



Zend  
The *php* Company



WAMP - LAMP

# Apostila de PHP do Maromo

[professormoraes@gmail.com](mailto:professormoraes@gmail.com)

# ÍNDICE

---

1.	Conceitos Básicos de PHP	4
1.1.	O Que é PHP?	4
1.2.	Histórico	4
1.3.	Instalando o PHP	5
1.3.1.	Instalando o PHP no Linux com Apache	5
1.3.2.	Instalando o PHP no Windows	11
2.	Testando o PHP	15
2.1.	Entendendo o código	15
2.1.1.	Separador de instruções	16
2.1.2.	Comentários uma explicação:	18
3.	Utilizando um Editor PHP para edição de códigos na linguagem (Windows)	20
3.1.	Conhecendo o ambiente do PHP Editor:	21
3.2.	Configurando o PHP Editor	23
3.3.	Algo Útil	25
4.	Conhecendo melhor os Tipos	29
4.1.	Booleanos	30
4.2.	Inteiros	31
4.3.	Números de ponto flutuante	32
4.4.	Strings	33
4.4.1.	Apóstrofes	33
4.4.2.	Aspas	34
4.4.3.	Heredoc	34
4.4.4.	Interpretação de variáveis	35
4.5.	Arrays	37
5.	Manipulando Arquivos	43
5.1.	Arquivos	43
5.2.	Abrindo um arquivo	43
5.3.	Lendo um Arquivo	44
5.4.	Verificando se um arquivo existe:	45
5.5.	Montando uma pequena página de recados:	48
6.	Gerenciamento de Banco de Dados com MySQL	53
6.1.	Linguagem SQL	53
6.2.	MySQL	53
6.2.1.	História	53
6.2.2.	Características	54
6.2.3.	Vantagens	54
6.2.4.	Notas	54
6.3.	A Linguagem SQL e seus componentes:	54
6.3.1.	DML - Linguagem de Manipulação de Dados	55
6.3.2.	DDL - Linguagem de Definição de Dados	55
6.3.3.	DCL - Linguagem de Controle de Dados	56
6.3.4.	DQL - Linguagem de Consulta de Dados	56
7.	Utilizando o PHPmyadmin	57
7.1.	phpMyAdmin	57
7.2.	Tipos de dados mais comuns do MySQL	57
7.3.	Explicando chave primária	61
7.4.	Manipulando dados nas tabelas	61
7.4.1.	Digitando as disciplinas	62

7.4.2.	Digitando os cursos	63
7.4.3.	Digitando os alunos	63
7.5.	Visualizando dados – Instrução SELECT	64
7.5.1.	Exemplo: Consulta para pesquisar todos os alunos da cidade de Campinas.	65
7.5.2.	Exemplo: Consulta para pesquisar todos os alunos ordenados por ordem de curso.	65
7.5.3.	Outros exemplos de consulta	65
7.5.4.	Visualizando dados de mais de uma tabela	66
7.6.	Alterando dados	67
8.	Fazendo o PHP se comunicar com o MySQL	68
8.1.	Páginas HTML do Sistema de Cadastros	68
8.2.	Scripts PHP de configurações	74
8.3.	Scripts de envio de Dados dos Cadastros	75
8.4.	Páginas HTML do Sistema de Pesquisas	76
8.5.	Script PHP do Sistema de Pesquisa	78
9.	Referências Bibliográficas	85

# 1. Conceitos Básicos de PHP

## 1.1. O Que é PHP?

PHP (um acrónimo recursivo para "PHP: Hypertext Preprocessor") é uma linguagem de programação de computadores interpretada, livre e muito utilizada para gerar conteúdo dinâmico na Web. Apesar de ser uma linguagem de fácil aprendizado e de uso para pequenos scripts dinâmicos simples, o PHP é uma linguagem poderosa orientada a objetos,



## 1.2. Histórico

A linguagem surgiu por volta de 1994, como um subconjunto de scripts Perl criados por Rasmus Lerdorf. Com as adições de Zeev Suraski e Andi Gutmans, dois programadores israelitas pertencentes ao Technion, o Instituto Israelita de Tecnologia, que reescreveram o parser, era lançada em 1997 a PHP 3, primeira versão estável e parecida com a linguagem atual. Ao reescrever o parser, foi criado o Zend Engine, que é mantido oficialmente pela empresa Zend em conjunto com a comunidade PHP. Em Maio de 2000 veio a público a versão 4, e em Julho de 2004, a versão 5, onde a principal mudança foi uma nova API para orientação a objetos provida pelo Zend Engine 2.



Trata-se de uma linguagem extremamente modularizada, o que a torna ideal para instalação e uso em servidores web. Diversos módulos são criados no repositório de extensões PECL (PHP Extension Community Library) e alguns destes módulos são introduzidos como padrão em novas versões da linguagem. É muito parecida, em tipos de dados, sintaxe e mesmo funções, com a linguagem C e com a C++. Pode ser, dependendo da configuração do servidor, embutida no código HTML. Além disso, destaca-se a extrema facilidade com que PHP lida com servidores de base de dados, como MySQL, PostgreSQL, Microsoft SQL Server e Oracle.

Existem versões do PHP disponíveis para os seguintes sistemas operacionais: Windows, Linux, Mac OS, OS/2, AS/400, Novell Netware, RISC OS, IRIX e Solaris. A Wikipédia funciona sobre um software inteiramente escrito em PHP, usando bases de dados MySQL: o MediaWiki.

Construir uma página dinâmica baseada em bases de dados é simples com PHP, este provê suporte a um grande número de bases de dados: Oracle, Sybase, PostgreSQL, InterBase, MySQL, SQLite, MSSQL etc, podendo abstrair o banco com a biblioteca ADOdb, entre outras.

PHP tem suporte aos protocolos: IMAP, SNMP, NNTP, POP3, HTTP, LDAP, XML-RPC, SOAP. É possível abrir sockets e interagir com outros protocolos. E as bibliotecas de terceiros expandem ainda mais estas funcionalidades.

Existem iniciativas para utilizar o PHP como linguagem de programação de sistemas fixos. A mais notável é a PHP-GTK. Define-se como um conjunto do PHP com a biblioteca, portada do C++, GTK, fazendo assim softwares inter-operacionais entre Windows e Linux.

**Fonte: wikipedia.org**

Uma das grandes vantagens do PHP é que ele é distribuído gratuitamente através do site <http://www.php.net>. No site oficial do PHP você encontrará sempre as versões mais recentes para downloads. Além de gratuito, o seu código-fonte é aberto, e por fim você encontra toda a documentação do software também no site.

O PHP é baseado no servidor, ou seja, quando você executa uma página PHP no seu navegador, todo o código PHP é executado no servidor de origem da página, ou seja, o navegador apenas exibe a página processada.

## **1.3. Instalando o PHP**

Nesse curso iremos aprender a instalar o PHP nos Sistemas Operacionais Linux e Windows. Nós vamos utilizar a plataforma Windows, acreditando que a maioria das pessoas possuam esse sistema no Desktop, porém é importante saber que todos os códigos escritos em PHP podem ser testado no Linux desde que o ambiente esteja preparado para isso.

### **1.3.1. Instalando o PHP no Linux com Apache**

**Artigo do prof. Morimoto disponível no <http://www.guiadohardware.net/tutoriais/113/>**

Atualmente quase 70% dos servidores web do mundo rodam o Apache, a maior parte deles sobre o Linux. O Apache é um dos servidores web mais antigos, seguro e com inúmeros módulos que adicionam suporte aos mais exóticos recursos.

Ao longo de sua história, o Apache vem sucessivamente derrotando todos os servidores web proprietários. O próprio IIS da Microsoft, que alguns anos atrás parecia um concorrente fortíssimo, hoje em dia é usado em pouco mais de 10% dos servidores.

A maioria das páginas atuais utiliza uma estrutura em PHP, freqüentemente com um banco de dados MySQL ou PostgreSQL. Existem inclusive muitos sistemas prontos, como o phpBB (fórum) e o PHP Nuke (e derivados) para gerenciamento de conteúdo, que podem ser instalados sem muita dificuldade depois que o servidor web já estiver rodando.

Outro recurso muito usado é a encriptação de páginas em SSL, necessário para a criação de páginas seguras (usadas em lojas virtuais, por exemplo) e um sistema de estatísticas de acesso como o Webalizer.

Por padrão, o Apache utiliza a pasta "/var/www" para armazenar os arquivos do site, mas isto pode ser mudado no arquivo de configuração, que vai na pasta "/etc/apache".

A primeira escolha é entre instalar o Apache 2, ou o Apache 1.3, que ainda é usado por muita gente. O Apache 2 traz muitas vantagens, sobretudo do ponto de vista do desempenho, mas por outro lado ele é incompatível com os módulos compilados para o Apache 1.3 e muitas opções de configuração são diferentes. A questão dos módulos não chega a ser um grande problema hoje em dia, pois todos os principais módulos já foram portados, mas muita gente que aprendeu a configurar o Apache 1.3

se sente mais confortável com ele e por isso continua usando-o até hoje, apesar das vantagens da nova versão.

Muitas distribuições continuam oferecendo as duas versões, de forma a satisfazer os dois públicos. No Debian, por exemplo, o Apache 1.3 é instalado através do pacote "apache", enquanto o Apache 2 é instalado através do "apache2".

Junto com o Apache propriamente dito, é interessante instalar também o pacote "**apache-ssl**", que adiciona suporte a páginas seguras (https).

```
# apt-get install apache
# apt-get install apache-ssl
```

Ao instalar o Apache 2, o suporte a SSL é instalado automaticamente junto com o pacote principal:

```
# apt-get install apache2
```

Depois de instalar os pacotes, inicie o serviço com o comando **/etc/init.d/apache start**.

Ao utilizar o Apache 2 o comando fica: **/etc/init.d/apache2 restart**

Em muitas distribuições o serviço do Apache chama-se "httpd" e não "apache" como no Debian. Este é um pedido da própria Apache Foundation, que além do servidor web desenvolve outros projetos, também distribuídos sob a marca "Apache". Neste caso o comando fica "**/etc/init.d/httpd restart**".

Acessando o endereço "**http://127.0.0.1**", você verá uma página de boas-vindas, que indica que o servidor está funcionando. Se não houver nenhum firewall no caminho, ele já estará acessível a partir de outros micros da rede local ou da internet. Por enquanto temos apenas uma versão básica do apache, que simplesmente exibe arquivos html colocados dentro da pasta **"/var/www"**, que por padrão fica sendo o diretório raiz do seu servidor web. A página "**http://seu.servidor/index.html**" é na verdade o arquivo **"/var/www/index.html"**.

A maior parte da configuração do Apache 1.3 pode ser feita através de um único arquivo, o **httpd.conf**, que no Debian pode ser encontrado no diretório **/etc/apache/**. Ao utilizar o Apache 2, o arquivo passa a ser o **"/etc/apache2/apache2.conf"**.

Observe que, assim como todos os arquivos de configuração, você precisa editá-lo como root. Para isso abra um terminal e rode o comando **"su"**, forneça a senha de root e depois abra o arquivo com o comando:

```
# kedit /etc/apache/httpd.conf
```

A primeira configuração importante é a (ou as) portas TCP que serão usadas pelo servidor. Por default, a porta é a 80, mas alguns serviços de banda larga, como por exemplo o Speedy da Telefonica bloqueiam esta porta, obrigando os usuários a manter seus servidores em portas alternativas. Você pode também alterar a porta para manter o seu servidor um pouco mais secreto, principalmente se for utilizada uma porta acima de 1024, já que, além do endereço IP ou domínio, os visitantes precisariam saber também a porta do servidor.

A configuração da porta está perto do final do arquivo, na linha:

**Port 80**

(use o localizar do editor de textos para encontrá-la mais facilmente).

Veja que por default o Apache escuta a porta 80. Basta alterar o 80 pela porta desejada e salvar o arquivo. Para que a alteração entre em vigor, é preciso reiniciar o apache com o comando **"/etc/init.d/apache restart"** ou, **"service httpd restart"**.

Lembre-se de que ao alterar a porta os visitantes precisarão incluir o novo número no endereço. Se você for utilizar a porta 1080, por exemplo, todos deverão acessar o endereço **"http://seu.dominio.com:1080"**.

Você pode também fazer com que o servidor escute em mais de uma porta simultaneamente, usando o recurso **Binding**. Para isso, basta incluir o parâmetro **"Listen porta"** logo abaixo da linha **"Port 80"** que configuramos acima. Para que ele escute também nas portas 1080 e 2480, por exemplo, você incluiria as linhas:

```
Port 80
Listen 1080
Listen 2480
```

Caso o servidor tenha mais de uma placa de rede, você pode utilizar o parâmetro **"Listen IP\_da\_placa:porta"**. Se, por exemplo, estão instaladas duas placas de rede, uma com o endereço 222.132.65.143 e a segunda no endereço 192.168.0.1 e você quer que ele escute em ambas, nas portas 1080 e 2480, bastaria incluir:

```
Listen 222.132.65.143 :1080
Listen 222.132.65.143 :2480
Listen 192.168.0.1 :1080
Listen 192.168.0.1 :2480
```

Não existe limitação para o uso deste recurso. Você pode fazer o servidor escutar quantas portas e placas de rede forem necessárias. Ao utilizar o Apache 2 no Debian, a configuração de portas fica separada, dentro do arquivo **"/etc/apache2/ports.conf"**.

Outro recurso suportado pelo Apache e muito usado, é a possibilidade de hospedar vários sites no mesmo servidor (shared hosting). Mais de 70% dos sites da internet são hospedados desta forma econômica.

Neste caso, os arquivos de cada site ficam guardados numa pasta diferente e o servidor se encarrega de direcionar cada visitante ao site correto. Servidores como os dos serviços de hospedagem gratuita chegam a hospedar mais de 10.000 sites num único servidor Apache usando este recurso.

Existem duas formas de fazer isso. A primeira é ter um servidor com vários endereços IP e vincular cada site a um endereço (IP-Based). Este sistema é pouco usado, pois atualmente os endereços IP válidos são um recurso escasso e valioso. A segunda forma (de longe a mais usada) é ter um único endereço IP válido e vincular cada site a um nome de domínio (Name-Based).

Vamos ver primeiro a opção com múltiplos endereços IP que é a mais simples, e em seguida a com vários nomes:

**IP-Based:** Esta opção é útil caso você tenha mais de um link no mesmo servidor. Você pode usar um único servidor para duas linhas ADSL, ou duas linhas T1, por exemplo, ou pode ainda ter uma única placa de rede configurada para receber conexões em vários endereços IP, usando aliases.

Para criar aliases para sua placa de rede, basta usar o **ifconfig**, informando a placa de rede que receberá o alias (eth0, eth1, etc.) e o endereço IP em que ela passará a escutar. O alias é apenas um apelido; ele não altera a configuração original da placa de rede, apenas faz com que ela passe a se comportar como se fosse várias placas, escutando em vários endereços diferentes. É sem dúvida um recurso muito interessante.

Se você deseja que a sua interface eth0 passe a escutar também nos endereços 220.177.156.2, 220.177.156.3 e 220.177.156.4, os comandos seriam:

```
# ifconfig eth0:0 220.177.156.2
# ifconfig eth0:1 220.177.156.3
# ifconfig eth0:2 220.177.156.4
```

Um detalhe importante é que os aliases são desativados sempre que o servidor é reiniciado. Para que a alteração seja permanente, é necessário adicionar os comandos no arquivo **"/etc/init.d/bootmisc.sh"** ou **"/etc/rc.d/rc.local"** para que eles sejam executados a cada boot.

No Apache, basta criar seções no arquivo **"httpd.conf"**, indicando as configurações de cada site, como por exemplo:

```
<VirtualHost 220.177.156.2>
ServerAdmin roberto@usuario.com
DocumentRoot /var/www/roberto/www
ServerName www.roberto.com.br
ErrorLog /sites/roberto/logs/error_log
TransferLog /sites/roberto/logs/access_log
</VirtualHost>

<VirtualHost 220.177.156.3>
ServerAdmin maria@usuario.com
DocumentRoot /var/www/maria/www
ServerName www.maria.com.br
ErrorLog /sites/maria/logs/error_log
TransferLog /sites/maria/logs/access_log
</VirtualHost>
```

Criamos aqui a configuração para dois sites distintos, um no endereço 220.177.156.2 e o outro no 220.177.156.3. Tanto faz se cada endereço corresponde a uma placa de rede separada ou se são aliases para uma única placa. O que interessa é que sempre que alguém digitar o endereço IP ou o domínio correspondente no browser será capaz de acessar o site. O IP de cada site é especificado na primeira linha, opção **"VirtualHost"**.

A próxima linha **"ServerAdmin"** permite especificar o e-mail do administrador, para onde serão enviadas mensagens de erro e avisos de anormalidades no servidor.

A opção **"DocumentRoot"** é outra configuração crucial, simplesmente porque diz em que pastas ficarão armazenados os arquivos do site em questão. Naturalmente cada site deve ter sua própria pasta, que deve ser acessível ao cliente via ftp, ssh ou outra forma qualquer, para que ele possa dar upload dos arquivos do site.

Isto significa que, além de configurar o Apache, você deve criar para ele um usuário no sistema e configurar um servidor de FTP ou SSH. Para finalizar, use o comando **"chown -R usuário pasta"** para transformar o usuário em dono da pasta e o comando **"chmod 755 pasta"** para acertar as permissões de acesso. Isto faz com que o dono tenha controle total e os demais usuários (e visitantes



do site) possam apenas ler os arquivos e executar scripts postos no servidor (como scripts em CGI ou páginas PHP), sem permissão para gravar ou alterar nada.

A opção "**ServerName**" indica o nome de domínio do servidor e não é necessária quando o site é acessado apenas através do endereço IP. Finalmente temos a localização dos dois arquivos de log: ErrorLog e TransferLog. Por padrão, estes arquivos devem ficar dentro da pasta logs, no diretório raiz do site, separados dos arquivos disponibilizados ao público, que ficam na pasta www. Naturalmente você pode usar outras localizações se quiser, é apenas uma convenção.

**Name-Based:** Esta segunda opção é bem mais usada que a IP-Based, por isso deixei por último, caso contrário era capaz de você pular o outro tópico ;-). A configuração baseada em nomes permite que você hospede vários sites, cada um com seu próprio nome de domínio num servidor com um único link e um único IP.

A configuração no arquivo httpd.conf é até mais simples que a baseada em IP. A seção fica:

```
NameVirtualHost 192.168.1.100

<VirtualHost kurumin>
ServerName www.kurumin.com.br
DocumentRoot /var/www/kurumin
</VirtualHost>

<VirtualHost kacique.com.br>
ServerName www.kacique.com.br
DocumentRoot /var/www/kacique
</VirtualHost>

<VirtualHost morimoto.com.br>
ServerName www.morimoto.com.br
DocumentRoot /var/www/morimoto
</VirtualHost>
```

A primeira linha ("**NameVirtualHost \***") especifica o endereço IP e a porta do servidor principal. Aqui estou usando como exemplo o endereço "192.168.1.100", pois estou configurando um servidor que vai ficar disponível apenas dentro da rede local. Ao configurar um servidor público, você usaria o endereço IP do servidor na internet, como por exemplo "215.23.45.143".

Em seguida temos as seções **VirtualHost**, que especificam o nome de domínio e o diretório local onde ficam os arquivos de cada um. A idéia aqui é que o visitante digita o nome de domínio do site no navegador e o Apache se encarrega de enviá-lo ao diretório correto. Mas, para que o cliente chegue até o servidor, faltam mais duas peças importantes.

A primeira é o registro do **domínio**, que pode ser feito na Fapesp, Internic ou outro órgão responsável. No registro do domínio, você deverá fornecer dois endereços de DNS (primário e secundário). Se você tiver apenas um, existe um pequeno truque: conecte-se via modem na hora de fazer o registro, assim você terá dois endereços (o do link e o do modem) e conseguirá fazer o registro. Naturalmente, neste caso, você perde a redundância; se o seu link principal cair seu site ficará fora do ar.

É aqui que acaba o trabalho deles e começa o seu. Ao acessar o domínio, o visitante é direcionado para o endereço de DNS fornecido no registro. Isto significa que... bingo! além do Apache você vai precisar de um servidor de DNS :-)

O DNS não precisa necessariamente ser uma máquina separada. Mais adiante veremos como instalar e configurar o Bind para esta função. Neste caso a requisição do cliente é direcionada da Fapesp para o seu servidor DNS e dele para o servidor Apache. O ciclo se fecha e o cliente consegue finalmente acessar a página.

