

The PHP logo is centered in the upper half of the slide. It consists of the lowercase letters 'php' in a bold, black, sans-serif font with a white outline. The logo is set against a light blue oval background that has a subtle radial gradient and a slight 3D effect. The entire slide background is a dark blue gradient with a large, faint circular shape on the right side.

**php**

[www.php.net](http://www.php.net)

Desenvolvimento Web com  
PHP

Felipe Ribeiro - [felipernb@gmail.com](mailto:felipernb@gmail.com)  
<http://feliperibeiro.com>  
ENECOMP 2008

# Quem é esse cara?

---

- Felipe Ribeiro
  - Estudante de Ciência da Computação na UFCG
  - Engenheiro de sistemas da *startup* americana Shoprizer.com
  - Experiência em sistemas distribuídos e desenvolvimento Web há 6 anos
  - <http://feliperibeiro.com>
  - [felipernb@gmail.com.br](mailto:felipernb@gmail.com.br)



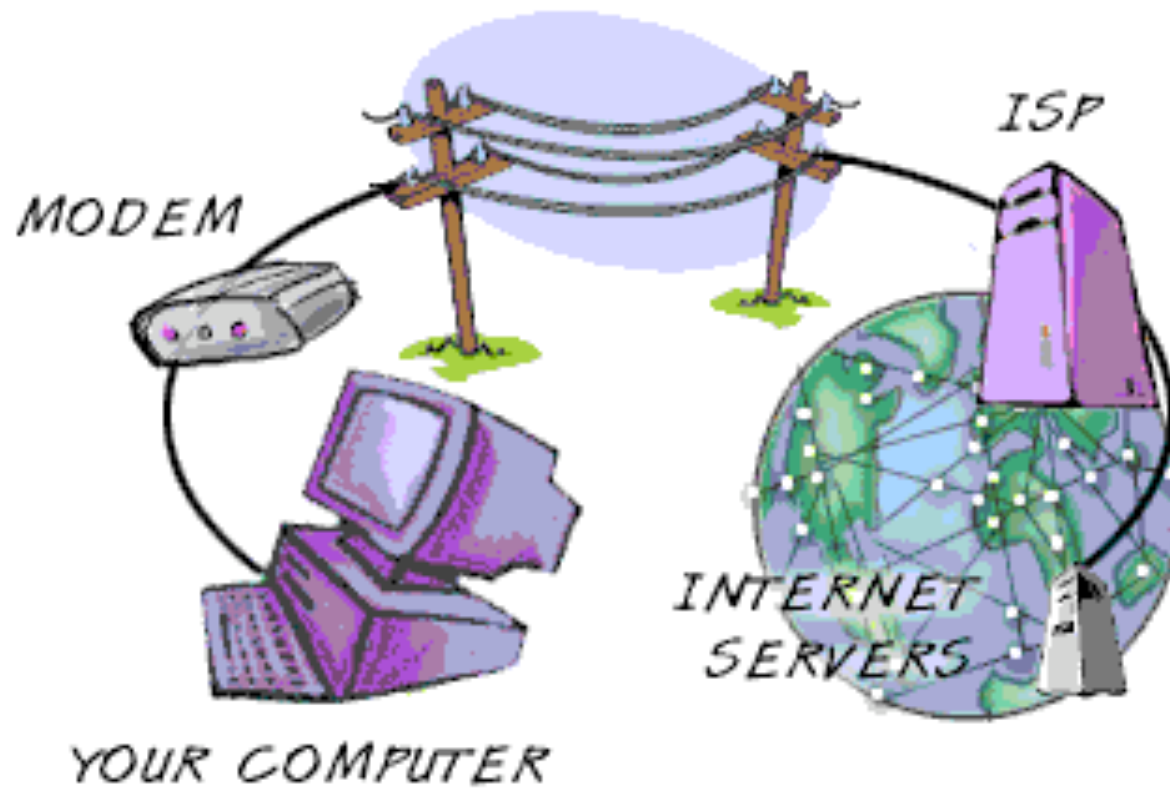
Como funciona a  
Web?

