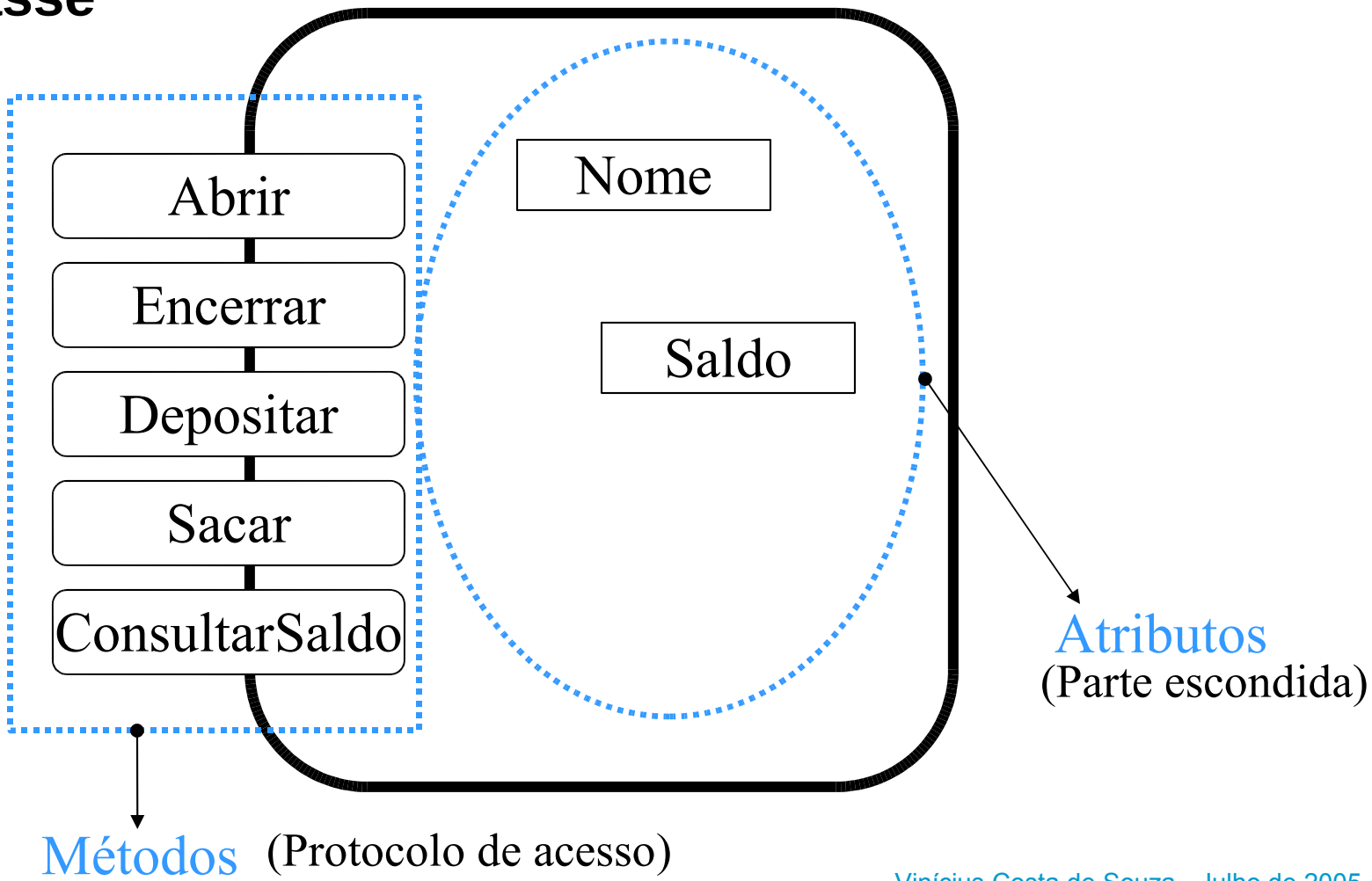
Atributo - dados de uma classe

Método - funções de uma classe

Herança - extensão de uma classe

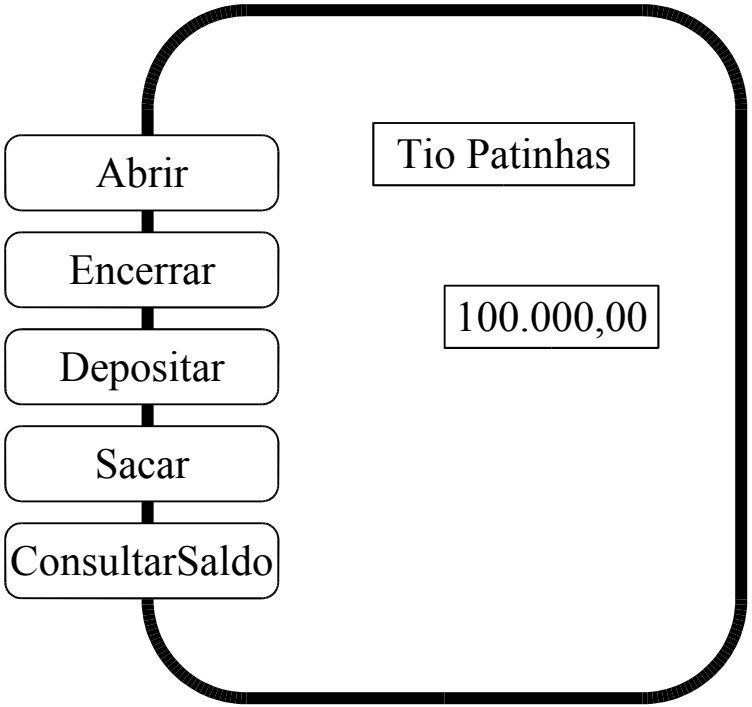
PHP OO

Classe

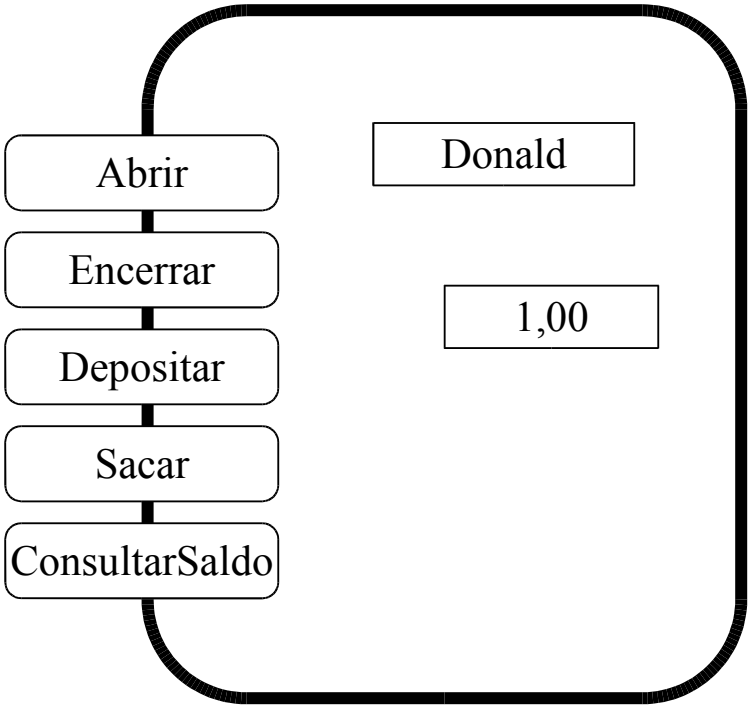


PHP OO

Objetos



Conta do Tio Patinhas



Conta do Donald

PHP OO

Classe

Construtor

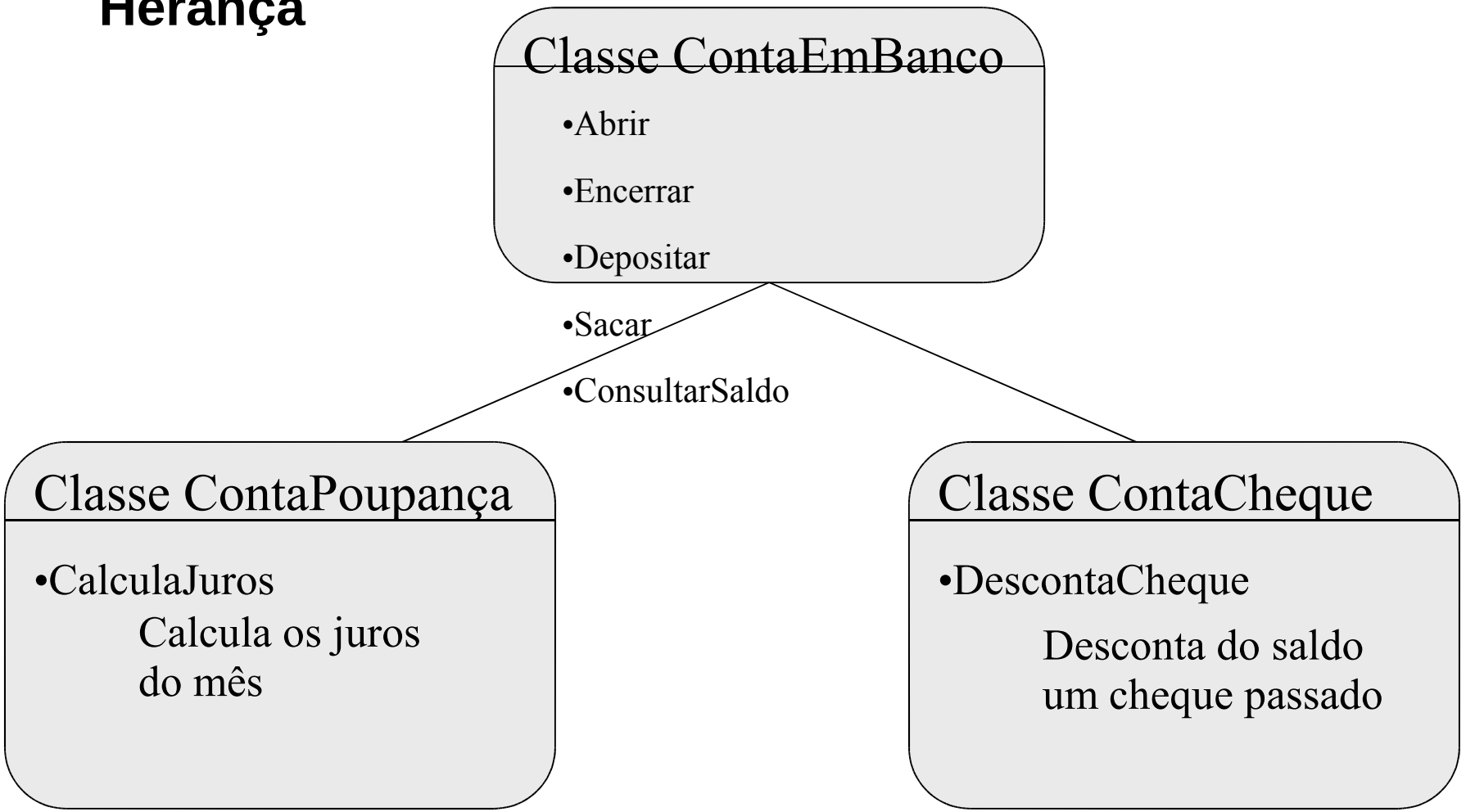
Métodos

Instância

```
class ContaEmBanco {
      var nome,saldo;
      function ContaEmBanco($nome,$valor){
            $this->nome = $nome;
            $this->saldo = $valor;
      }
      function depositar($valor){
            $this->saldo += $valor;
      }
      function consultarSaldo(){
            echo $this->saldo;
      }
      ...
}
$conta1 = new ContaEmBanco("Donald",100);
$conta1->depositar(500);
$conta1->consultarSaldo();
```

PHP OO

Herança



PHP OO

Sobrescrevendo métodos

```
class ContaCheque extends ContaEmBanco {
      var num_cheques = 0;
      function ContaCheque($nome,$valor){
            $this->nome = $nome;
            $this->saldo = $valor;
      }
      function consultarSaldo(){
            echo $this->saldo;
            $this->saldo -= 0,40;
      }
      ...
}
$contaCh = new ContaCheque("Pluto",100);
$contaCh->consultarSaldo();
```

PHP OO

PHP 5

- Modelo OO (Zend 1)
  - objetos copiados
- Novo modelo OO (Zend 2)
  - referências a objetos
  - *private, public, protected, abstract*
  - *permite construtores e destrutores*
  - *controle de duplicação*
  - set e get