Se seu servidor estiver hospedando subdomínios, ou seja, endereços como "www.fulano.guiadohardware.net", "www.ciclano.guiadohardware.net", etc., como fazem serviços como o hpg, a configuração continua basicamente a mesma. Você especifica o sub-domínio do cliente na configuração do VirtualHost do Apache e também no servidor de DNS.

Uma observação importante é que para o Apache, o domínio "www.fulano.guiadohardware.net" é diferente de apenas "fulano.guiadohardware.net". Para que o site possa ser acessado tanto com o www quanto sem ele, é necessário cadastrar os dois endereços, usando um alias para que o segundo leve ao primeiro.

Como no caso anterior, você deve informar o endereço do seu servidor de DNS no registro do domínio. Como os servidores de registro de domínio lêem as URLs de trás para a frente, todos os acessos a subdomínios dentro do guiadohardware.net serão enviados para o seu servidor DNS e daí para o servidor Apache.

Ao usar o **Apache 2**, a configuração dos hosts virtuais fica ligeiramente diferente. Ao invés de colocar as configurações de todos os sites diretamente no httpd.conf, a configuração é quebrada em vários arquivos individuais (um para cada site) armazenados dentro da pasta **"`/etc/apache2/sites-available`"**.

Dentro da pasta você encontrará o arquivo "default" que contém uma configuração padrão da página "default", que é exibido quando o usuário não especifica um domínio válido no servidor. Para adicionar outros sites, basta criar um arquivo para cada um (como por exemplo "kurumin.com.br"), contendo o seguinte:

```
<VirtualHost *>
ServerName www.kurumin.com.br
ServerAlias kurumin.com.br
DocumentRoot /var/www/html/kurumin/www/
</VirtualHost>
```

Note que adicionei uma nova diretiva, a "ServerAlias", que permite que o site seja acessado tanto com, quanto sem o "www". O DocumentRoot também foi alterado, com relação aos exemplos anterior para refletir a organização padrão adotada no Apache 2, onde os arquivos são por padrão armazenados dentro da pasta /var/www/html, ao invés de simplesmente /var/www.

Para adicionar mais sites, basta criar vários arquivos, um para cada um, não esquecendo de criar as pastas default de cada.

A pasta **"`/etc/apache2/sites-available`"** contém todos os sites disponíveis no servidor. Para que os sites fiquem realmente disponíveis, é necessário criar um link simbólico para cada um, dentro da pasta **"`/etc/apache2/sites-enabled`"**. São estes links que determinam se o site vai realmente ficar disponível. Esta configuração foi adotada para facilitar a vida de quem administra servidores compartilhados por muitos sites. Para desativar um site por falta de pagamento, por exemplo, basta remover o link e recriá-lo quando a situação for normalizada.

Para criar o link para o site "kurumin.com.br" que criamos no exemplo anterior, o comando seria:

```
# cd /etc/apache2/sites-enabled
# ln -s ../sites-available/kurumin.com.br kurumin.com.br
```

Não se esqueça de reiniciar o Apache no final do processo para que as mudanças entrem em vigor.

Esta configuração manual funciona para pequenos servidores, que hospedam algumas dezenas ou centenas de páginas. Grandes serviços de hospedagem geralmente acabam desenvolvendo algum tipo de sistema para automatizar a tarefa. Nos serviços de hospedagem gratuita, por exemplo, onde o número de clientes é assustadoramente grande, as alterações são feitas de forma automática quando o visitante faz seu cadastro, geralmente através de um sistema escrito em PHP ou Java.

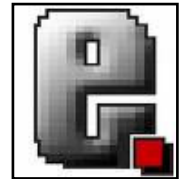
Conforme o número de usuários cresce e o espaço em disco no servidor começa a ficar escasso, você começará a sentir falta de um sistema de quotas que limite o espaço que cada usuário pode usar. Para isso, consulte o tópico sobre quotas de disco mais adiante.

### 1.3.2. Instalando o PHP no Windows



Para que possamos trabalhar com o PHP no Windows, precisamos instalar no computador alguns programas que chamamos de WAMP (Windows + Apache + MySQL + PHP).

Uma maneira simples de instalar esses softwares é através do Easy PHP um programa distribuído sob a licença GPL, que em apenas em poucos



passos instala o servidor Apache, o módulo para programação em PHP e o banco de dados MySQL.

**Local para download na Web: <http://www.easyphp.org/>**

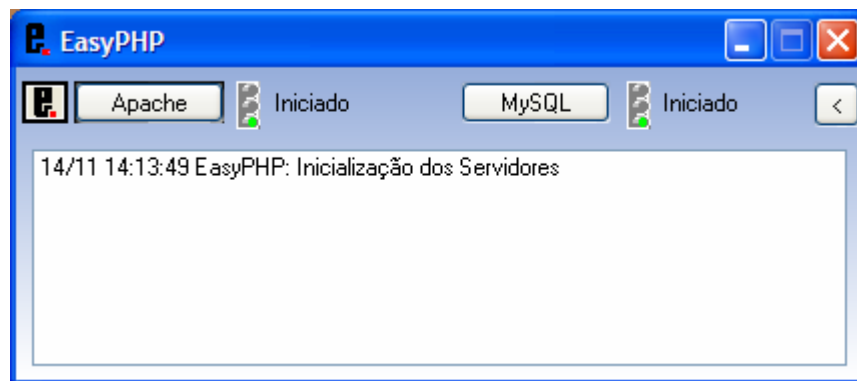
Este é um programa muito interessante para as pessoas que querem no menor tempo possível poder trabalhar com PHP sobre Windows, sem ter que passar pelas dificuldades de instalar e configurar todos os servidores e módulos necessários para trabalhar com esta linguagem de criação de páginas do lado do servidor.

O Easy PHP não é só um programa, são três em um:

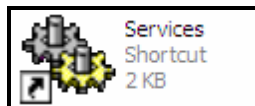
- \_ Apache, o servidor mais popular de páginas web.
- \_ MySQL, o banco dados mais difundido de código livre.
- \_ PHP, a linguagem ou tecnologia mais difundida para realizar páginas com programação no servidor, acesso ao banco de dados, etc.

### Instalação de EasyPHP

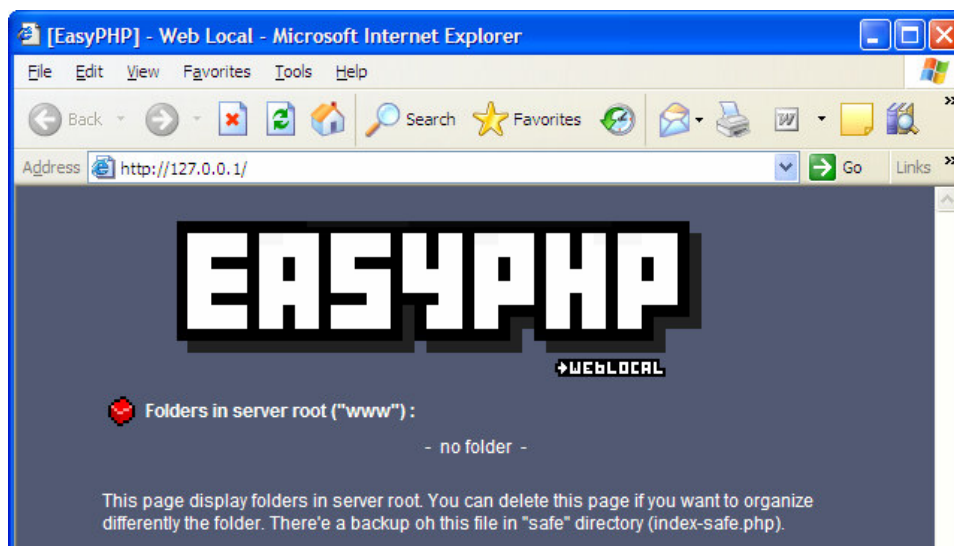
Uma vez baixado, a instalação é imediata. Para iniciar, abriremos Easy PHP, se já não o tivermos aberto. No botão Iniciar - Todos os programas - EasyPHP - EasyPHP. O Resultado será a abertura de uma janela como a figura abaixo se os servidores PHP e MySQL estiverem iniciados corretamente.



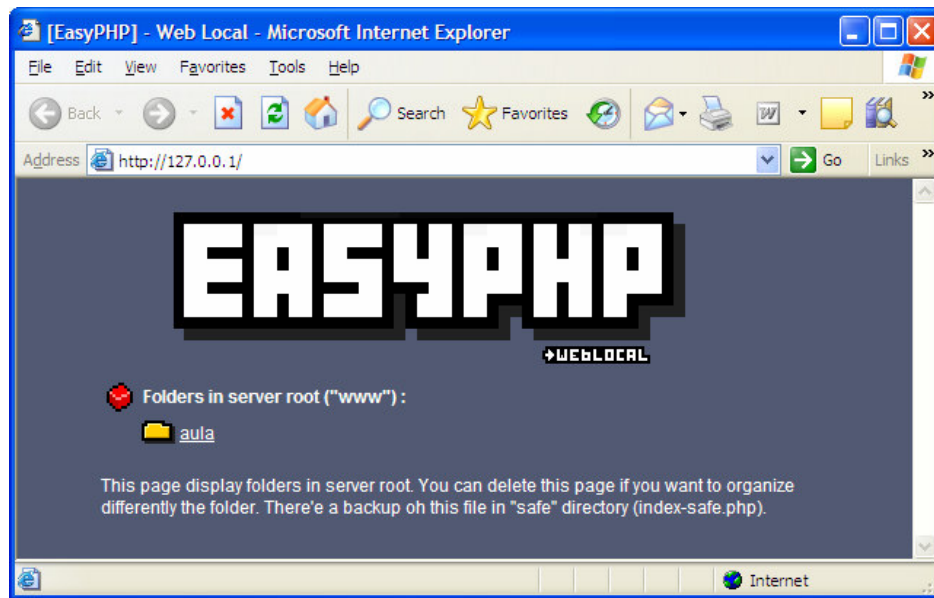
Caso você tenha outro servidor **web** levantado na rede, como o IIS, por exemplo, ou outro servidor de banco de dados, pare os serviços no Pannel de Controle – Ferramentas Administrativas – Serviços.



Clique com o botão direito do mouse no ícone do EasyPHP na barra de tarefas e escolha a opção Localhost, ou digite diretamente no navegador o endereço <http://localhost>, assim se tudo estiver funcionando corretamente deverá aparecer a página de início do EasyPHP.

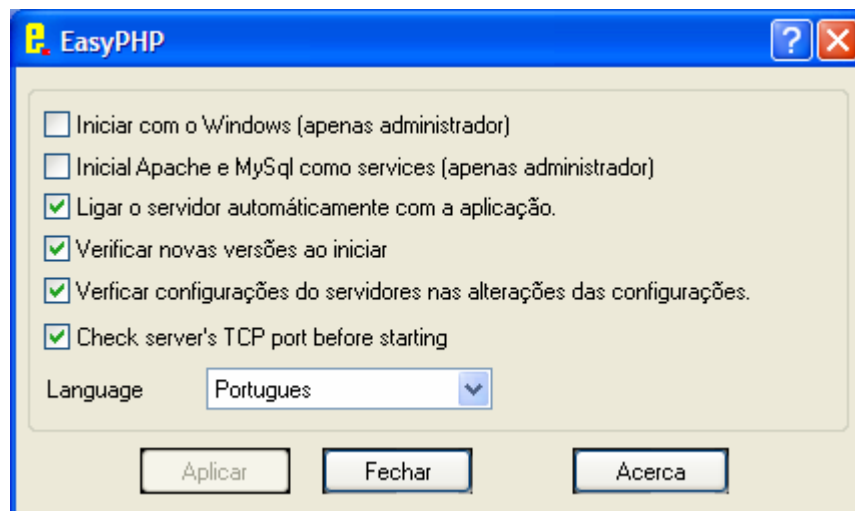


Os documentos web deverão ser armazenados na pasta de publicação de documentos do EasyPHP “C:\Arquivos de programa\EasyPHP1-8\www”. Crie uma pasta chamada aula dentro dessa pasta para adicionarmos nossos primeiros scripts em PHP. Se você visitar a página local agora verificará que a pasta aula aparece na mesma.



## Como configurar o EasyPHP

Na Barra de Tarefas clique no ícone do EasyPHP aponte para configuração – EasyPHP. Assim poderemos alterar qualquer uma das opções abaixo:



- Iniciar com o Windows (apenas administrador): como diz a própria opção, estando essa checada, o EasyPHP será iniciado automaticamente quando o Windows for iniciado.
- Iniciar o Apache e o MySQL como services (apenas administrador): com essa opção marcada os servidores Apache e MySQL são iniciados como serviços mesmo que esteja logado na máquina outro usuário que não seja o administrador.
- Ligar o servidor automaticamente com a aplicação: essa opção indica que quando o EasyPHP for ativado os servidores serão iniciados, essa opção selecionada é o padrão.

- Verificar novas versões ao iniciar: essa opção selecionada, indica que toda a vez que o EasyPHP for iniciado fará uma busca no site do projeto do EasyPHP e colocará uma mensagem informando se houver nova versão disponível.
- Verificar configurações dos servidores nas alterações de configurações: Indica que se as configurações dos servidores sofrerem alterações, os mesmos serão reiniciados.
- Verificar a porta TCP antes de iniciar o servidor: opção marcada por padrão.

Para maiores informações sobre a configuração consulte as FAQs no site do projeto:  
<http://www.easyphp.org/faq.php3>

O programa não termina por aqui, ainda podemos instalar alguns complementos ideais para começar a trabalhar com PHP e banco de dados MySQL, como PhpMyAdmin, um administrador de banco de dados bastante conhecido. Podemos encontrá-lo no endereço <http://localhost/mysql/>

O acesso a PhpMyAdmin fica bloqueado a outros computadores da rede local, visto que se instala como um diretório virtual de Apache com acesso restringido à rede local, mas através das FAQ também aprenderemos a dar acesso a outras máquinas.

**Fonte:** <http://www.criarweb.com/artigos/537.php>

**Adaptado por Maromo**

## 2. Testando o PHP

Para criar e editar scripts em PHP podemos utilizar qualquer editor Html, ou até mesmo o bloco de notas. Vamos inicialmente utilizar o bloco de notas, depois optaremos por outro editor PHP.

### 2.1. Entendendo o código

Um código php pode conter ou não tags Html, essas tags não são processadas pelo servidor, são simplesmente passadas ao browser solicitante. Normalmente utiliza-se Html para fazer a parte estática da página, sua estrutura e o php para a parte lógica, que exige processamento. “Deve-se salvar os códigos em PHP com extensão “.php”.

*Nesse material, estarei considerando que você tenha conhecimentos básicos de HTML, quando necessário explicarei os códigos. ☺*

Há quatro conjuntos de tags que podem ser usadas para marcar blocos de código PHP. Delas, somente duas (<?php. . .?> e <script language="php">. . .</script>) são sempre disponíveis. As outras podem ser ativadas ou desativadas a partir do arquivo de configuração php.ini.

O primeiro método, <?php. . .?>, é o preferencial, já que ele permite o uso do PHP em códigos padrão XML como o XHTML.

Vamos usar no início a sintaxe configurado no arquivo php.ini abaixo:

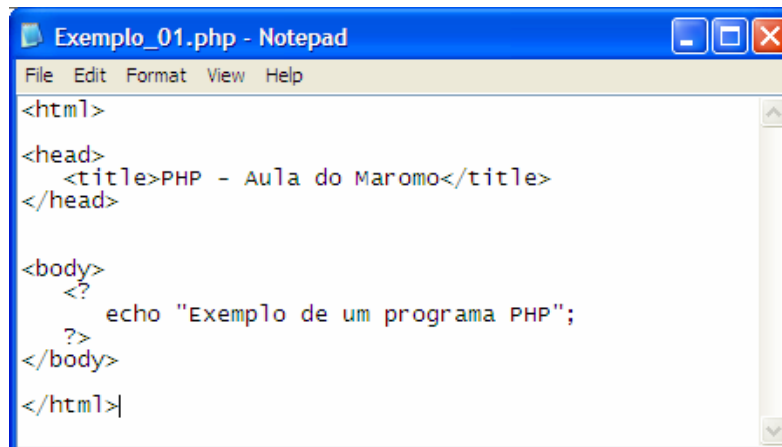
```
<?
  Código php
?>
```

Depois passaremos a utilizar a preferencial:

```
<?php
  Código php
?>
```

#### Primeiro Exemplo em PHP

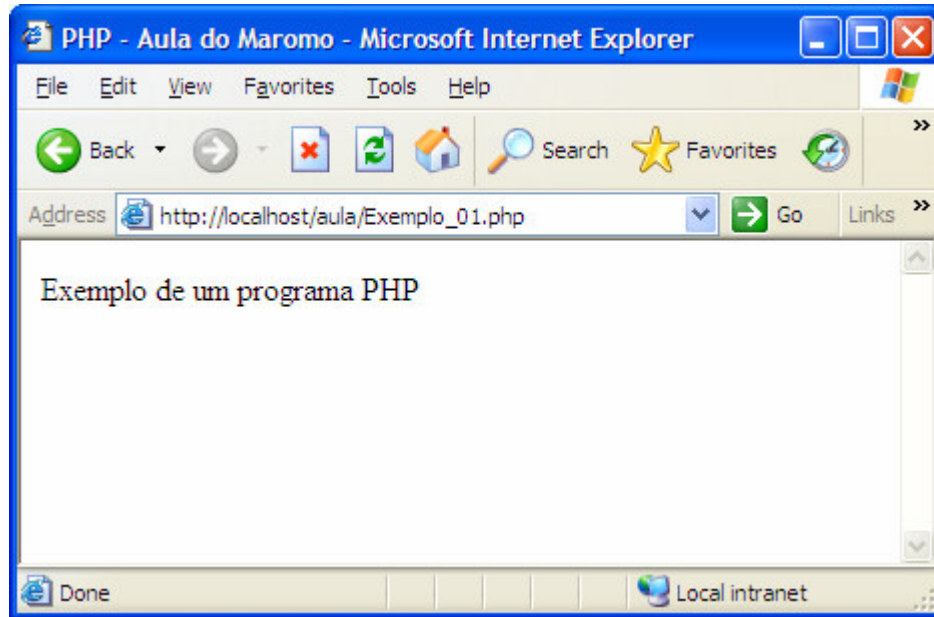
1) Carregue o bloco de Notas e digite o código:



```
Exemplo_01.php - Notepad
File Edit Format View Help
<html>
<head>
  <title>PHP - Aula do Maromo</title>
</head>
<body>
  <?
    echo "Exemplo de um programa PHP";
  ?>
</body>
</html>
```

2) Salve com o nome Exemplo\_01.php na pasta “C:\Arquivos de programa\EasyPHP1-8\www\aula”

3) Para executar, abra o navegador e digite o endereço: [http://localhost/aula/Exemplo\\_01.php](http://localhost/aula/Exemplo_01.php), você terá como resultado a seguinte tela:



O comando **echo** retorna uma string para o resultado em html, podemos passar esta string diretamente (como no exemplo) ou uma variável contendo uma string.

Quando o PHP interpreta um arquivo, ele simplesmente repassa o texto do arquivo até encontrar uma das tags especiais que lhe diz para começar a interpretar o texto como código PHP. O interpretador então executa todo o código que encontra, até chegar em uma tag de fechamento PHP, que novamente o coloca simplesmente repassando texto novamente. Este é o mecanismo que permite a inclusão de código PHP dentro do HTML: qualquer coisa fora das tags PHP é deixado como encontrado, enquanto tudo dentro é interpretado e executado.

### 2.1.1. Separador de instruções

Instruções são separadas da mesma forma que o C ou o Perl - cada instrução termina com um ponto e vírgula.

A tag de fechamento (`?>`) também implica no fim de uma instrução, então os exemplos seguintes são equivalentes:

Ex.:

```
echo “Exemplo de um programa em PHP”;
```

#### Variáveis

As variáveis em PHP não precisam ser declaradas, mas devem sempre ter **\$** na frente do nome. Os nomes devem sempre começar com letras ou com o caractere “**\_**”.

Ex.: \$nome

\$\_cpf



**Observação:** O PHP é case sensitive, portanto \$nome é diferente de \$Nome ou \$NOME. Em uma variável podemos armazenar os mais diversos tipos de dados. Iremos conhecer agora quais os tipos de dados que podem ser utilizados na linguagem PHP.

Abaixo temos uma tabela com os tipos de dados utilizados no PHP.