---



# Como funciona a Web?

---



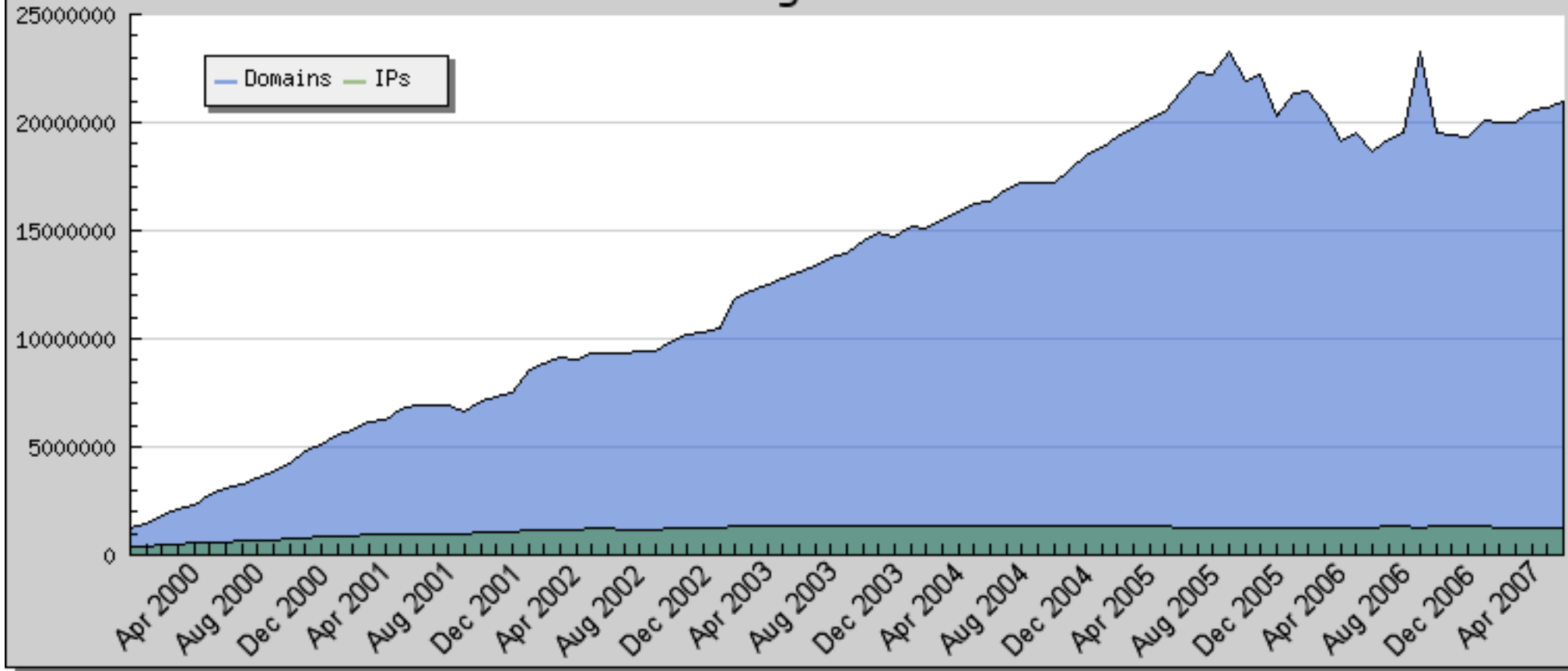
# Mas o que danado é PHP?

---

- É uma linguagem de programação focada na Web
- É uma linguagem de programação dinâmica (a exemplo de Python e Ruby)
- É open source, sob licença Apache
- É mantido pela comunidade com coordenação da Zend Technologies
- A sigla que originalmente significava Personal Home Page, hoje significa: PHP Hypertext Preprocessor



## PHP Usage for Jul 2007



PHP é popular!