PHP OO

PHP 5 – objetos por referência

```
class Conta {
	function setSaldo($value){
		$this->saldo = $value;
	}
	function getSaldo(){
		return $this->saldo;
	}
}

function zeraConta($obj){
	$obj->setSaldo(0);
}

$object = new Conta();
$object->setSaldo(100);
seraConta($object);
echo $object->getSaldo();
```

PHP OO

PHP 5 – referenciando objetos retornados

```
class Conta {
        function Conta($valor){
                $this->saldo = $valor;
        }
        funciton getSaldo(){
                return $this->saldo;
        }
}

function criarConta($valor){
        return new Conta($valor);

}

PHP4:
$conta1 = criarConta(250);
echo $conta1->getSaldo();

PHP5:
echo criarConta(250)->getSaldo();
```

PHP OO

PHP 5 – construtores

```
class ClasseBase {
   function __construct() {
       print "No construtor da ClasseBase\n";
   }
}

class SubClasse extends ClasseBase {
   function __construct() {
       parent::__construct();
       print "No construtor da SubClasse\n";
   }
}

$obj = new ClasseBase();
$obj = new SubClasse();
```

PHP OO

PHP 5 – destrutores

```
class MinhaClasse {
   function __construct() {
       print "No construtor\n";
       $this->name = "MinhaClasse";
   }

   function __destruct() {
       print "Destruindo " . $this->name . "\n";
   }
}

$obj = new MinhaClasse();
```

PHP OO

PHP 5 – protected

```
class Conta {
   protected $saldo;
}


class ContaPoupanca extends Conta{
   function getSaldo(){
       return $this->saldo;
   }
}

$conta1 = new ContaPoupanca;
echo $conta1->getSaldo();          → OK
echo $conta1->saldo;               → NÃO
```

PHP OO

PHP 5 – get e set

```
class Conta {
   function __get($atributo){
      return $this->members[$atributo];
   }
   function __set($atributo,$valor){
      $this->members[$atributo] = $valor;
   }
}
```

PHP OO

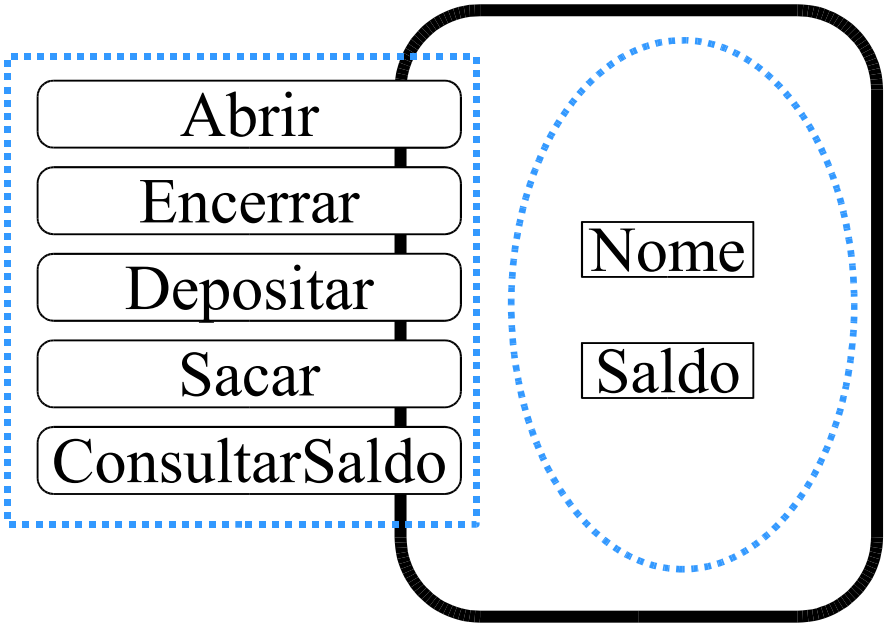
PHP 5 – abstract

```
abstract class Conta {
      //...
}

class ContaEspecial extends Conta{
      //...
}

$conta1 = new Conta;          → NÃO
$conta1 = new ContaEspecial;  → SIM
```

PHP OO

Exercício

Implementar através do paradigma OO uma conta bancária utilizando e testando os novos recursos do PHP5



OBRIGADO!

Vinícius Costa de Souza

viniciuscs@unisinos.br

www.inf.unsinos.br/~vinicius

São Leopoldo, julho de 2005