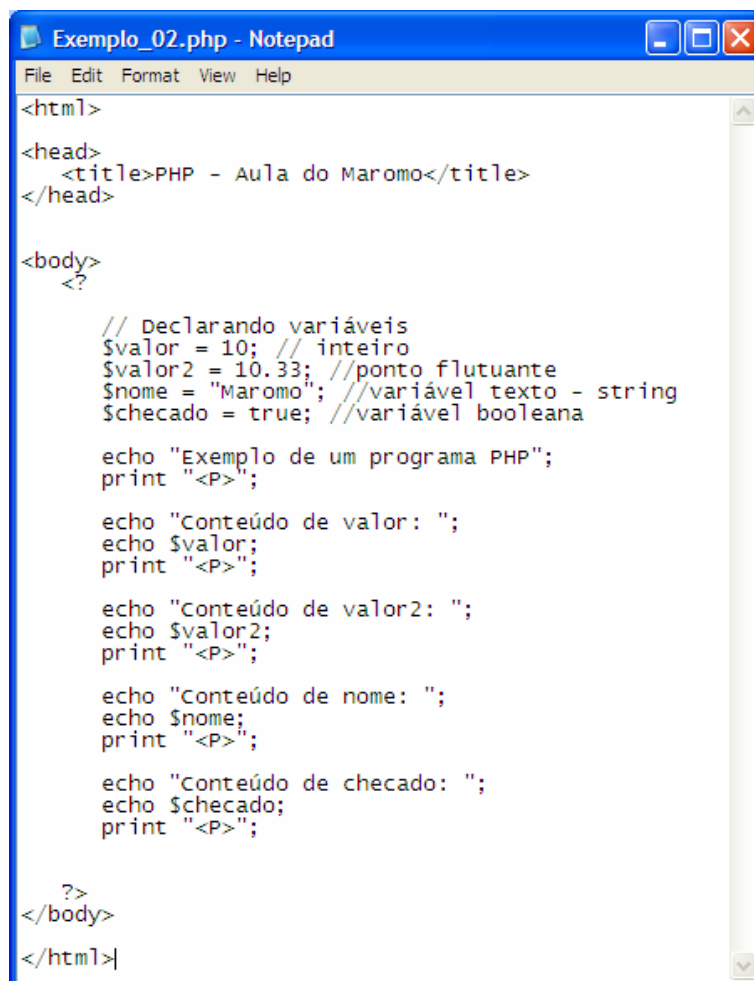
Tipo	Exemplo	Descrição
Inteiro	100	Um número inteiro.
Double	10.25	Um número de ponto flutuante.
String	"Aprendendo PHP"	Uma coleção de caracteres.
Boolean	True	Um dos seguintes valores: true ou false.
Object		Estudaremos este tipo mais à frente.
Array		Estudaremos este tipo mais à frente.

**OBS:** Falaremos com maior detalhes desse assunto mais adiante.

## Segundo Exemplo em PHP

Vamos ver como realizar a declaração de variáveis no PHP, nesse segundo exemplo. Para tanto execute os passos a seguir.

1) Carregue o bloco de Notas e digite o código abaixo:



```
<html>
<head>
  <title>PHP - Aula do Maromo</title>
</head>

<body>
  <?
    // Declarando variáveis
    $valor = 10; // inteiro
    $valor2 = 10.33; //ponto flutuante
    $nome = "Maromo"; //variável texto - string
    $checado = true; //variável booleana

    echo "Exemplo de um programa PHP";
    print "<P>";

    echo "Conteúdo de valor: ";
    echo $valor;
    print "<P>";

    echo "Conteúdo de valor2: ";
    echo $valor2;
    print "<P>";

    echo "Conteúdo de nome: ";
    echo $nome;
    print "<P>";

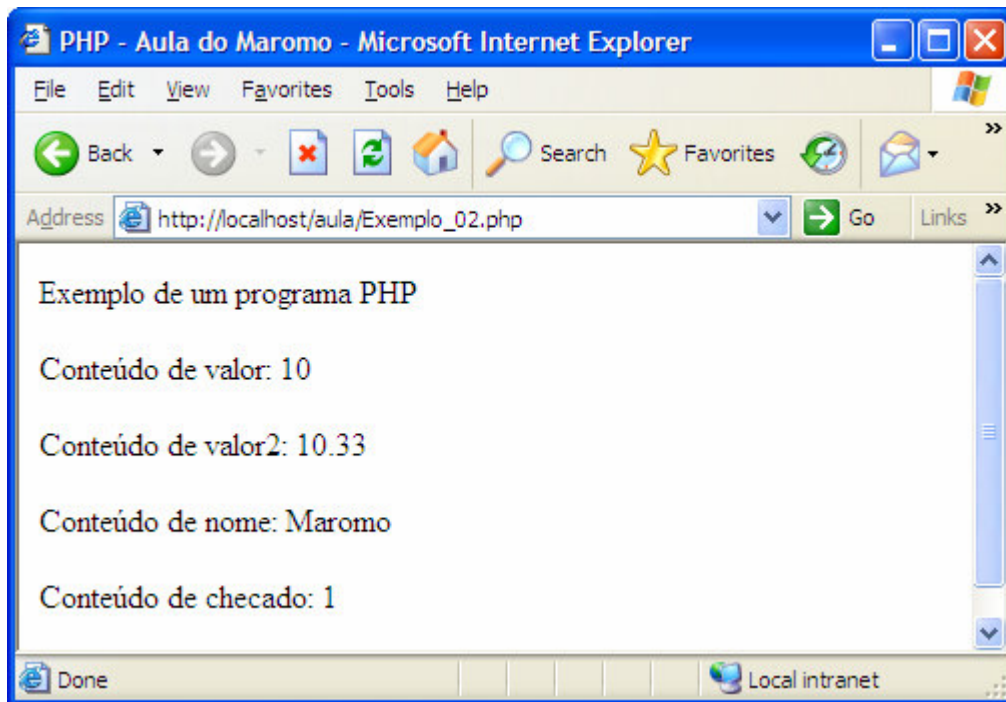
    echo "Conteúdo de checado: ";
    echo $checado;
    print "<P>";

  ?>
</body>
</html>
```

**Observação:** você pode acrescentar comentários para as linhas de código de várias formas, umas delas é a utilização da string “//” antes do comentário. Quando o interpretador PHP encontra essa sequência ele ignora o restante da linha.

2) Salve com o nome Exemplo\_02.php na pasta “C:\Arquivos de programa\EasyPHP1-8\www\aula”

3) Para executar, abra o navegador e digite o endereço: [http://localhost/aula/Exemplo\\_02.php](http://localhost/aula/Exemplo_02.php), você terá como resultado a seguinte tela:



### 2.1.2. Comentários uma explicação:

O PHP suporta comentários do 'C', 'C++' e Unix shell. Por exemplo

```
<?php
echo "Isto é um teste"; //Comentário de uma linha no C++
/* Isto é um comentário de mais de uma linha
   e aqui temos outra linha */
echo "Isto é um outro teste";
echo "O último teste"; #Comentário no estilo Unix shell
?>
```

Os comentários de uma linha só terão efeito até o fim da linha ou fim do bloco de código PHP atual, o que ocorrer primeiro.

```
<h1>Isto é um <?php # echo " simples";?> exemplo.</h1>
<p>No título acima você lerá 'Isto é um exemplo'.
?>
```

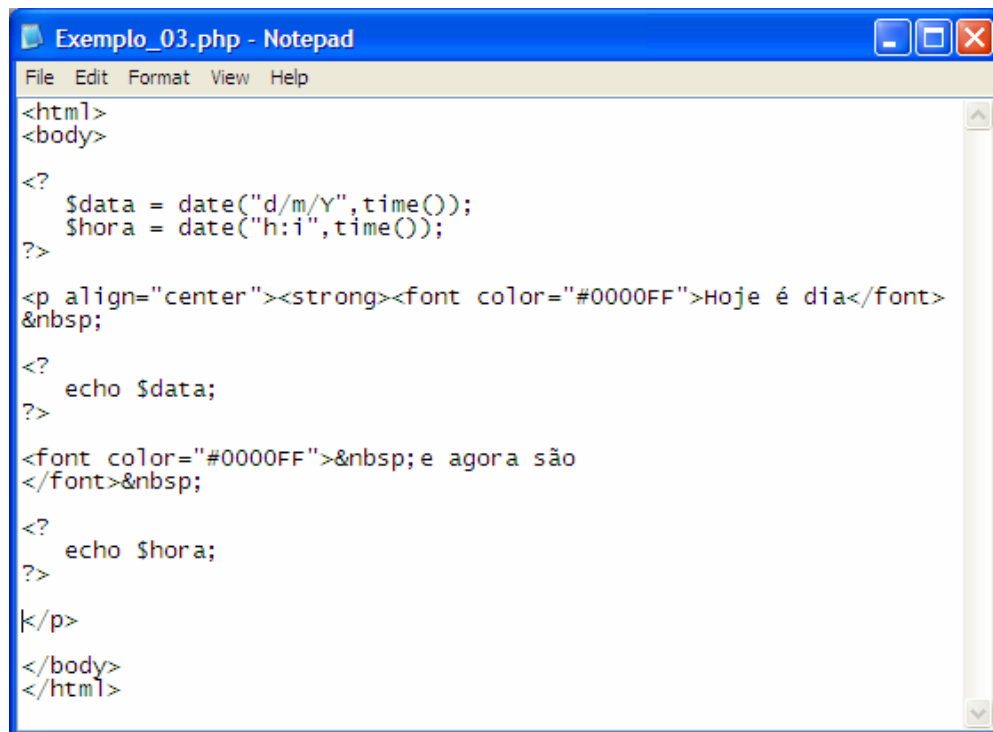
Os comentários de uma linha somente agem até o fim da linha atual ou o fim do bloco de código PHP, o que ocorrer primeiro. Isto significa que código HTML após // ?> SERÁ impresso: ?>

continuará desligando o modo PHP, retornando para o modo HTML, e o // não pode influenciar isso.

### Terceiro Exemplo:

Continuando ainda o assunto sobre tipo de dados, vamos para mais um exemplo, dessa vez utilizaremos duas variáveis que receberão a data e hora atual respectivamente.

1) Carregue novamente o Bloco de Notas e digite o Código abaixo:



```
<html>
<body>

<?
    $data = date("d/m/Y",time());
    $hora = date("h:i",time());
?>

<p align="center"><strong><font color="#0000FF">Hoje é dia</font>
&nbsp;

<?
    echo $data;
?>

<font color="#0000FF">&nbsp;e agora são
</font>&nbsp;

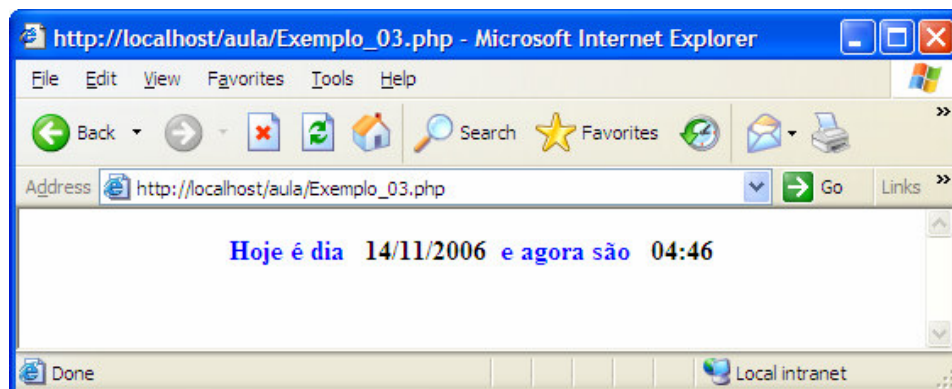
<?
    echo $hora;
?>

</p>

</body>
</html>
```

2) Salve com o nome Exemplo\_03.php na pasta “C:\Arquivos de programa\EasyPHP1-8\www\aula”

3) Para executar, abra o navegador e digite o endereço: [http://localhost/aula/Exemplo\\_03.php](http://localhost/aula/Exemplo_03.php), você terá como resultado a seguinte tela:



No exemplo acima as variáveis \$data e \$hora, receberam o valor da data e hora respectiva do sistema através da função date( ).

### 3. Utilizando um Editor PHP para edição de códigos na linguagem (Windows)

Com o PHP Editor você poderá editar e visualizar facilmente suas páginas PHP ou HTML, de modo mais agradável que o bloco de notas. Possui o recurso de coloração de código fonte, ambiente multi-documentos, bookmarks no texto, auto-complemento de funções e outros recursos. Essa versão está em português. O software pode ser encontrado no endereço:

- <http://baixaki.ig.com.br/site/dwnld5761.htm>

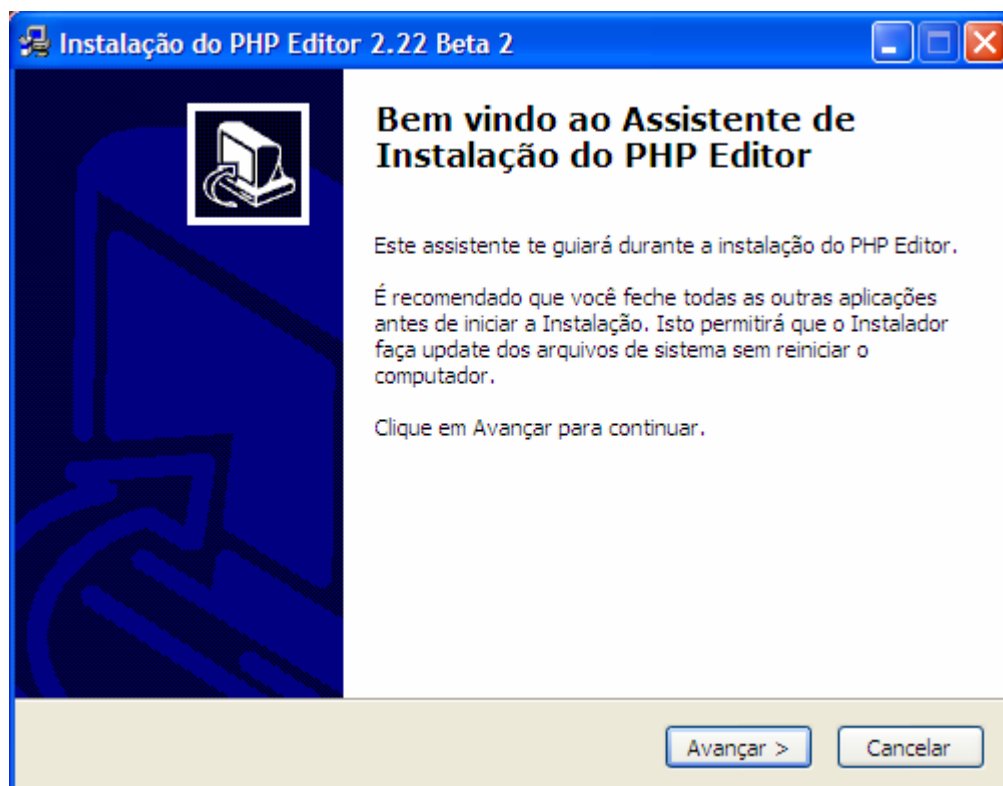
ou no site do projeto:

- <http://www.PHP Editor.kit.net/>

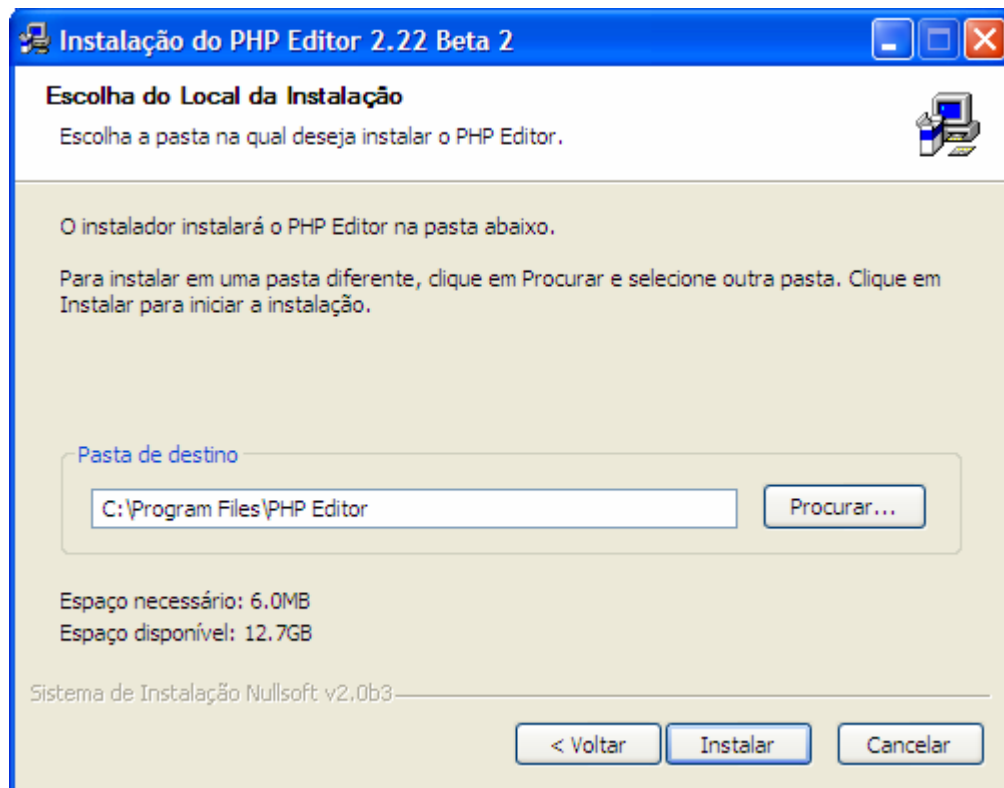
Depois de baixado, execute o arquivo de instalação:



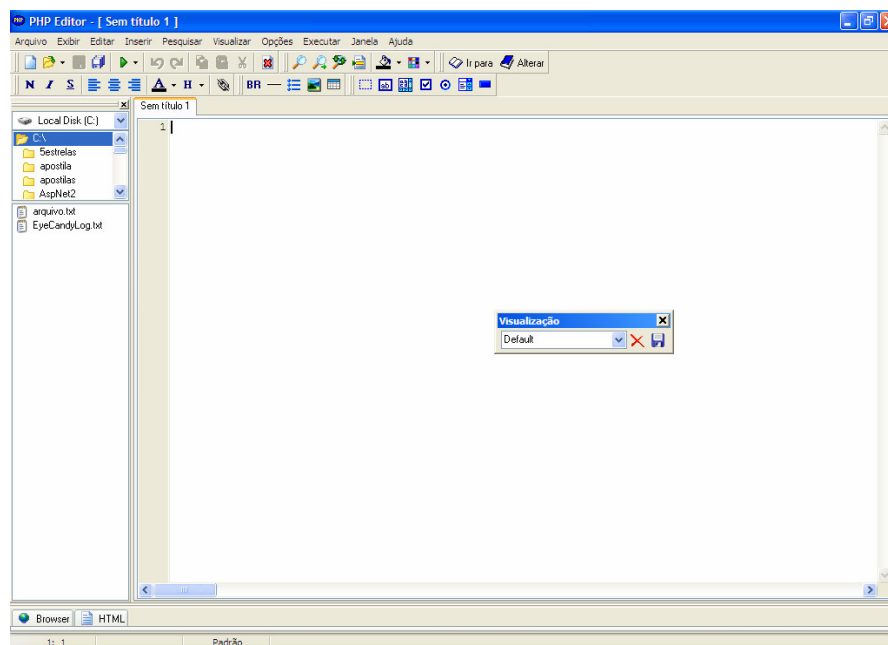
Executando o arquivo do Setup do PHP Editor surgirá a tela do assistente de instalação, conforme figura:



Clique no botão avançar, na próxima tela indique o local de instalação, deixe o padrão mesmo, ou seja, C:\Arquivos de Programas\PHP Editor\. Clique em seguida em Instalar:



Agora é só terminar e executar o ícone do PHP Editor. A tela inicial deve estar parecida com a figura:



### 3.1. Conhecendo o ambiente do PHP Editor:

O PHP Editor possui 08 barra de ferramentas, são elas:

1) Principal.



Com os botões:

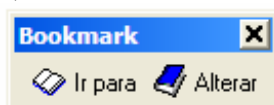
- Novo: para iniciar um novo documento em PHP,
- Abrir: para carregar um documento já existente em PHP,
- Salvar: para gravar o documento atual,
- Salvar Todos: para salvar todos os documentos abertos,
- Visualizar: visualiza o código em uma janela de simulação do navegador (Browser),
- Desfazer: desfaz a última ação de edição,
- Refazer: refaz a última ação de edição,
- Copiar: copia código selecionado,
- Colar: cola o conteúdo da área de transferência para a janela de código,
- Recortar: recorta o código selecionado,
- Fechar: fecha o documento ativo.