Fonte: Netcraft



# Quem usa PHP?

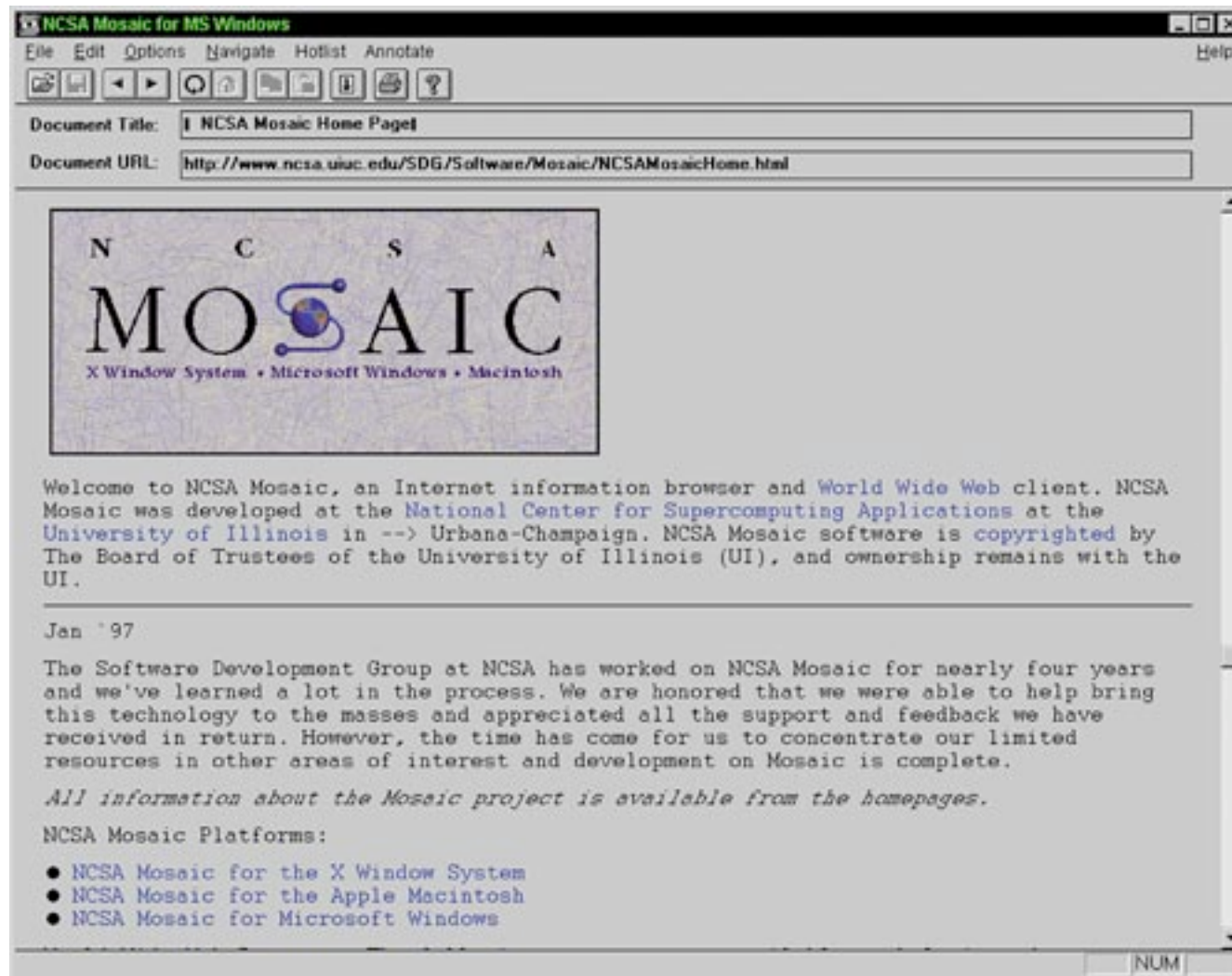
---

- Yahoo!
- Facebook
- Digg
- Flickr
- del.icio.us
- Wikipédia
- Sourceforge
- Last.fm
- E mais de 20 milhões de sites!