2) Ferramentas



- Procurar por algum texto: como o próprio nome diz, serve para procurar texto na janela corrente,
- Procurar próximo: procura a próxima ocorrência da palavra,
- Substituir texto: substituir ocorrência de texto,
- Inserir cor: inserir código de cores no html,
- Coloração: padrões de coloração do editor, (dê preferência ao padrão).

3) Bookmark



- Ir para: ir para bookmark no texto,
- Alterar: alterar bookmark do texto.

4) Fonte



- Negrito: texto em negrito,
- Itálico: texto em itálico,
- Alinhar à esquerda: alinhamento do texto à esquerda,
- Centralizar: centralizar o texto,
- Cor da fonte: cor do texto,
- Estilo: estilo do texto selecionado,
- Link: inserir hiperlink.

## 5) Html



- BR: inserir quebra de linha
- Traço: inserir barra horizontal
- Marcadores: inserir marcadores de parágrafo,
- Imagem: inserir imagem no documento,
- Tabela: inserir tabela no documento.

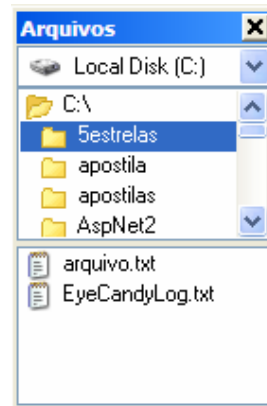
## 6) Formulário



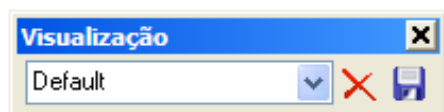
- Form: inserir um formulário no documento,
- Caixa de texto: inserir uma caixa de texto no documento,
- Caixa de texto de rolagem: inserir uma caixa de texto no documento com barra de rolagem,
- Caixa de seleção: inserir uma caixa de seleção no documento,
- Botão de opção: inserir um botão de opção no documento,
- Menu suspenso: menu de opções suspenso no documento,
- Botão: botão de comando.

## 7) Arquivos:

Janela de visualização dos arquivos no computador.



## 8) Visualização

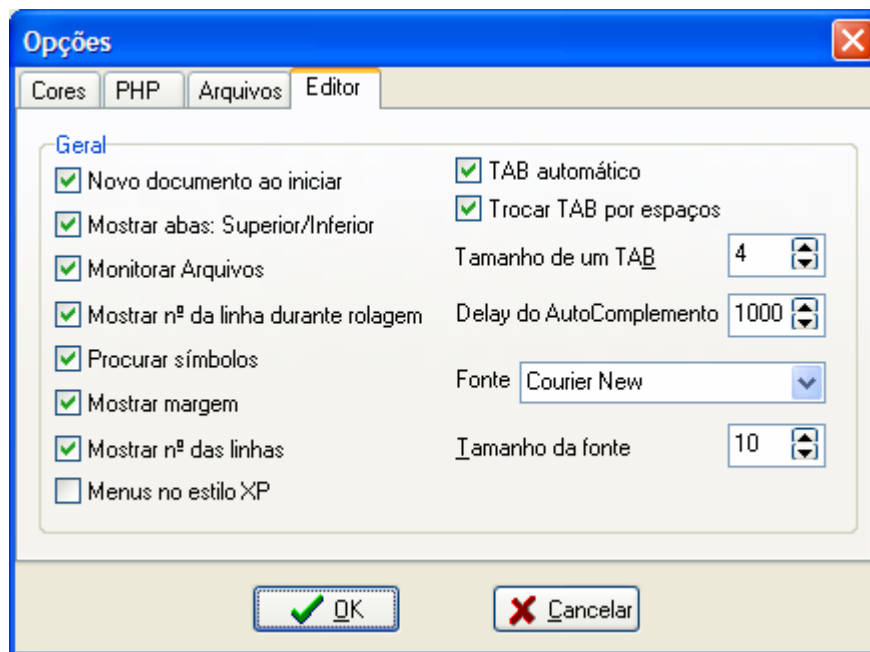


- Default: Alterna janela de exibição: (código/navegador),
- Excluir: Exclui janela de exibição,
- Salvar: Salva janela de exibição.

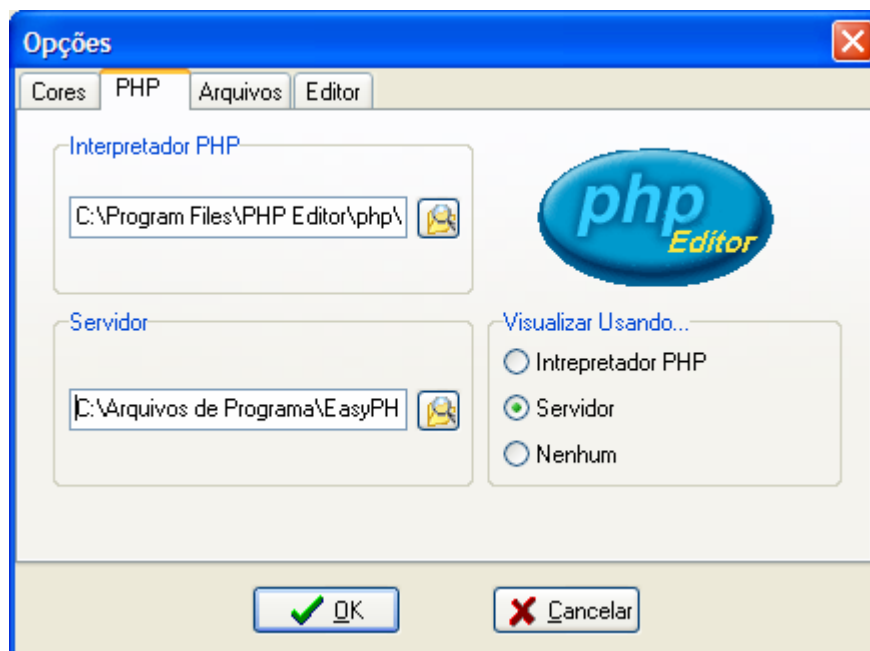
**Importante:** Vamos mudar as opções de configuração da janela de exibição do navegador para o caminho do Easy PHP para facilitar a verificação dos resultados. Para tanto execute os passos a seguir.

## 3.2. Configurando o PHP Editor

1) Clique no menu Opções – Gerais. Surgirá a janela de diálogo abaixo:



2) Clique na Aba PHP, e altere o caminho da opção do servidor para o Easy PHP, conforme abaixo, caminho (“C:\Arquivos de Programa\EasyPHP1-8\www\”).



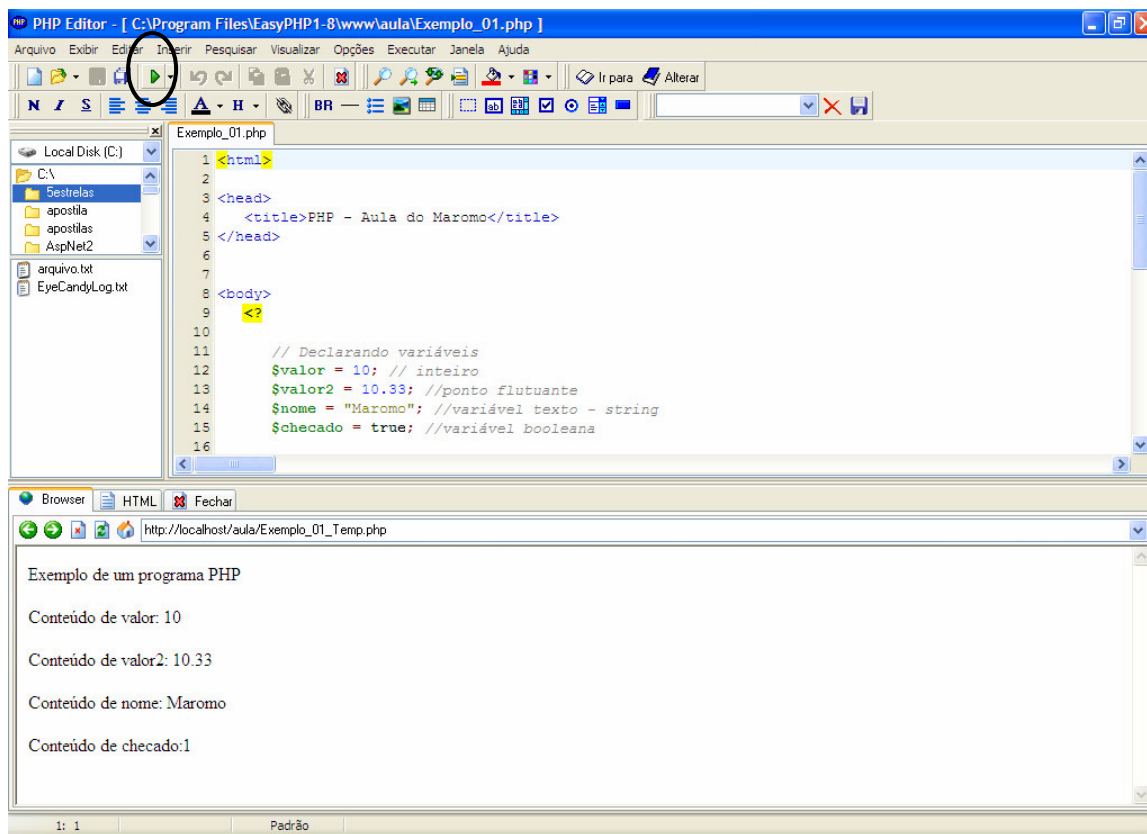
3) Clique no botão OK para confirmar.

**OBS:** com esse procedimento ele executa o código do lado do servidor no Easy PHP, sendo assim, é possível verificar o resultado de um código PHP na janela do navegador quando clicamos no botão Visualizar da Barra de Ferramentas Principal.

### Exemplo:

Com o arquivo Exemplo\_01.php aberto no PHP Editor clique no botão executar e veja o resultado na janela do navegador logo abaixo do código:





### Exercícios com o PHP Editor:

- 1) Abra o arquivo Exemplo\_03.php e execute no navegador.
- 2) Agora altere a cor da fonte para a cor que deseja selecionando o código da cor e alterando utilizando o botão inserir cor da barra: ferramentas.
- 3) Salve e feche o arquivo.

## 3.3. Algo Útil

Vamos fazer alguma coisa um pouco mais útil agora. Nós iremos checar qual é o tipo de navegador que o visitante está utilizando para ver a nossa página. De fato, para fazer isto nós teremos que checar qual é o valor da string agente que o navegador envia como parte de sua requisição HTTP. Esta informação é armazenada em uma variável. Variáveis sempre começam com um símbolo de cifrão no PHP. A variável que nos interessa no momento é a `$_SERVER["HTTP_USER_AGENT"]`.

Nota sobre as Auto-Globais do PHP: `$_SERVER` é uma variável especial reservada do PHP que contém todas as informações sobre o servidor web. Ela é conhecida como uma Auto-Global (ou Superglobal). Estas variáveis especiais foram introduzidas no PHP 4.1.0. Antes desta versão, nós usávamos os velhos arrays `$HTTP_*_VARS`, como o `$HTTP_SERVER_VARS`. Entretanto, este estilo antigo foi removido, porém ainda existem. (Veja a nota sobre códigos antigos.)

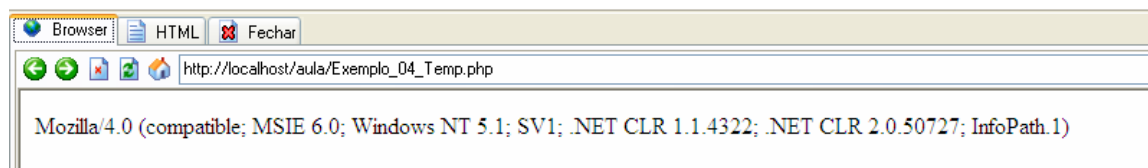
Para chamar esta variável, nós podemos fazer isto:

1) Carregue o PHP Editor. Clique no botão novo documento e escolha (PHP).

2) Digite o código conforme figura abaixo:

```
1 <HTML>
2 <HEAD>
3 <TITLE>Documento PHP</TITLE>
4 </HEAD>
5 <BODY>
6 <?
7 echo $_SERVER["HTTP_USER_AGENT"];
8 ?>
9 </BODY>
10 </HTML>
```

3) Salve o arquivo como Exemplo\_04.php, e execute-o.



Há muitos tipos de variáveis disponíveis no PHP. No exemplo acima nós escrevemos um elemento Array. Arrays podem ser muito úteis.

## Tratamento de Formulários

Uma das características mais fortes do PHP é o jeito como ele trata formulários HTML. O conceito básico que é importante entender é que qualquer elemento de formulário em um formulário irá automaticamente ficar disponível para você usá-los em seus scripts PHP.

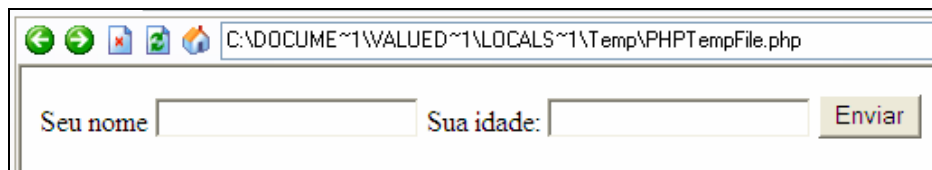
1) Inicie um novo documento HTML no PHP Editor.

2) Digite o Código abaixo:

```
1 <HTML>
2 <HEAD>
3   <TITLE>Formulário</TITLE>
4 </HEAD>
5 <BODY>
6   <form action="acao.php" method="POST">
7     Seu nome <input type="text" name="nome" />
8     Sua idade: <input type="text" name="idade" />
9     <input type="submit" value="Enviar">
10  </BODY>
11 </HTML>
```

3) Salve o código acima como ExemploForm.htm.

4) Execute. O resultado deverá ser como a figura abaixo:



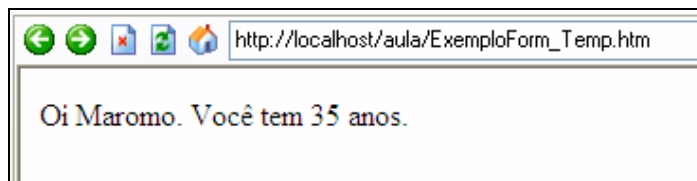
Não há nada de especial neste formulário. É um formulário HTML comum sem nenhuma tag especial de qualquer tipo. Quando o usuário preencher este formulário e clicar no botão enviar, a página `acao.php` é chamada.

Vamos escrever esse arquivo agora. Para tanto inicie um novo documento PHP no PHP Editor e digite o código abaixo:

```
1 <HTML>
2 <HEAD>
3   <TITLE>Documento acao.php</TITLE>
4 </HEAD>
5 <BODY>
6   Oi
7   <? echo $_POST["nome"]; ?>
8   . Você tem
9   <? echo $_POST["idade"]; ?>
10  anos.
11 </BODY>
12 </HTML>
```

Salve o arquivo acima com o nome **“acao.php”**.

Agora abra o abra novamente o arquivo ExemploForm.htm e execute-o totalmente, digitando o seu nome, sua idade e enviando os dados para o servidor. Você terá como resposta algo assim:



É óbvio o que este script faz. Não há nada de mais nele. As variáveis `$_POST["nome"]` e `$_POST["idade"]` são automaticamente criadas para você pelo PHP. Nós simplesmente usamos a auto-global `$_POST` que contém todos os dados vindos do POST.

## 4. Conhecendo melhor os Tipos

O PHP suporta os oitos tipos primitivos.

São quatros tipos básicos:

- boolean,
- integer,
- float (número de ponto flutuante, ou também 'double'),
- string.

Dois tipos compostos:

- array,
- \* object. (consulte documentação do PHP).

E finalmente dois tipos especiais:

- \* resource, (consulte documentação do PHP).
- \* NULL. (consulte documentação do PHP).

\* Nota: Nesse material faremos apenas menção a esses tipos, para maiores informações, consulte a documentação do PHP no manual do PHP no PHP Editor.

Você também pode encontrar algumas referências ao tipo "double". Considere o tipo double como sendo o float, e os dois nomes existem por razões históricas.

O tipo da variável geralmente não é configurado pelo programador: isto é decidido em tempo de execução pelo PHP, dependendo do contexto no qual a variável é usada.

**Nota:** Se você quiser checar o tipo e valor de uma certa expressão, utilize `var_dump()`.

**Nota:** Se você simplesmente quiser uma representação legível de seu tipo para debugagem, use `gettype()`. Para verificar por certos tipos, não use `gettype()`, mas sim as funções `is_type`. Vejamos alguns exemplos:

### Exemplo de tipos de dados:

1) Inicie um novo arquivo PHP no PHP Editor, digite o código abaixo:

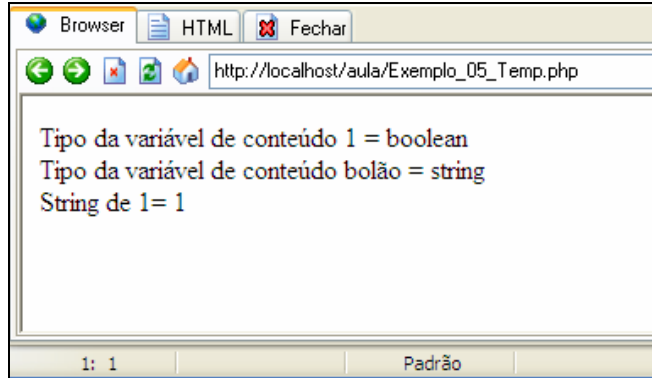
```

1 <HTML>
2 <HEAD>
3 <TITLE>Exemplo de Tipos de Dados</TITLE>
4 </HEAD>
5 <BODY>
6 <?php
7     $bool = TRUE;    // um booleano
8     $str  = "bolão"; // uma string
9     $int  = 12;      // um inteiro
10    echo "Tipo da variável de conteúdo $bool = "; echo gettype($bool); // imprime "boolean"
11    print "<br>";
12    echo "Tipo da variável de conteúdo $str = "; echo gettype($str); // imprime "string"
13    print "<br>";
14    // Se eh um inteiro, incrementa em quatro
15    if (is_int($int)) {
16        $int += 4;
17    }
18    // Se $bool eh string, imprime
19    // (e nao faz nada se nao for)
20    if (is_bool($bool)) {
21        echo "String de $bool= $bool";
22    }
23 ?>
24 </BODY>
25 </HTML>

```

2) Salve o arquivo com o nome Exemplo\_05.php e execute-o.

Você terá como resultado:



Vamos aos tipos básicos:

## 4.1. Booleanos

Este é o tipo mais fácil. Um booleano expressa um valor de verdade. Ele pode ser TRUE ou FALSE. True é representado pelo número inteiro 1.

**Nota:** O tipo booleano foi introduzido no PHP 4.

### Sintaxe

Para especificar um literal booleano, use as palavras chave TRUE ou FALSE.

```
<?php
```

```
$valor = True; // assimila o valor TRUE para $valor
?>
```

Usualmente você pode utilizar algum tipo de operador que retorne um valor booleano, e passá-lo para uma estrutura de controle.

```
<?php
if ($action == "mostrar-versao") {
    echo "A versão é 1.23";
}

// isto não é necessário ...
if ($exibir_separadores == TRUE) {
    echo "<hr>\n";
}

// ... porque você pode simplesmente escrever isso:
if ($exibir_separadores) {
    echo "<hr>\n";
}
?>
```

## 4.2. Inteiros

Um inteiro é um número do conjunto  $Z = \{..., -2, -1, 0, 1, 2, ...\}$ .

### Sintaxe

Inteiros podem ser especificados em notação decimal (base 10), hexadecimal (base 16) ou octal (base 8), opcionalmente precedido de sinal (- ou +).

Para usar a notação octal, você precisa preceder o número com um 0 (zero). Para utilizar a notação hexadecimal, preceda número com 0x.

```
<?php
$a = 1234; # número decimal
$a = -123; # um número negativo
$a = 0123; # número octal (equivalente a 83 em decimal)
$a = 0x1A; # número hexadecimal (equivalente a 26 em decimal)
?>
```

Formalmente, as possíveis representação de inteiros são:

```
<?php
decimal      : [1-9][0-9]*
              | 0

hexadecimal  : 0[xX][0-9a-fA-F]+

octal        : 0[0-7]+

integer      : [+]?decimal
              | [+]?hexadecimal
              | [+]?octal
?>
```

O tamanho de um inteiro é dependente de plataforma, sendo um numero aproximado a 2 bilhões o valor mais comum (número de 32 bits com sinal). O PHP não suporta inteiros sem sinal.

## Overflow de inteiros

Se você especifica um número além dos limites do tipo inteiro, ele será interpretado como um ponto flutuante. Assim, se você realizar uma operação que resulte em um número além dos limites do tipo inteiro, um ponto flutuante será retornado também.

```
<?php
$ numero_grande = 2147483647;
var_dump($numero_grande);
// saida: int(2147483647)

$numero_grande = 2147483648;
var_dump($numero_grande);
// saida: float(2147483648)

// isto também é permitido para especificação de números hexadecimais:
var_dump( 0x80000000 );
// saida: float(2147483648)
$milhao = 1000000;
$numero_grande = 50000 * $milhao;
var_dump($numero_grande);
// saida: float(500000000000)
?>
```

**Atenção:** Infelizmente, há um bug no PHP que faz que ele nem sempre trabalhe corretamente quando há números negativos envolvidos. Por exemplo, quando você faz `-50000 * $milhao`, o resultado será `-429496728`. Entretanto, quando ambos os operadores são positivos, isso não ocorre. Isto foi resolvido no PHP 4.1.0.

Não há operador de divisão inteira no PHP. `1/2` retorna o ponto flutuante `0.5`. Você pode moldar (cast) o valor para inteiro para sempre truncar o número, ou você pode usar a função `round()`.

```
<?php
var_dump(25/7);           // float(3.5714285714286)
var_dump((int) (25/7));   // int(3)
var_dump(round(25/7));     // float(4)
?>
```

## 4.3. Números de ponto flutuante

Números de ponto flutuante (AKA "floats", "doubles" ou "números reais") podem ser especificados utilizando qualquer uma das sintaxes seguintes:

```
<?php
$a = 1.234;
$b = 1.2e3;
$c = 7E-10;
?>
```

Formalmente:



LNUM	[0-9]+
DNUM	([0-9]*[\.]{LNUM})   ({LNUM}[\.]{0-9}*)
EXPONENT_DNUM	( ({LNUM}   {DNUM}) [eE] [+]? {LNUM})

O tamanho de um número de ponto flutuante é dependente de plataforma, sendo o máximo de ~1.8e308 com uma precisão de 14 decimais digitais um valor comum (número de 64 bits no formato IEEE).

#### ***Precisão de números de ponto flutuante***

*É sabido que frações simples como 0.1 ou 0.7 não podem ser convertidos em sua representação binária interna sem uma pequena perda de precisão. Isto pode causar erros confusos: por exemplo, floor((0.1+0.7)\*10) irá retornar 7 em vez do esperado 8, como resultado da representação interna realmente ser algo como 7.999999999....*

*Isto está relacionado ao fato de que é impossível expressar, exatamente, algumas frações em notação decimal com um número finito de dígitos. Por exemplo, 1/3 na forma decimal se torna 0.3333333... .*

## 4.4. Strings

Uma string é uma série de caracteres. No PHP, um caracter é o mesmo que um byte, ou seja, há exatamente 256 caracteres diferentes possíveis. Isto implica que o PHP não tem suporte nativo ao Unicode. Veja utf8\_encode( ) e utf8\_decode( ) para suporte ao Unicode.

***Nota:*** Não há nenhum problema nas strings se tornarem muito grandes. Não há nenhum limite para o tamanho de strings imposta pelo PHP, então não há razão para se preocupar com strings longas.

### Sintaxe

Uma string literal pode ser especificada de três formas diferentes.

- Apóstrofo.
- Aspas.
- Sintaxe heredoc.

#### 4.4.1. Apóstrofos

A maneira mais simples para especificar uma string é delimitá-la entre apóstrofos (o caracter ').

Para especificar um apóstrofo você precisará "escapá-la" com uma contra barra (\), como em muitas outras linguagens. Se uma contra barra precisa ocorrer antes de um apóstrofo ou no final da string, você precisa duplicá-la. Note que se você tentar escapar qualquer outro caracter, a contra barra também será impressa! Então geralmente não é necessário escapar a própria contra barra.

```
<?php
echo 'isto é uma string comum';

echo 'Você pode incluir novas linhas em strings,
dessa maneira que estará
tudo bem';

// Imprime: Arnold disse uma vez: "I\'ll be back"
```

```

echo 'Arnold once said: "I\'ll be back"';

// Imprime: Você tem certeza em apagar C:\*.*?
echo 'Você tem certeza em apagar C:\\*.*?';

// Imprime: Você tem certeza em apagar C:\*.*?
echo 'Você tem certeza em apagar C:\*.*?';

// Imprime: Isto não será substituído: \n uma nova linha
echo 'Isto não será substituído: \n uma nova linha';

// Imprime: Variáveis $também não $expandem
echo 'Variáveis $também não $expandem';
?>

```

#### 4.4.2. Aspas

Se a string é delimitada entre aspas ("), o PHP entende mais seqüências de escape para caracteres especiais:

**Tabela - Seqüências de escape**

Seqüência	Significado
\n	fim de linha (linefeed ou LF ou 0x0A (10) em ASCII)
\r	retorno de carro (carriage return ou CR ou 0x0D (13) em ASCII)
\t	TAB horizontal (HT ou 0x09 (9) em ASCII)
\\	contra barra ou barra invertida
\\$	sinal de cifrão
\"	aspas
\[0-7]{1,3}	a seqüência de caracteres batendo a expressão regular dos caracteres em notação octal
\x[0-9A-Fa-f]{1,2}	a seqüência de caracteres batendo a expressão regular de um caracter em notação hexadecimal

Novamente se você tentar escapar qualquer outro caracter, a contra barra será impressa também!

Mas o mais importante recurso de strings delimitadas por aspas está no fato de que nome de variáveis serão substituídos. Veja interpretação de strings para detalhes.

#### 4.4.3. Heredoc

Outra maneira para delimitar strings é utilizando a sintaxe heredoc ("<<<"). É informado um identificador depois de <<<, então a string, e então o mesmo identificador para fechar a delimitação.

O identificador de fechamento precisa começar na primeira coluna da linha. Além, o identificador utilizado precisa seguir as mesmas regras de nomeação que qualquer outro rótulo no PHP: só pode conter caracteres alfanuméricos e sublinhados, e precisa começar com um caracter não numérico ou sublinhado.

Textos heredoc se comportam como strings delimitadas por aspas, com apenas uma diferença. Você não precisa escapar apóstrofes e aspas em seus heredocs, mas você ainda pode continuar utilizando

os códigos de escape listados acima. Variáveis são substituídas, mas o mesmo cuidado precisa ser tomado quando expressando variáveis complexas dentro de heredocs assim como nas strings.

### Exemplo de delimitação de strings heredoc

```
<?php
$str = <<<EOD
Exemplo de uma string distribuída em
várias linhas utilizando a sintaxe heredoc.
EOD;
/* Exemplo mais complexo, com variáveis */
class foo
{
    var $foo;
    var $bar;
    function foo()
    {
        $this->foo = 'Foo';
        $this->bar = array('Bar1', 'Bar2', 'Bar3');
    }
}
$foo = new foo();
$name = 'Meu nome';
echo <<<EOT
Meu nome é "$name". Eu estou imprimindo $foo->foo.
Agora, eu estou imprimindo {$foo->bar[1]}.
Isto deve imprimir um 'A' maiúsculo: \x41
EOT;
?>
```

#### 4.4.4. Interpretação de variáveis

Quando uma string é especificada dentro de aspas ou heredoc, variáveis são interpretadas dentro delas.

Há dois tipos de sintaxe: um **simples** e um **complexo**. A sintaxe simples é a mais comum e conveniente, provendo uma maneira de interpretar uma variável, um valor de array ou uma propriedade de object.

A sintaxe completa foi introduzida no PHP 4, e pode ser reconhecida por chaves ({} ) envolvendo a expressão.

### Sintaxe simples

Se um sinal de cifrão (\$) é encontrado, o interpretador tentará obter tantos identificadores quanto possíveis para formar um nome de variável válido. Envolve o nome da variável com chaves se você deseja explicitamente especificar o fim do nome.

```
<?php
$cerveja = 'Heineken';
echo "O sabor das '$cerveja's é otimo"; // funciona, "'" é um caracter inválido
para nome de variáveis
echo "Ele bebeu algumas $cervejas";      // não funciona, 's' é um caracter
válido para nome de variáveis
echo "Ele bebeu algumas ${cerveja}s";    // funciona
echo "Ele bebeu algumas {$cerveja}s";    // funciona
?>
```

Similarmente, você também pode interpretar um índice de array ou uma propriedade de objeto. Com índices de arrays, o colchete de fechamento (]) marca o final do índice. Para propriedades de objetos se aplicam as mesmas regras das variáveis comuns, embora não exista um truque para as propriedades de objetos como para as variáveis.

```
<?php
// Esses exemplos são específicos para utilização de arrays dentro de strings
// Quando fora de strings, sempre delimite suas chaves de array strings
// e nao use {colchetes} fora das strings tambem.

// Vamos ver todos os erros
error_reporting(E_ALL);

$fruits = array('morango' => 'vermelho', 'banana' => 'amarelo');

// Funciona, mas note que funciona de maneira diferente fora dos delimitadores
de strings
echo "A banana é $fruits[banana].";

// Funciona
echo "A banana é {$fruits['banana']}.";

// Funciona, mas o PHP procura por uma constante chamada 'banana' antes,
// como descrito a seguir
echo "A banana é {$fruits[banana]}.";

// Nao funciona, use colchetes. Isto lanca um parse error.
echo "A banana é $fruits['banana'].";

// Funciona
echo "A banana é " . $fruits['banana'] . ".";

// Funciona
echo "Este quadrado tem $square->width metros de lado.";

// Nao funciona. Para uma solução, veja a sintaxe complexa.
echo "Este quadrado tem $square->width00 centímetros de lado.";
?>
```

Para qualquer coisa mais complexa, você precisa utilizar a sintaxe complexa.

## Sintaxe complexa (chaves)

Isto não é chamado sintaxe complexa porque a sintaxe em si é complexa, mas porque você pode incluir expressões complexas desta maneira.

De fato, você pode incluir qualquer valor no que esteja no espaço de nomes dentro de strings com esta sintaxe. Você simplesmente escreve a expressão da mesma maneira que faria fora da string, e então incluí-la entre chaves. Desde que você não pode escapar '{', esta sintaxe somente será reconhecida quando o \$ é imediatamente seguido de um {. (Utilize "{\\$" ou "\{\$" para obter um literal "{\$"). Alguns exemplos para tornar isso mais claro:

```
<?php
// Vamos ver todos os erros
error_reporting(E_ALL);

$bom = 'fantastico';
```

```
// Nao funciona, imprimindo: Isto é { fantastico}
echo "Isto é { $bom}";

// Funciona, imprimindo: Isto é fantástico
echo "Isto é {$bom}";
echo "Isto é ${bom}";

// Funciona
echo "Este quadrado tem {$square->width}00 centímetros de lado.";

// Funciona
echo "Isto funciona: {$arr[4][3]}";

// Isto está errado pela mesma razão que $foo[bar] eh errado
// fora de uma string. Em outras palavras, isto ainda funciona MAS
// porque o PHP primeiro procura por uma constante nomeada foo, e ele
// lancara um erro do tipo E_NOTICE (undefined constant).
echo "Isto é errado: {$arr[foo][3]}";

// Funciona. Quanto mexendo com arrays multi dimensionais, sempre use
// colchetes em volta dos arrays quando dentro de strings
echo "Isto funciona: {$arr['foo'][3]}";

// Funciona
echo "Isto funciona: " . $arr['foo'][3];

echo "Você pode escrever também {$obj->values[3]->name}";

echo "Este é o valor da variável chamada $name: ${$name}";
?>
```

## Acesso a caracteres da string

Caracteres nas strings podem ser acessados apenas especificando o deslocamento baseado em zero do caractere desejado depois da string dentro de chaves.

### Alguns exemplos com strings

```
<?php
// Pega o primeiro caractere da string
$str = 'Isto é um teste.';
$first = $str{0};

// Pega o terceiro caractere da string
$third = $str{2};

// Pega o último caractere da string
$str = 'Isto ainda é um teste.';
$last = $str{strlen($str)-1};
?>
```

## 4.5. Arrays

Um array no PHP é atualmente um mapa ordenado. Um mapa é um tipo que relaciona valores para chaves. Este tipo é otimizado de várias maneiras, então você pode usá-lo como um array real, ou uma lista (vetor), hashtable (que é uma implementação de mapa), dicionário, coleção, pilha, fila e provavelmente mais. Como você pode ter outro array PHP como um valor, você pode facilmente simular árvores.

## Sintaxe

### Especificando com array( )

Um array pode ser criado com o construtor de linguagem array( ). Ele pega um certo número de pares separados por vírgula chave => valor .

```
array( [chave =>] valor
      , ...
      )
// chave pode ser tanto string ou um integer
// valor pode ser qualquer coisa
```

```
<?php
$arr = array("foo" => "bar", 12 => true);

echo $arr["foo"]; // bar
echo $arr[12];    // 1
?>
```

A chave pode ser tanto um integer ou uma string. Se a chave é uma representação padrão de um integer, ele será interpretado assim (por exemplo, "8" será interpretado como 8, enquanto "08" será interpretado como "08"). Não há diferença entre arrays indexados e associativos em PHP, apenas um tipo de array, que pode ter índices inteiros ou string.

O valor pode ser qualquer tipo PHP:

```
<?php
$arr = array("somearray" => array(6 => 5, 13 => 9, "a" => 42));

echo $arr["somearray"][6];    // 5
echo $arr["somearray"][13];   // 9
echo $arr["somearray"]["a"];  // 42
?>
```

Se omitir a chave quando fornece um novo item, o maior índice inteiro é obtido, e a nova chave será esse máximo + 1. Se você especificar uma chave que já possui um valor assimilada a ela, então o valor é sobrescrito.

```
<?php
// Esse array eh como ...
array(5 => 43, 32, 56, "b" => 12);

// ... este array
array(5 => 43, 6 => 32, 7 => 56, "b" => 12);
?>
```

**Atenção:** A partir do PHP 4.3.0, o comportamento da geração de índice descrito acima foi modificado. Agora, se você aumentar um array em que o maior índice atual for negativo, então a próxima chave criada será zero (0). Antes, o novo índice seria o maior índice existente mais 1, do mesmo jeito que os índices positivos.

### Exemplo de array:

```
<?php
// Criando um array normal
$array = array(1, 2, 3, 4, 5);
print_r($array);

// Agora apagando todos os itens, mas deixando o array intacto:
foreach ($array as $i => $value) {
    unset($array[$i]);
}
print_r($array);

// Acrescentando um item (note que a chave eh 5, em vez de zero
// como voce pode ter esperado).
$array[] = 6;
print_r($array);

// Reindexando:
$array = array_values($array);
$array[] = 7;
print_r($array);
?>
```

O exemplo acima deve produzir a seguinte saída:

```
Array
(
    [0] => 1
    [1] => 2
    [2] => 3
    [3] => 4
    [4] => 5
)
Array
(
)
Array
(
    [5] => 6
)
Array
(
    [0] => 6
    [1] => 7
)
```

### Exemplos

O tipo array do PHP é muito versátil, por isso temos aqui alguns exemplos para mostrar todo o poder dos arrays.

```
<?php
// isto
```

```

$a = array( 'cor'    => 'vermelha',
            'sabor' => 'doce',
            'forma' => 'redonda',
            'nome'  => 'maçã',
            4        // a chave será 0
        );

// isto é equivalente a acima
$a['cor']    = 'vermelha';
$a['sabor']  = 'doce';
$a['forma']  = 'redonda';
$a['nome']   = 'maçã';
$a[]        = 4;        // a chave será 0

$b[] = 'a';
$b[] = 'b';
$b[] = 'c';
// o mesmo de array( 0 => 'a' , 1 => 'b' , 2 => 'c' ),
// ou simplesmente array('a', 'b', 'c')
?>

```

### Utilizando array()

```

<?php
// Array como (propriedade-)mapa
$map = array( 'versao'    => 4,
            'OS'         => 'Linux',
            'lang'       => 'inglês',
            'short_tags' => true
        );

// apenas chaves numéricas
$array = array( 7,
            8,
            0,
            156,
            -10
        );
// que é o mesmo que array( 0 => 7, 1 => 8, ...)

$switching = array(
            10, // chave = 0
            5  => 6,
            3  => 7,
            'a' => 4,
            11, // chave = 6 (o índice máximo era 5)
            '8' => 2, // chave = 8 (inteiro!)
            '02' => 77, // chave = '02'
            0  => 12 // o valor 10 será sobrescrito por 12
        );

// array vazio
$empty = array();
?>

```

### Coleção

```

<?php
$scores = array('vermelho', 'azul', 'verde', 'amarelo');

foreach ($scores as $cor) {
    echo "Você gosta de $cor?\n";
}

```



```
}

/* saida:
Você gosta de vermelho?
Você gosta de azul?
Você gosta de verde?
Você gosta de amarelo?
*/
?>
```

Note que atualmente não se pode mudar os valores de um array diretamente dentro de um loop. Superar essa limitação é possível da seguinte forma:

```
<?php
foreach ($cores as $key => $cor) {
    // não funciona:
    //$cor = strtoupper($cor);

    //funciona:
    $cores[$key] = strtoupper($cor);
}
print_r($cores);

/* saida:
Array
(
    [0] => VERMELHO
    [1] => AZUL
    [2] => VERDE
    [3] => AMARELO
)
*/
?>
```

### Este exemplo cria um array na base 1.

```
<?php
$primeiroquarto = array(1 => 'Janeiro', 'Fevereiro', 'Março');
print_r($primeiroquarto);

/* saida:
Array
(
    [1] => 'Janeiro'
    [2] => 'Fevereiro'
    [3] => 'Março'
)
*/
?>
```

### Preenchendo um array real

```
// preenchendo um array com todos os itens de um diretório
$handle = opendir('.');
while (false !== ($file = readdir($handle))) {
    $files[] = $file;
}
closedir($handle);
?>
```

Arrays são ordenados. Você pode mudar sua ordem utilizando várias funções de ordenação. Veja as funções de arrays para mais informações. Você pode contar o número de itens de um array com a função `count()`.

### **Ordenando arrays**

```
<?php
sort($files);
print_r($files);
?>
```

## 5. Manipulando Arquivos

Este capítulo mostra como manipular arquivos através de um exemplo prático de utilização de forms e de armazenamento de dados em arquivos do tipo texto por meio de funções próprias.

### 5.1. Arquivos

Em PHP você pode acessar arquivos locais e remotos, no nosso caso, toramemos como base os arquivos armazenados no nosso servidor de teste.

***Nota:** Nossos testes se baseiam no sistema operacional windows, no caso do Linux / Unix deverá haver uma preocupação especial com as permissões de arquivos.*

### 5.2. Abrindo um arquivo

Para abrir um arquivo usa-se a função fopen:

```
$arquivo = fopen(PATH,MODO, int[use_include_path]);
```

onde arquivo é uma variável que armazenará o ponteiro do arquivo, PATH é o caminho do arquivo e MODO pode ser:

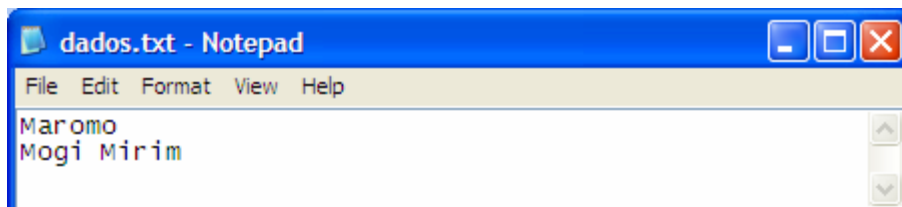
r	leitura
r+	leitura e gravação
w	gravação. Se o arquivo existir será substituído
w+	gravação. Se o arquivo existir será substituído
a	gravação. Acrescenta ao fim do arquivo
a+	gravação. Acrescenta ao fim do arquivo
b	binário. Utilizado em sistemas que diferenciam arquivos de texto de binários (windows)

O terceiro parâmetro (use\_include\_path), deve ser setado como 1 ou 0 para determinar se você quer utilizar o path parametrizado no arquivo php.ini para procurar o arquivo.

No caso de erro a função fopen retorna false.

#### Exemplo:

1) Crie um arquivo do tipo texto no bloco de notas com o seu nome na primeira linha e sua cidade de origem na segunda linha. Grave esse arquivo na pasta dos nossos scripts (c:\arquivos de programas\easy php1.8\www\aula) com o nome “dados.txt”.