# Um pouco de história

1993



# Um pouco de história

---

1993 - Sistemas web muito simples, geralmente escritos em Perl

```
1 <HTML>
2   <HEAD>
3     <TITLE>My Personal Home Page</TITLE>
4   </HEAD>
5   <BODY>
6     This is my cool page<P>
7     And look at my counter<P>
8     <IMG SRC="/cgi-bin/counter.pl">
9   </BODY>
10 </HTML>
```



# Um pouco de história

---

## 1994 - Nasce o PHP

```
1 <!--getenv HTTP_USER_AGENT-->
2 <!--ifsubstr $exec_result Mozilla-->
3 Hey, you are using Netscape!<p>
4 <!--endif-->
5
6 <!--sql database select * from table where user='$username'-->
7
8 <!--ifless $numentries 1-->
9 Sorry, that record does not exist<p>
10 <!--endif exit-->
11
12 Welcome <!--$user-->!<p>
13 You have <!--$index:0--> credits left in your account.<p>
14
15 <!--include /text/footer.html-->
```



# Um pouco de história

---

1995 - PHP passa a se chamar PHP/FI

```
1 <?
2 $name = "bob";
3 $db = "db";
4 $result = mysql($db,"select * from table where firstname='$name'");
5 $num = mysql_numrows($result);
6 echo "$num records found!<p>";
7 $i=0;
8 while($i<$num);
9     echo mysql_result($result,$i,"fullname");
10    echo "<br>";
11    echo mysql_result($result,$i,"address");
12    echo "<br>";
13    $i++;
14 endwhile;
15 ?>
```



# Um pouco de história

---

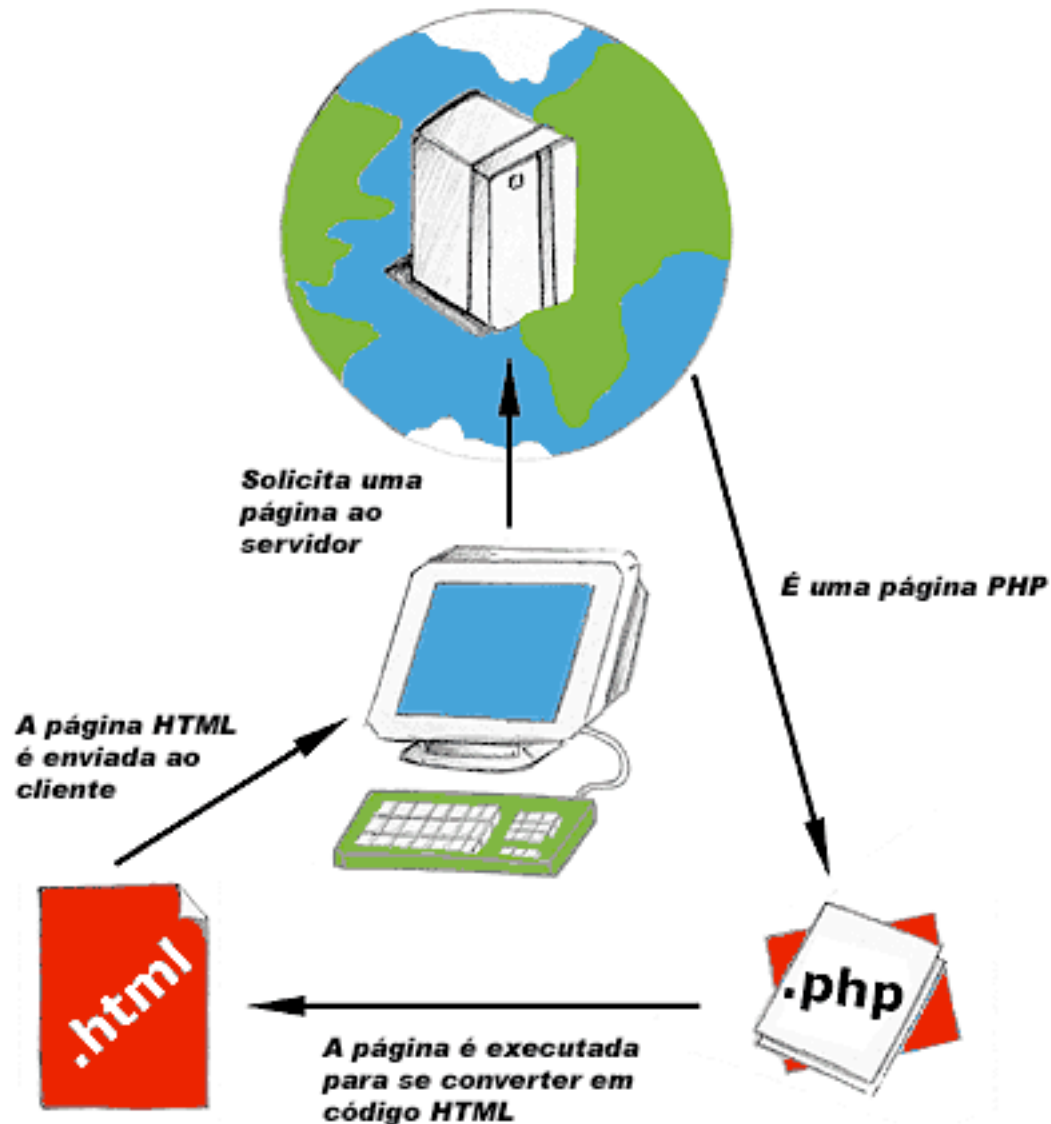
## 2005 - É lançado o PHP5

```
1 <?php
2 class SimpleClass {
3     private $attribute;
4
5     public function foo($bar) {
6         echo $this->attribute * $bar;
7     }
8 }
9
10 class ExtendedClass extends SimpleClass {
11
12     public function foo($bar) {
13         try {
14             do_something();
15         } catch(FooBarException $e) {
16             die($e->getMessage());
17         }
18         parent::foo($bar);
19     }
20 }
21
22 $o = new ExtendedClass;
23 $o->foo();
24 ?>
```

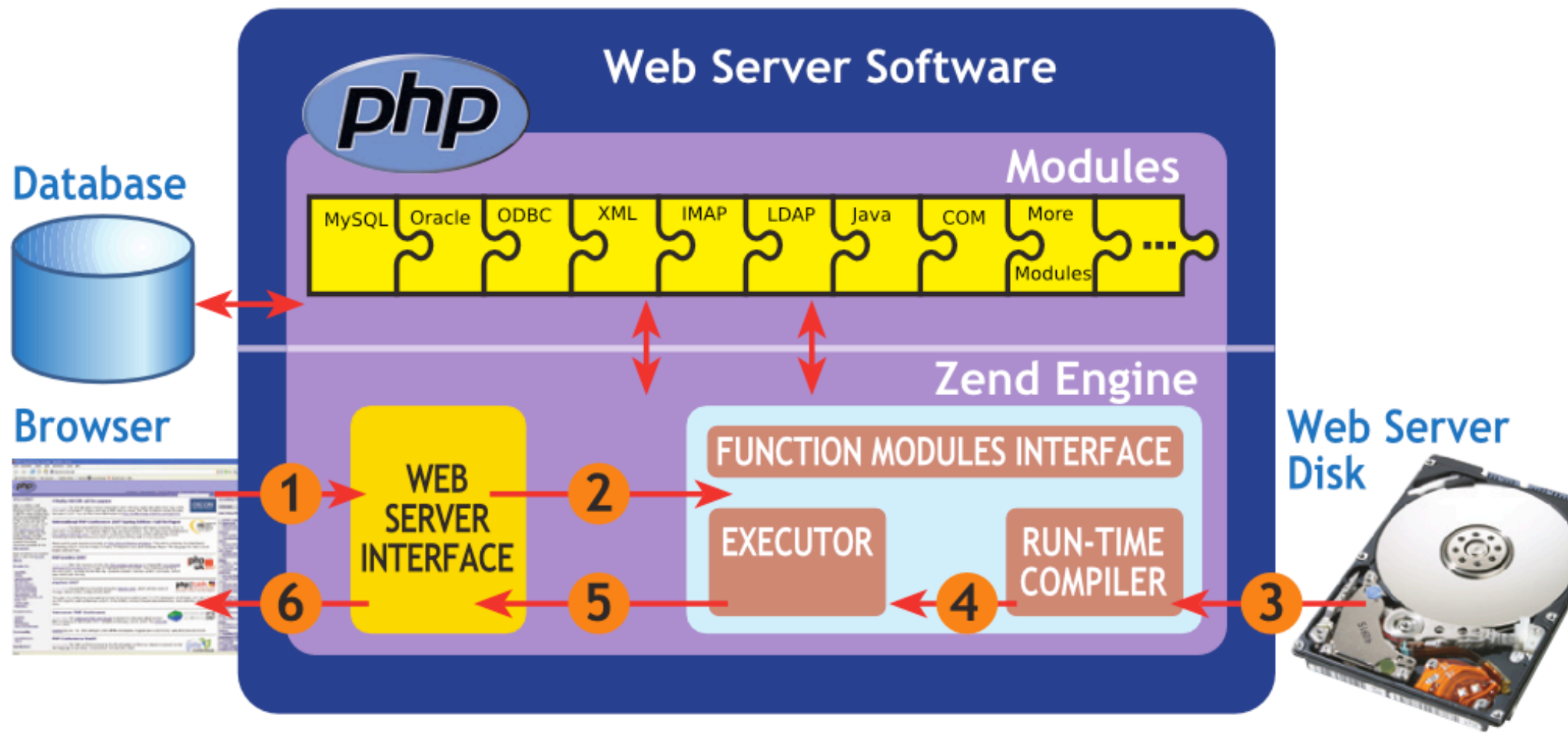


# Como funciona?

---



# The Zend Engine – the heart of PHP



Como funciona?

Fonte: Zend Technologies



# Vantagens do PHP

---

- Simplicidade
- Produtividade
- Escalabilidade
- Custo
- Comunidade
- Apache
- Deployment simples
- Independente de Plataforma



# Ambientes de desenvolvimento

---

- Existem diversos ambientes para desenvolvimento em PHP
  - Eclipse PDT
  - Zend Studio (\$\$\$)
  - Netbeans 6.1 Early Access for PHP
  - E outros mais genéricos como:
    - gEdit
    - Vim e Emacs :-)



# O código PHP pode ser embutido no HTML

---

```
1 <?php
2     include 'classes/User.php';
3     session_start();
4 ?>
5 <html>
6 <head><title>Embedded code</title></head>
7 <body>
8 <?php
9     $user = new User();
10    $user->setLogin($_POST['login']);
11    $user->setPassword($_POST['pass']);
12    $user->authenticate();
13    $_SESSION['user'] = $user;
14    if($user->isAuthenticated()) {
15        echo "Welcome, you are authenticated!";
16    } else {
17        echo "Either your login or password is wrong.";
18        //print login html form...
19    }
20 ?>
21 </body>
22 </html>
```



Mas essa não é uma  
boa prática

---

Para separar a lógica do layout  
normalmente se utiliza o engine  
de templates Smarty



<http://smarty.php.net>



# O código

---

```
1 <?
2   include 'classes/User.php';
3   include 'classes/smarty/Smarty.class.php';
4   session_start();
5
6   $user = new User();
7   $user->setLogin($_POST['login']);
8   $user->setPassword($_POST['pass']);
9   $user->authenticate();
10  $_SESSION['user'] = $user;
11
12  $smarty = new Smarty();
13
14  if($user->isAuthenticated()) {
15      $smarty->assign('user',$user);
16      $smarty->display('restricted_area.tpl');
17  } else {
18      $smarty->assign('error_message',"Either your login or password is
wrong.");
19      $smarty->display('login_form.tpl');
20  }
21 ?>
```



# O layout

---

## restricted\_area.tpl

```
1 <html>
2   <head>
3     <title>Restricted Area</title>
4   </head>
5   <body>
6     Hello {$user->getName()}! Welcome to your restricted area.
7     What do you want to do? ...
8   </body>
9 </html>
10
```