2) Abra o PHP Editor e digite o código abaixo para abertura do arquivo no modo somente leitura:

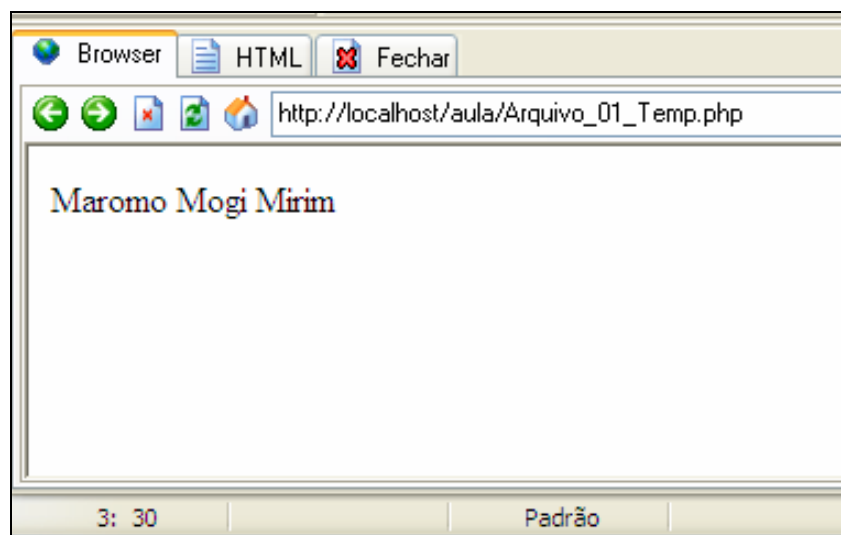
```
1 <HTML>
2 <HEAD>
3 <TITLE>Manipulação de Textos</TITLE>
4 </HEAD>
5 <BODY>
6 <?
7 // a linha abaixo indica que a variável
8 // $arq está setada para o arquivo dados.txt
9 $arq = fopen("dados.txt","r",0);
10
11 // a linha abaixo indica que a variável
12 // $texto receberá o conteúdo do arquivo dados.txt
13 // em uma linha
14 $texto = fread($arq, filesize("dados.txt"));
15
16 // a linha abaixo, exibe o conteúdo da variável $texto
17 echo $texto;
18 ?>
19 </BODY>
20 </HTML>
```

3) Salve o arquivo com o nome Arquivo\_01.php na pasta aula, em seguida execute-o, o resultado será a exibição do seu nome e cidade de origem em uma linha no navegador.

## 5.3. Lendo um Arquivo

A função filesize(arq) retorna o tamanho da cadeia de caracteres do arquivo especificado, já o arquivo fread() executa a leitura do arquivo informado.

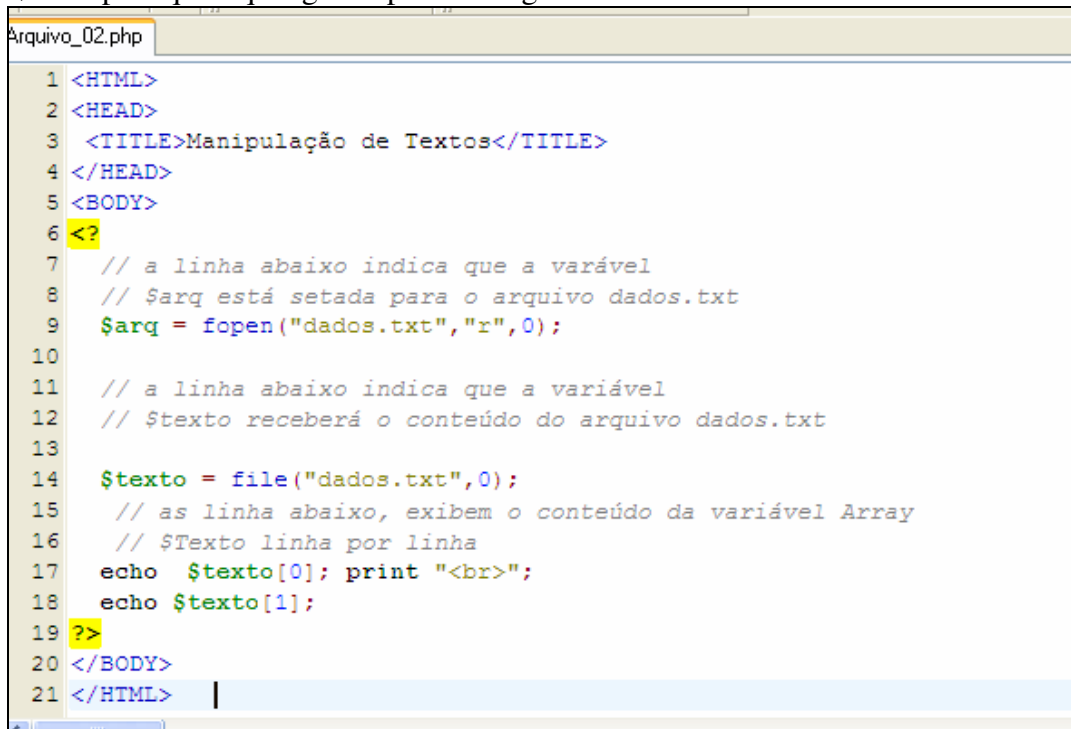
Resultado será:



Nota: Uma função muito útil para leitura de arquivos de texto é a função *file()*, pois ela lê todo o conteúdo do arquivo e retorna um array onde cada posição é uma linha do arquivo. Sintaxe do comando:

**file(string arquivo, int use\_include\_path);**

4) Salve o arquivo Arquivo\_01.php como Arquivo\_02.php e faça as alterações a partir da linha da variável \$texto para que fique igual a próxima figura:



```

1 <HTML>
2 <HEAD>
3 <TITLE>Manipulação de Textos</TITLE>
4 </HEAD>
5 <BODY>
6 <?
7 // a linha abaixo indica que a variável
8 // $arq está setada para o arquivo dados.txt
9 $arq = fopen("dados.txt","r",0);
10
11 // a linha abaixo indica que a variável
12 // $texto receberá o conteúdo do arquivo dados.txt
13
14 $texto = file("dados.txt",0);
15 // as linha abaixo, exibem o conteúdo da variável Array
16 // $texto linha por linha
17 echo $texto[0]; print "<br>";
18 echo $texto[1];
19 ?>
20 </BODY>
21 </HTML>

```

5) Salve e teste novamente, você verá o conteúdo separado por linha.

## 5.4. Verificando se um arquivo existe:

Para evitar erros ou tratá-los, é importante saber se um determinado arquivo existe, para isto utilizamos a função *file\_exists*. Sintaxe:

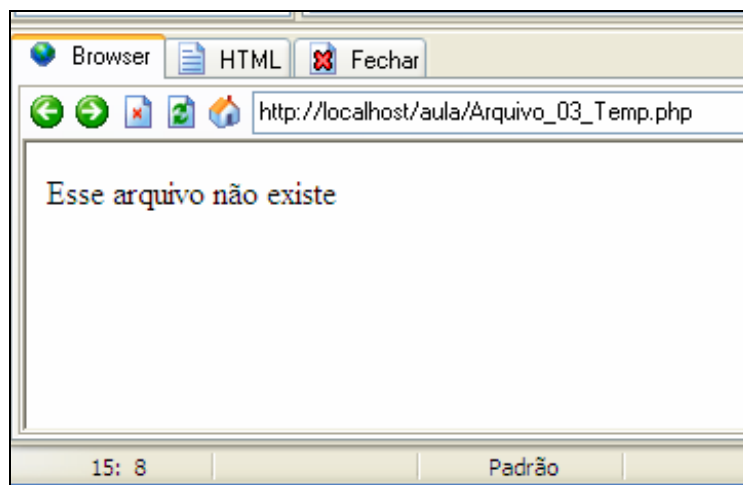
```
int file_exists(string arquivo);
```

Esta função retorna true ou false, veja o exemplo utilizando um comando IF para verificação. Exemplo:

1) Inicie um novo projeto PHP no PHP Editor e digite o código conforme abaixo:

```
1 <HTML>
2 <HEAD>
3   <TITLE>Documento PHP</TITLE>
4 </HEAD>
5 <BODY>
6 <?php
7     clearstatcache();
8     if (file_exists("dados1.txt")) {
9         $arq = fopen("dados1.txt", "r");
10        $conteudo = fread($arq, filesize("dados1.txt"));
11        echo $conteudo;
12        fclose($arq);
13    } else {
14        echo "Esse arquivo não existe"; print "<BR>";
15    }
16 ?>
17 </BODY>
18 </HTML>
```

2) Salve na pasta aula com o nome Arquivo\_03.php. Execute-o. Você terá como resposta no seu browser a resposta: “esse arquivo não existe”.



3) Agora altere o nome do arquivo a ser aberto de “dados1.txt” para “dados.txt”. conforme figura abaixo.

```
Sem título 1 Arquivo_03.php
1 <HTML>
2 <HEAD>
3   <TITLE>Documento PHP</TITLE>
4 </HEAD>
5 <BODY>
6 <?php
7     clearstatcache();
8     if (file_exists("dados.txt")) {
9         $arq = fopen("dados.txt", "r");
10        $conteudo = fread($arq, filesize("dados.txt"));
11        echo $conteudo;
12        fclose($arq);
13    } else {
14        echo "Esse arquivo não existe"; print "<BR>";
15    }
16 ?>
17 </BODY>
18 </HTML>
```

4) Salve e execute novamente, você terá como resultado o arquivo aberto com o conteúdo em uma linha.

#### **Alguns comentários sobre esse exemplo:**

- A função `clearstatcache( )` é responsável por limpar as informações em cache de arquivos.
- A função `file_exists("dados.txt")` retorna `TRUE` pois existe o arquivo na "dados.txt" na pasta onde o script `Arquivo_03.php` foi salvo.

#### **Gravando dados em Arquivos (Escrevendo)**

Primeiramente, antes de escrever dados no arquivo, o mesmo deve ser aberto em modo que permita a escrita, ou seja `w,w+,a,a+`. Para escrever dados utilizamos a função:

```
int fwrite ( int fp, string string [, int comprimento])
```

A função **`fwrite( )`** grava os conteúdos de `string` para o stream de arquivo apontado por `fp`. Se o argumento `comprimento` é dado, a gravação irá parar depois de que `comprimento` bytes foram escritos ou o fim da `string` é alcançada, o que ocorrer primeiro. A função retorna o número de bytes gravados, ou `FALSE` em caso de erro.

1) Inicie um novo projeto PHP no PHP Editor e digite o código conforme próxima figura:

```

Arquivo_04.php
1 <HTML>
2 <HEAD>
3   <TITLE>Documento PHP</TITLE>
4 </HEAD>
5 <BODY>
6 <?php
7   $arq = 'dados.txt';
8   $conteudo = "\nMariana da Silva\nJaguariúna";
9   // Tendo certeza que o arquivo existe e que há permissão de escrita primeiro.
10  if (is_writable($arq)) {
11    // Em nosso exemplo, nós estamos abrindo $arq em modo de append (acréscimo).
12    // O ponteiro do arquivo estará no final dele desde
13    // que será aqui que $conteudo será escrito com fwrite().
14    if (!$fp = fopen($arq, 'a')) {
15      print "Erro abrindo arquivo ($arq)";
16      exit;
17    }
18    // Escrevendo $conteudo para o arquivo aberto.
19    if (!fwrite($fp, $conteudo)) {
20      print "Erro escrevendo no arquivo ($conteudo)";
21      exit;
22    }
23    print "Sucesso: escrito ($conteudo) no arquivo ($arq)";
24    fclose($fp);
25  } else {
26    print "O Arquivo $arq não possui permissão de escrita";
27  }
28  ?>
29 </BODY>
30 </HTML>

```

2) Salve como Arquivo\_04.php e execute. O Texto Mariana da Silva será incluído dentro do arquivo dados.txt e com uma quebra de linha (\n) será incluída na sequência a palavra Jaguariúna.

O script Arquivo\_04.php já está comentado para o seu auxílio.

Nota: pela segunda vez utilizamos a função fclose ( ) que é responsável por fechar o ponteiro para um arquivo aberto anteriormente pelo fopen ( ).

## 5.5. Montando uma pequena página de recados:

Para montar nossa página de recados, começaremos pela página principal em PHP a qual chamaremos de recados.php

1) Inicie um novo projeto em PHP no PHP Editor, e digite o código abaixo:

```

<HTML>
<HEAD>
  <TITLE>Página Principal dos Recados</TITLE>
</HEAD>
<BODY>
<H1>Recados</H1>

```



```

<?php
    $dados = file("recados.txt",0);
    for ($i=0;$i<sizeof($dados);$i++)
    {
        echo $dados[$i];
        echo "<br>";
    }
?>

<form action="envia.php" method="POST">
<table border="1" width="100%">
<br>
<h2>Enviar Recados</h2>
<br>
<tr>
    <td>Data:</td>
    <td><input type="text" size="20" name="txtData"></td>
</tr>
<tr>
    <td>Hora:</td>
    <td><input type="text" size="20" name="txtHora"></td>
</tr>
<tr>
    <td>Recado:</td>
    <td><textarea rows="4" cols="40" name="txtRecado"></textarea><input
type="submit" value="Enviar" name="btnEnviar"></td>
</tr>
</table>
</form>
</BODY>
</HTML>

```

2) Salve o arquivo como recados.php

3) Crie um arquivo no bloco de notas chamado “recados.txt” sem conteúdo e grave na pasta aula.

4) Execute o arquivo para verificar se não há erros. No entanto nesse momento ainda não criamos o arquivo “envia.php” (nosso próximo passo), dessa maneira não poderemos ainda enviar recados para a página.

5) Agora vamos criar o arquivo “envia.php” no PHP Editor. Digite o código abaixo:

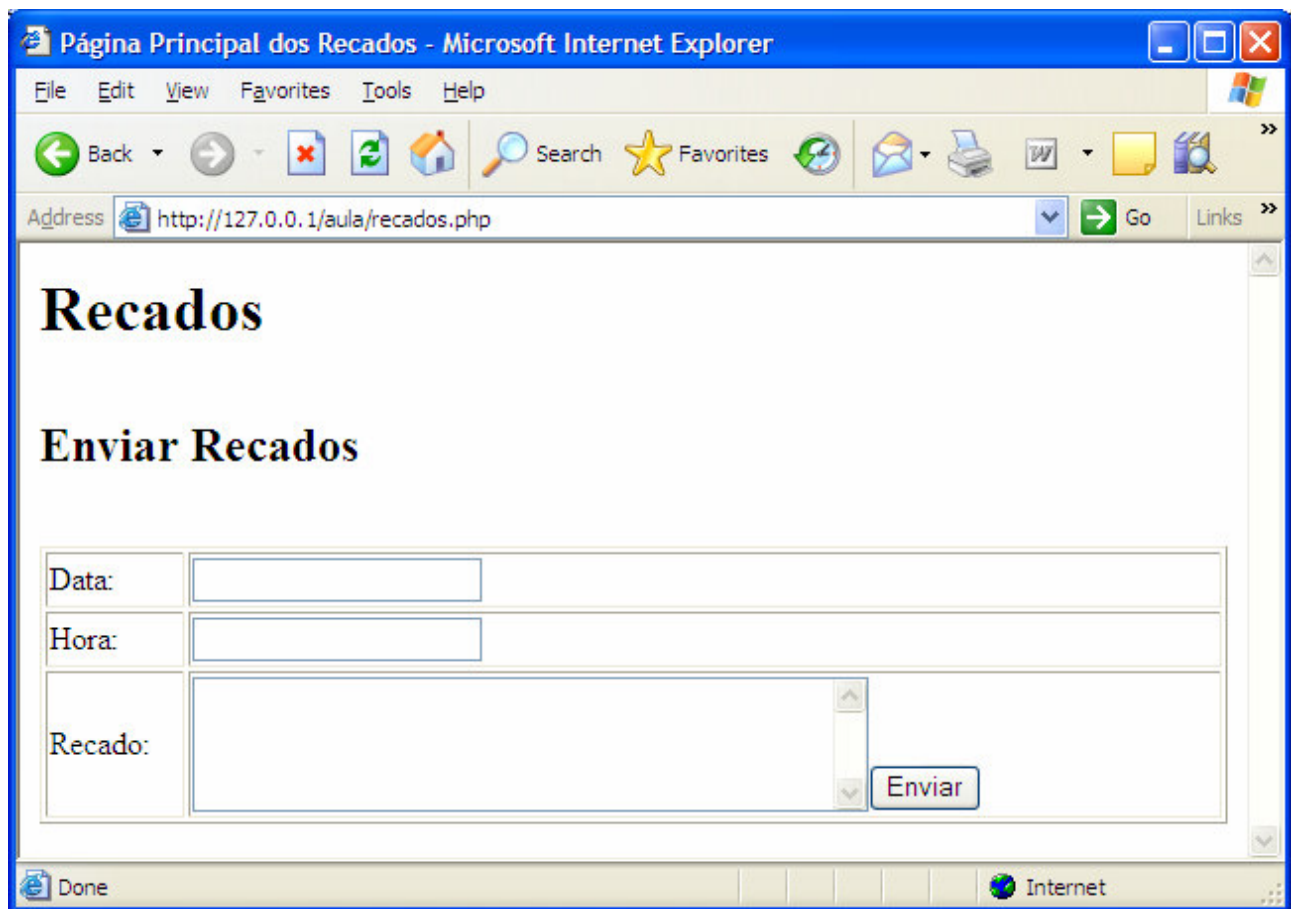
```

<HTML>
<HEAD>
    <TITLE>Enviando Dados</TITLE>
</HEAD>
<BODY>
<?php
    // abrindo o arquivo recados.txt e
    // posicionando o ponteiro no fim do arquivo
    $data= $_POST["txtData"];
    $hora= $_POST["txtHora"];
    $recado = $_POST["txtRecado"];
    $arq = fopen("recados.txt","a+",0);
    $conteudo=" \n\n";
    $conteudo=$conteudo."Data: ".$data."\n"; //Inclui a data
    $conteudo=$conteudo."Hora: ".$hora."\n"; //Inclui a data
    $conteudo=$conteudo."Recado: ".$recado."\n\n"; //Inclui a data
    fwrite($arq,$conteudo,strlen($conteudo));
    fclose($arq);

```

```
//executar a página recados.php  
echo '<script>window.location="recados.php";</script>';  
?>  
</BODY>  
</HTML>
```

- 6) Salve o arquivo envia.php na pasta aula.
- 7) Abra o navegador e execute o arquivo recados.php.



- 8) Digite os dados e clique no botão enviar.

**Página Principal dos Recados - Microsoft Internet Explorer**

File Edit View Favorites Tools Help

Back Forward Stop Home Search Favorites Reload Print Mail News RSS Feeds

Address <http://127.0.0.1/aula/recados.php> Go Links

# Recados

Data: 12/12/2006  
Hora: 20:45  
Recado: Teste da página de recados.

## Enviar Recados

Data:	<input type="text"/>
Hora:	<input type="text"/>
Recado:	<input type="text"/>

Done Internet

**Nota:** Método POST - Quando um formulário é submetido para um script PHP, qualquer variável do formulário será automaticamente disponível para o script.  
Dependendo da configuração local e suas preferências pessoais, essas são as vias pela qual você pode acessar os dados de seus formulários:

### Exemplo - Acessando dados de um formulário HTML via POST

```
<?php
// Disponível desde o PHP 4.1.0
print $_POST['username'];
print $_REQUEST['username'];
import_request_variables('p', 'p_');
print $p_username;
// Disponível desde o PHP 3. A partir do PHP 5.0.0, essas grandes
// variáveis pre-definidas podem ser desabilitadas pela diretiva register_long_arrays.
print $HTTP_POST_VARS['username'];
// Disponível se a diretiva register_globals = on.
// Desde o PHP 4.2.0 o valor default de register_globals é off
```

// Usar/manter esse método é preferível.

```
print $username;
```

```
?>
```

## 6. Gerenciamento de Banco de Dados com MySQL

Faremos nesse capítulo uma breve introdução ao Sistema Gerenciador de Banco de Dados MySQL.

Um Sistema Gerenciador de Banco de Dados ou é o conjunto de programas de computador (softwares) responsáveis pelo gerenciamento de uma base de dados. O principal objetivo é retirar da aplicação cliente a responsabilidade de gerenciar o acesso, manipulação e organização dos dados. O SGBD disponibiliza uma interface para que os seus clientes possam incluir, alterar ou consultar dados. Em bancos de dados relacionais a interface é constituída pelas APIs ou drivers do SGBD, que executam comandos na linguagem SQL.

### 6.1. Linguagem SQL

É uma linguagem de pesquisa declarativa para banco de dados relacional (bases de dados relacionais). Muitas das características originais do SQL foram inspiradas na álgebra relacional. SQL é normalmente pronunciado em português como "esse-quê-ele", porém sua pronúncia correta deveria se "síquel", do inglês "sequel", ou "alguma coisa que segue outra coisa". SQL é uma brincadeira com o nome da primeira linguagem de consulta QUEL.

Embora o SQL tenha sido originalmente criado pela IBM, rapidamente surgiram vários "dialectos" desenvolvidos por outros produtores. Essa expansão levou à necessidade de ser criado e adaptado um padrão para a linguagem. Esta tarefa foi realizada pela American National Standards Institute (ANSI) em 1986 e ISO em 1987.

O SQL foi revisto em 1992 e a esta versão foi dado o nome de SQL-92. Foi revisto novamente em 1999 e 2003 para se tornar SQL:1999 (SQL3) e SQL:2003, respectivamente. O SQL:1999 usa expressões regulares de emparelhamento, queries recursivas e gatilhos (triggers). Também foi feita uma adição controversa de tipos não-escalados e algumas características de orientação a objeto. O SQL:2003 introduz características relacionadas ao XML, sequências padronizadas e colunas com valores de auto-generalização (inclusive colunas-identidade).

### 6.2. MySQL

O MySQL é um sistema de gerenciamento de banco de dados (SGBD), que utiliza a linguagem SQL (Structured Query Language - Linguagem de Consulta Estruturada) como interface. É atualmente um dos bancos de dados mais populares, com mais de 4 milhões de instalações pelo mundo.

#### 6.2.1. História

O MySQL foi criado na Suécia por dois Suecos e um finlandês: David Axmark, Allan Larsson e Michael "Monty" Widenius, que trabalham juntos desde a década de 1980. Hoje seu desenvolvimento e manutenção empregam aproximadamente 70 profissionais no mundo inteiro, e mais de mil contribuem testando o software, integrando-o a outros produtos, e escrevendo a respeito do mesmo.

***Nota:** O sucesso do MySQL deve-se em grande medida à fácil integração com o **PHP** incluído, quase que obrigatoriamente, nos pacotes de hospedagem de sites da Internet oferecidos atualmente. Empresas como Yahoo! Finance, MP3.com, Motorola, NASA, Silicon Graphics e Texas Instruments usam o MySQL em aplicações de missão crítica.*

O MySQL hoje suporta Unicode, Full Text Indexes, replicação, Hot Backup, GIS, OLAP e muitos outros recursos.

### **6.2.2. Características**

Principais características:

- Portabilidade (suporta praticamente qualquer plataforma atual)
- Compatibilidade (existem drivers ODBC, JDBC e .NET e módulos de interface para diversas linguagens de programação, como Java, C/C++, Python, Perl, PHP e Ruby)
- Excelente desempenho e estabilidade
- Pouco exigente quanto a recursos de hardware
- Facilidade de uso
- Suporte a vários tipos de tabelas (como MyISAM e InnoDB), cada um específico para um fim.

### **6.2.3. Vantagens**

- Outra grande vantagem é a de ter código aberto e funcionar em, quase, qualquer plataforma e sistema operacional : Windows, Linux, FreeBSD, BSDI, Solaris, Mac OS X, SunOS, SGI, etc.
- É reconhecido pelo seu desempenho e robustez e também por ser multi-tarefa e multi-usuário. A Wikipédia, usando o programa MediaWiki, utiliza o MySQL para gerenciar seu banco de dados, demonstrando que é possível utilizá-lo em sistemas de produção de alta exigência e em aplicações sofisticadas.
- No passado, devido a não possuir (até a versão 3.x) funcionalidades consideradas essenciais em muitas áreas, como stored procedures, two-fase commit, subselects, foreign keys ou integridade referencial, é frequentemente considerado um sistema mais "leve" e para aplicações menos exigentes, sendo preterido por outros sistemas como o PostgreSQL.

### **6.2.4. Notas**

- O MySQL a partir da versão 4.1 adicionou suporte a Transações, SubSelects, Foreign Keys e Integridade Referencial. Esse suporte foi graças ao database engine InnoDB.
- Com a versão 5.0, o MySQL incorporou mais recursos avançados ao sistema, incluindo views , triggers, storage procedures e transações XA.

## **6.3. A Linguagem SQL e seus componentes:**

### 6.3.1. DML - Linguagem de Manipulação de Dados

Primeiro há os elementos da DML (Data Manipulation Language - Linguagem de Manipulação de Dados). A DML é um subconjunto da linguagem usada para selecionar, inserir, atualizar e apagar dados.

- SELECT é o comumente mais usado do DML, comanda e permite ao usuário especificar uma query como uma descrição do resultado desejado. A questão não especifica como os resultados deveriam ser localizados.
- INSERT é usada para somar uma fila (formalmente uma tupla) a uma tabela existente.
- UPDATE para mudar os valores de dados em uma fila de tabela existente.
- DELETE permite remover filas existentes de uma tabela.
- BEGIN WORK (ou START TRANSACTION, dependendo do dialeto SQL) pode ser usado para marcar o começo de uma transação de banco de dados que pode ser completada ou não.
- COMMIT envia todos os dados das mudanças permanentemente.
- ROLLBACK faz com que as mudanças nos dados existentes desde que o último COMMIT ou ROLLBACK sejam descartadas.

COMMIT e ROLLBACK interagem com áreas de controle como transação e locação. Ambos terminam qualquer transação aberta e liberam qualquer cadeado ligado a dados. Na ausência de um BEGIN WORK ou uma declaração semelhante, a semântica de SQL é dependente da implementação.

### 6.3.2. DDL - Linguagem de Definição de Dados

O segundo grupo é a DDL (Data Definition Language - Linguagem de Definição de Dados). Uma DDL permite ao usuário definir tabelas novas e elementos associados. A maioria dos bancos de dados de SQL comerciais tem extensões proprietárias no DDL.

Os comandos básicos da DDL são:

- CREATE cria um objeto (uma Tabela, por exemplo) dentro da base de dados.
- DROP apaga um objeto do banco de dados.

Alguns sistemas de banco de dados usam o comando ALTER, que permite ao usuário alterar um objeto, por exemplo, adicionando uma coluna a uma tabela existente.

outros comandos DDL:

- ALTER TABLE
- CREATE INDEX
- ALTER INDEX
- DROP INDEX
- CREATE VIEW

- DROP VIEW

### 6.3.3. DCL - Linguagem de Controle de Dados

O terceiro grupo é o DCL (Data Control Language - Linguagem de Controle de Dados). DCL controla os aspectos de autorização de dados e licenças de usuários para controlar quem tem acesso para ver ou manipular dados dentro do banco de dados.

Duas palavras-chaves da DCL:

- GRANT - autoriza ao usuário executar ou setar operações.
- REVOKE - remove ou restringe a capacidade de um usuário de executar operações.

Outros comandos DCL:

- ALTER PASSWORD
- CREATE SYNONYM

### 6.3.4. DQL - Linguagem de Consulta de Dados

Embora tenha apenas um comando a DQL é a parte da SQL mais utilizada. O comando SELECT é composta de várias cláusulas e opções, possibilitando elaborar consultas das mais simples as mais elaboradas. Exemplos:

```
SELECT
    nome
FROM
    pessoas;

SELECT
    aP.codigo,
    aP.nome,
    aP.data_nascimento,
    aO.nome,
    aO.local
FROM
    pessoas aP,
    objetos aO,
WHERE
    aP.codigo = aO.codigo_pessoa and
    aP.codigo = (
        SELECT
            codigo_pessoa
        FROM
            catalogo
        WHERE
            cod_catalogo = 5
    );
```

Fonte: wikipedia.org

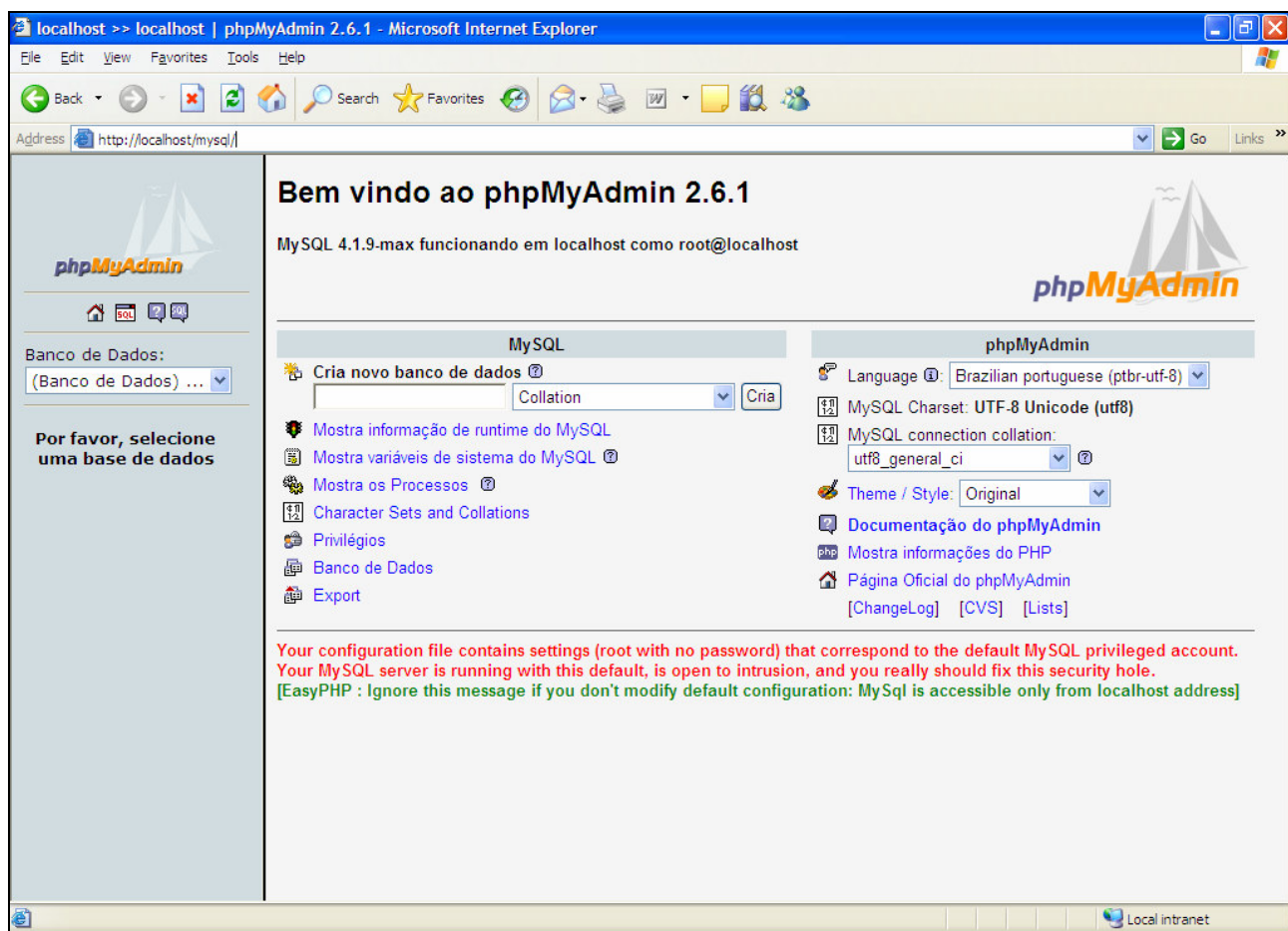


## 7. Utilizando o PHPmyadmin

### 7.1. phpMyAdmin

É um programa desenvolvido em php para administração do MySQL que roda na plataforma WEB. A partir deste sistema é possível criar e remover bases de dados, criar/remover/alterar tabelas, inserir/deletar/editar campos, executar códigos SQL e manipular campos chaves.

Como já visto, quando instalamos o EasyPHP ele instala automaticamente o phpMyAdmin, para acessá-lo digite o seguinte endereço no seu web browser: <http://localhost/mysql/>. Se tudo estiver correto deverá surgir no browser a página parecida como a próxima figura.



Agora que o phpMyAdmin está carregado, poderemos começar a criar nossa base de dados, mas antes, vamos conhecer os tipos de dados suportados pelo MySQL para elaborar nossas tabelas.

### 7.2. Tipos de dados mais comuns do MySQL

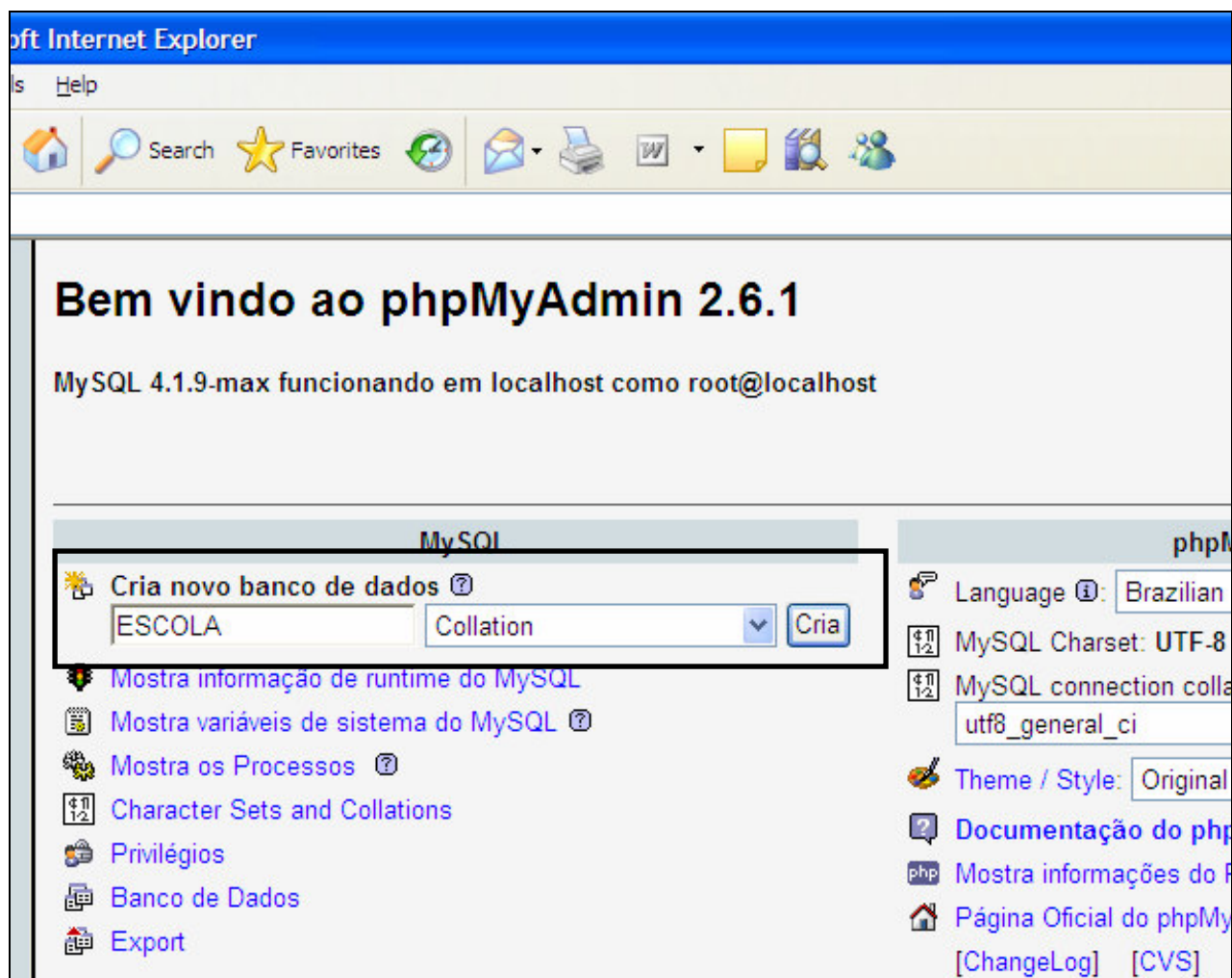
Tipo	Descrição
INT	Valor inteiro
REAL	Valor de ponto flutuante
CHAR(tamanho)	Valor de caractere de tamanho fixo. Valor inferior ao definido será

	deixado em branco.
TEXT(tamanho)	Valor de caractere de tamanho variável.
VARCHAR(tamanho)	Valor de caractere de tamanho variável. Valores inferiores ao definido serão suprimidos.
DATE	Valor para datas do tipo (AAAA-MM-DD)
TIME	Valor de tempo padrão
DATETIME	Valor para data e hora agregados

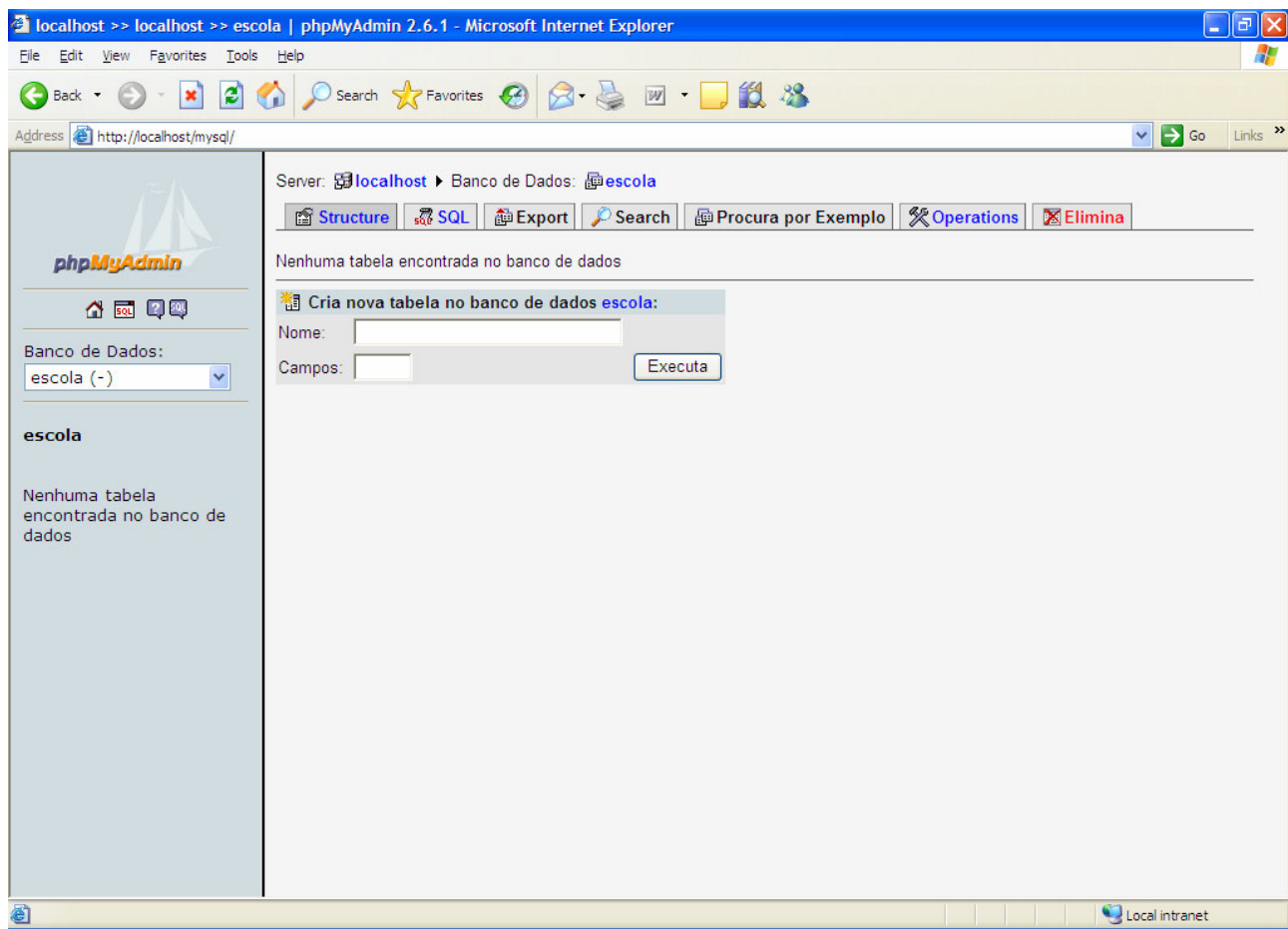
Agora que você sabe os tipos de dados suportados e a sintaxe de como criar uma tabela, é fundamental que você saiba o que deseja criar, isto é, como será a sua base de dados.