# O layout

---

## login\_form.tpl

```
1 <html>
2 <head>
3     <title>Login Form</title>
4 </head>
5 <body>
6     {if $error_message}
7         <h3>{$error_message}</h3>
8     {/if}
9     <form action='login.php' method='post'>
10     Login <input type='text' name='login'><br/>
11     Password <input type='password' name='password' /><br/>
12     <input type='submit' value='Sign in!' />
13 </form>
14 </body>
15 </html>
```



# Interação com bancos de dados

---

- PHP tem suporte nativo a vários SGBDs
- Também oferece camadas de abstração como o PDO



# Um pouco de SQL

---

- Structured Query Language, ou Linguagem de Consulta Estruturada ou SQL, é uma linguagem de pesquisa declarativa para banco de dados relacional (base de dados relacional).
- A linguagem SQL é um grande padrão de banco de dados. Isto decorre da sua simplicidade e facilidade de uso.
- Existem basicamente quatro comandos principais em SQL
  - INSERT
  - SELECT
  - DELETE
  - UPDATE



# INSERT

---

- O comando Insert serve para inserir dados em uma tabela no banco

```
INSERT INTO usuarios (nome,login,senha) VALUES  
( 'Felipe', 'felipernb', '123mudar' );
```



# SELECT

---

- Select permite que você faça consultas ao banco

```
SELECT * FROM usuarios;
```

```
SELECT nome FROM usuarios WHERE login='felipernb';
```



# DELETE

---

- O Delete remove registros do banco

```
DELETE FROM usuarios WHERE login='felipernb';
```



# UPDATE

---

- Update permite alterar registros já existentes

```
UPDATE usuarios SET senha='nova_senha' WHERE  
login='felipernb';
```



# Interação com MySQL

---

```
1 <?php
2     $host = 'localhost';
3     $login = 'meu_login';
4     $senha = 'minha_senha';
5     $database = 'meu_banco';
6     mysql_connect($host,$login,$senha);
7     mysql_select_db($database);
8     mysql_query('INSERT INTO noticias (titulo,texto)
9         VALUES ("Meu Título","Meu Texto")');
10 ?>
```



# PDO

---

- The PHP Data Objects (PDO) extension defines a lightweight, consistent interface for accessing databases in PHP. Each database driver that implements the PDO interface can expose database-specific features as regular extension functions. Note that you cannot perform any database functions using the PDO extension by itself; you must use a database-specific PDO driver to access a database server.



# PDO

---

```
1 <?php
2 $pdo = new PDO("mysql:host=localhost;dbname=my_database",
3     "user", "password", array(PDO::ATTR_PERSISTENT=>true));
4 $pdo->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
5
6 $statement = $pdo->prepare("SELECT * FROM users WHERE `login`= ? ");
7 $params = array($_POST['login']);
8 $statement->execute($params);
9 $statement->setFetchMode(PDO::FETCH_CLASS, "User");
10 $user = $statement->fetch();
11 ?>
```



# PHP trata Web Services com simplicidade

---

- SOAP

```
<?php
    $soapClient = new SoapClient("url_to_wsdl");
    $params = array('xpto1', 'xpto2');
    $soapClient->__soapCall("getContact", $params);
    //Ou pode chamar diretamente:

    $soapClient->getContact($params);

?>
```



# PHP trata Web Services com simplicidade

---

- REST
- <http://www.example.com/rss>

```
1 <?xml version="1.0" encoding="UTF-8" ?>
2 <rss>
3     <channel>
4         <item>
5             <title><![CDATA[Tom & Jerry]]></title>
6         </item>
7         <item>
8             <title><![CDATA[Flintstones]]></title>
9         </item>
10
11     </channel>
12 </rss>
```



# PHP trata Web Services com simplicidade

---

- REST
- rss\_consumer.php

```
1 <?php
2 $xml = simplexml_load_file("http://www.example.com/rss");
3 foreach($xml->channel->item as $item) {
4     echo $item->title . "<br/>";
5 }
6 ?>
7
```



# PHP trata Web Services com simplicidade

---

- REST
- Output

Tom & Jerry  
Flintstones



Vamos desenvolver uma aplicação de exemplo!