Iremos desenhar o modelo para a criação de uma base de dados simples, que seria o início do controle das informações de uma escola. Nossa base de dados será chamada de ESCOLA e nela teremos as tabelas ALUNOS, CURSOS e DISCIPLINAS. Vamos então ao nosso modelo.

1) Primeiro passo, criar o Banco de Dados como nome: ESCOLA, para tanto no phpMyAdmin clique digite ESCOLA no campo apropriado, veja a próxima figura:

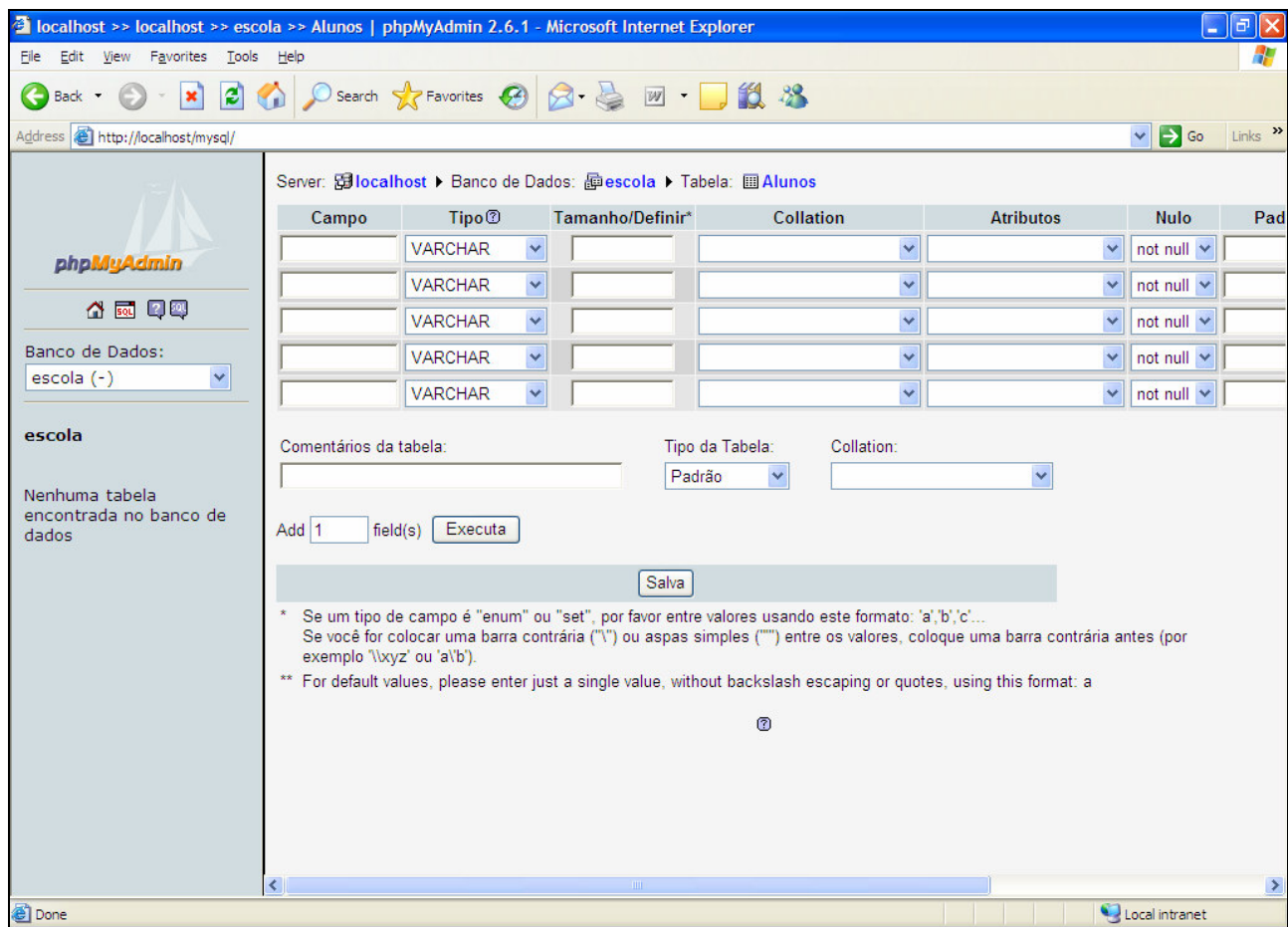


2) Depois que você executar [depois de criada] clique no botão atualizar do seu browser, e do lado esquerdo na seção de Banco de Dados do MySQL escolha o banco criado.

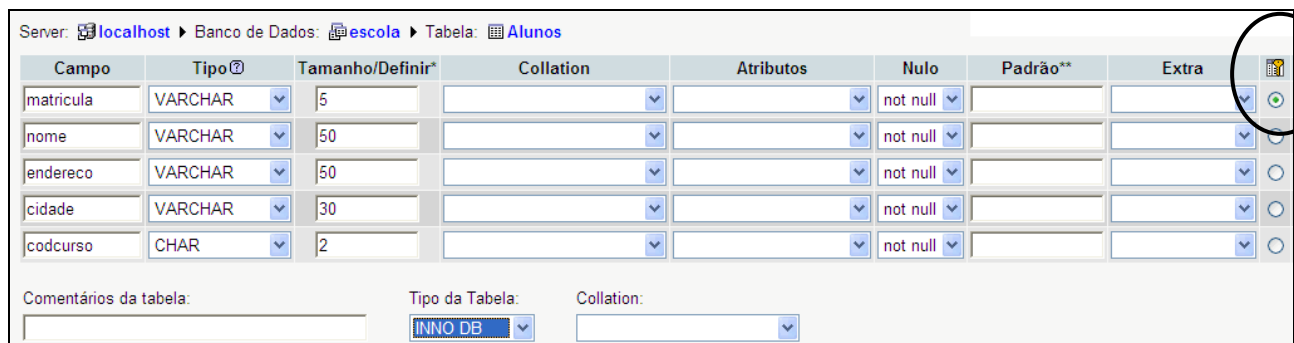


A figura acima mostra que o banco escola, ainda não possui nenhuma tabela, nesse momento iremos criar nossa primeira tabela chamada ALUNOS.

3) No campo nome digite: ALUNOS e no campo Campos digite: 5, e em seguida clique no botão executar, veja figura:



4) Nessa janela preencheremos a definição de cada campo, preencha para que fique exatamente igual à imagem abaixo e depois clique no botão [ SALVA ]. Campos: matricula, nome, endereço, cidade, codcurso.



Observe que o campo matricula é chave primária.

5) Vamos repetir o procedimento para criar a tabela CURSOS. Campos: codcurso, nome, coddisc1, coddisc2 e coddisc3.

Campo	Tipo
<input type="checkbox"/> codcurso	char(2)
<input type="checkbox"/> nome	varchar(50)
<input type="checkbox"/> coddisc1	char(2)
<input type="checkbox"/> coddisc2	char(2)
<input type="checkbox"/> coddisc3	char(2)

Obs: o campo codcurso deve ser chave primária da tabela.

6) Repetindo os procedimentos para criar a última tabela do nosso exemplo, ou seja, DISCIPLINAS, com os campos: coddisciplina, nomedisciplina.

	Campo	Tipo	Collation	Atr
<input type="checkbox"/>	coddisciplina	char(2)	latin1_swedish_ci	
<input type="checkbox"/>	nomedisciplina	varchar(30)	latin1_swedish_ci	

Obs: o campo coddisciplina deverá ser chave primária da tabela.

7) Se tudo ocorrer de forma correta, selecionando o seu banco a esquerda, aparecerá o nome das três tabelas criadas.



### 7.3. Explicando chave primária

Campo chave é o campo mais importante de nossa tabela, pois este é o campo que irá identificar a posição de todos os outros dados de um registro. Os dados deste campo são exclusivos, isto é, não poderão existir dois registros deste campo em sua tabela com o mesmo valor. Por isso, toda tabela deve ter um campo designado como chave primária para o controle dos registros.

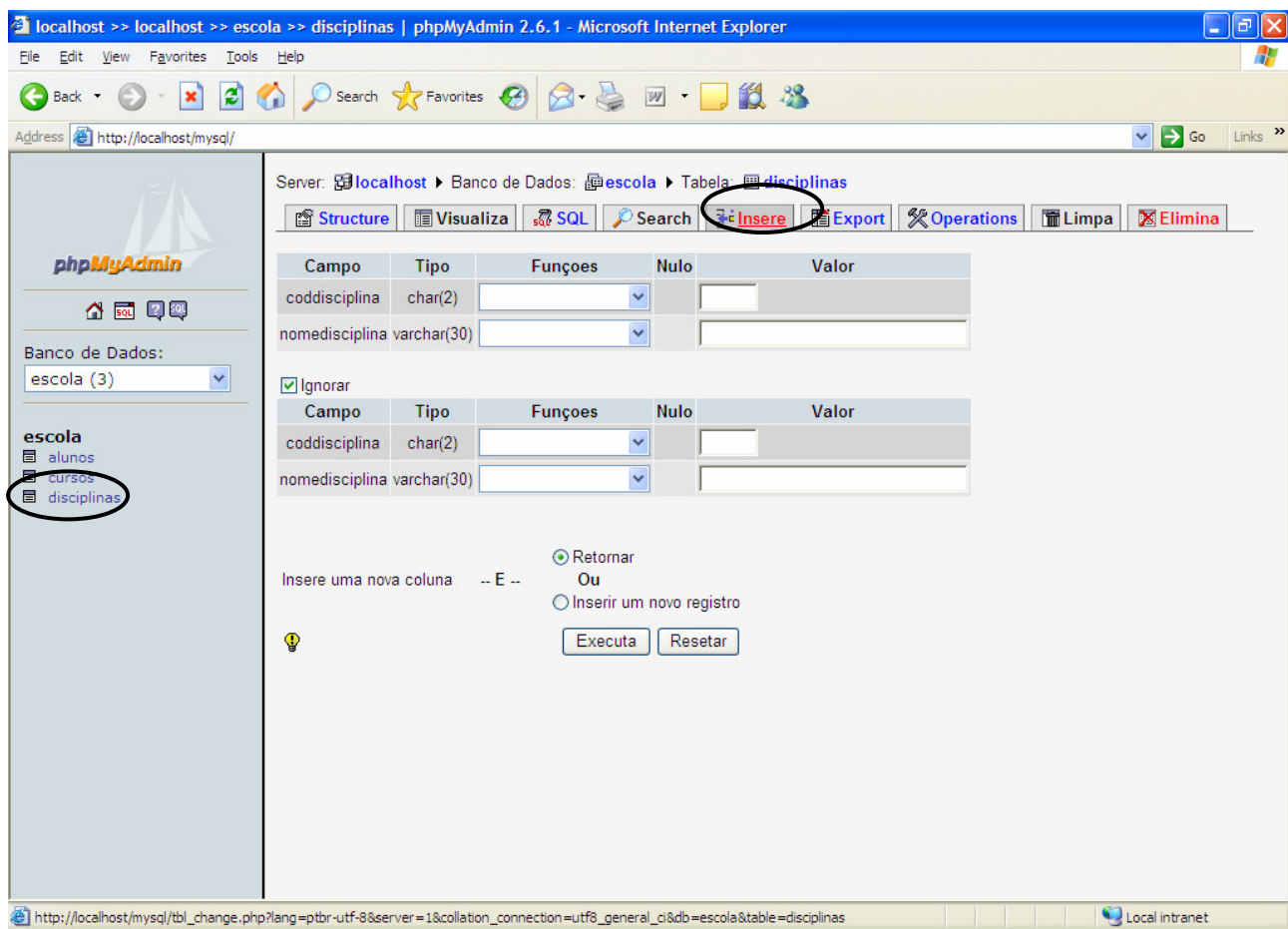
### 7.4. Manipulando dados nas tabelas

Agora que já temos a nossa estrutura de tabelas prontas, é hora de começar a inserir os dados, ou seja, popular as tabelas, pois é para isso que elas servem, armazenam dados para que estes sejam pesquisados mais tarde.

Utilizando o phpMyAdmin nesse momento para popular as tabelas, poderíamos utilizar a instrução SQL chamada INSERT INTO, no entanto para simplificar o uso, vamos digitar diretamente na tabela. É simples basta você selecionar a tabela clicando sobre o nome dela no lado esquerdo da tela, e em seguida clicar no link insere.

### 7.4.1. Digitando as disciplinas

Clique no nome da tabela disciplinas e em seguida no link insere, conforme figura:



No campo Valor na linha do coddisciplina digite “11”, em nome da disciplina digite “Banco de Dados”, escolha a opção inserir um novo registro, e repita o procedimento de inserção até que sua tabela tenha todos os registros abaixo:



coddisciplina	nomedisciplina
11	Banco de Dados
12	Lógica de Programação
13	Desenvolvimento de Software 1
21	Banco de Dados 2
22	Desenvolvimento de Software 2
23	Programação de Computadores 1
31	Banco de Dados 3
32	Programação de Computadores 2
33	Desenvolvimento de Software 3

Para visualizar os dados na tabela, clique no menu VISUALIZA.

#### 7.4.2. Digitando os cursos

Agora que você já sabe como inserir dados nas tabelas digite os dados para a tabela **CURSOS** conforme próxima figura:

codcurso	nome	coddisc1	coddisc2	coddisc3
01	Auxiliar de Informática	11	12	13
02	Programador de Computadores	21	22	23
03	Técnico em Informática	31	32	33

#### 7.4.3. Digitando os alunos

Repita o procedimento para popular a tabela **ALUNOS** conforme próxima figura:

matricula	nome	endereco	cidade	codcurso
10001	Marcos Moraes	Rua Pe Roque, 2057	Mogi Mirim	01
10002	Maria Conceição Lopes	Rua Araras, 23	Mogi Guaçu	01
10003	Ana Carina Lopes	Rua Peraltas, 222	Santos	01
10004	Carlos Aguiar	Rua Botafogo, 33	Santos	01
10005	André Luiz dos Santos	Rua Lopes Conte, 3343	Sapucaí	01
10006	Pedro Antonio Pimenta	Rua Altair Lopes, 33	Itapira	02
10007	Rita de Cássia da Silva	Rua Eletromais, 33	Itapira	02
10008	Caique dos Santos	Rua das Amoreiras, 55	Campinas	02
10009	Carlos Tavares	Rua Peixe, 99	Santos	02
10010	Antonio Carlos Caetano	Rua Josefina Alface, 987	Campinas	02
10011	Ricardo Moreira	Rua do Pinhal, 332	Aparecida	03
10012	Richardson S. P. Campeao	Rua do Tricolor, 433	São Paulo	03
10013	Junior Camisa Seis	Rua do Morumbi, 433	São Paulo	03
10014	Carina Melo	Rua Osvaldo Ramos, 88	Mogi Guaçu	03
10015	Pedro Mello	Rua Itororó, 3999	Mogi Mirim	03

## 7.5. Visualizando dados – Instrução SELECT

Não iria fazer sentido você armazenar dados se não pudesse visualizá-los quando fosse preciso, correto? Por isso, iremos aprender agora como visualizar estes dados. O comando a ser utilizado é o SELECT, veja a sintaxe dele abaixo:

```
SELECT {campo(s)} FROM {tabela(s)}
WHERE {condição}
ORDER BY {campo(s)}
GROUP BY {campo(s)}
```

Esclarecendo as informações da sintaxe a cima:

SELECT comando para se chamar a visualização de registros campo(s) campos da tabela, para referência da visualização dos registros.

FROM chamada para indicar a tabela ou tabelas a serem utilizadas.

WHERE chamada para uma condição que deve ser verdadeira para que os registros sejam visualizados.

ORDER BY permite ordenar a visualização de registros em função de um campo específico.

GROUP BY permite agrupar a visualização de registros em função de um campo específico

Obs: O "\*" (asterisco), terá um papel importante neste comando.



### 7.5.1. Exemplo: Consulta para pesquisar todos os alunos da cidade de Campinas.

- 1) Clique na tabela ALUNOS, e em seguida no link de menu SQL.
- 2) Digite a instrução abaixo no quadro “Fazer procura(s) SQL no banco de dados escola”.

```
SELECT *
FROM alunos
WHERE cidade = "Campinas"
```

- 3) Clique no botão executa, e você terá como resultado da pesquisa a figura a seguir:

Mostrando registros 0 - 1 (2 total, Query took 0.0005 sec)

comando SQL:  
 SELECT \*  
 FROM alunos  
 WHERE cidade = "Campinas"  
 LIMIT 0, 30

[Edita] [Explain SQL] [Create PHP Code] [Refresh]

Mostrar : 30 colunas começando de 0  
 no modo horizontal e repetindo cabeçalhos após 100 células

	matricula	nome	endereco	cidade	codcurso
<input type="checkbox"/> ?	10008	Caique dos Santos	Rua das Amoreiras, 55	Campinas	02
<input type="checkbox"/> ?	10010	Antonio Carlos Caetano	Rua Josefina Alfaca, 987	Campinas	02

↑ Marcar All / Desmarca Todos Com marcados:

Mostrar : 30 colunas começando de 0  
 no modo horizontal e repetindo cabeçalhos após 100 células

### 7.5.2. Exemplo: Consulta para pesquisar todos os alunos ordenados por ordem de curso.

- 1) Digite o seguinte comando SQL e execute

```
SELECT *
FROM alunos
ORDER BY codcurso
```

### 7.5.3. Outros exemplos de consulta

- 1) Pesquisar o nome do aluno de matrícula = 10008.

```
SELECT nome
FROM alunos
WHERE matricula = "10008"
```

2) Selecionar todos os dados dos alunos com número de matrícula maior que 10011.

```
SELECT *
FROM alunos
WHERE matricula >10011
```

3) Visualizar nome e endereço dos alunos que moram em São Paulo.

```
SELECT nome, endereco
FROM alunos
WHERE cidade = "São Paulo"
```

4) Visualizar todos os dados da tabela cursos.

```
SELECT *
FROM cursos
```

5) Visualizar todos os dados do curso de código = "01"

```
SELECT *
FROM cursos
WHERE codcurso = "01"
```

#### 7.5.4. Visualizando dados de mais de uma tabela

Para que isto ocorra, será fundamental o campo chave (chave primária), que ajudará a fazer a ligação com um campo comum de outra tabela (chave estrangeira), mas que tem o mesmo tipo de dado.

Sintaxe:

```
select TABELA1.campo, TABELA2.campo from TABELAS
where TABELA1.campo_chave = TABELA2.campo_comum
and TABELA.campo = valor;
```

Exemplo: Consulta que exhibe todos os alunos que freqüentam cursos que freqüentam o curso de Auxiliar de Informática.

```
SELECT alunos . *, cursos.nome
FROM alunos, cursos
WHERE cursos.codcurso = alunos.codcurso
AND cursos.nome = "Auxiliar de Informática"
ORDER BY alunos.nome
```

Resultado da consulta:

matricula	nome ▲	endereco	cidade	codcurso	nome
10003	Ana Carina Lopes	Rua Peraltas, 222	Santos	01	Auxiliar de Informática
10005	André Luiz dos Santos	Rua Lopes Conte, 3343	Sapucaí	01	Auxiliar de Informática
10004	Carlos Aguiar	Rua Botafogo, 33	Santos	01	Auxiliar de Informática
10001	Marcos Moraes	Rua Pe Roque, 2057	Mogi Mirim	01	Auxiliar de Informática
10002	Maria Conceição Lopes	Rua Araras, 23	Mogi Guaçu	01	Auxiliar de Informática

## 7.6. Alterando dados

Você poder alterar uma informação através do comando UPDATE, veja a sintaxe abaixo:

```
UPDATE tabela
SET campo = valor
WHERE {condição};
```

Vamos a uma explicação prática desta sintaxe:

O exemplo abaixo atualiza a tabela alunos, altere a cidade para “Mogi Mirim” quando a matrícula for igual a “10002”.

```
UPDATE alunos
SET cidade="Mogi Mirim"
WHERE matricula="10002"
```

Executando essa consulta a cidade será alterada para Mogi Mirim, quando o MySQL encontrar uma ocorrência de matrícula igual a “10002”. Visualize a tabela depois de alterado para observar a atualização.

10002	Maria Conceição Lopes	Rua Araras, 23	Mogi Mirim	01
-------	-----------------------	----------------	------------	----

### Excluindo Dados

Para excluir uma informação do Banco de Dados, utilizamos o comando DELETE, veja a sintaxe a seguir:

```
DELETE FROM tabela
WHERE condição;
```

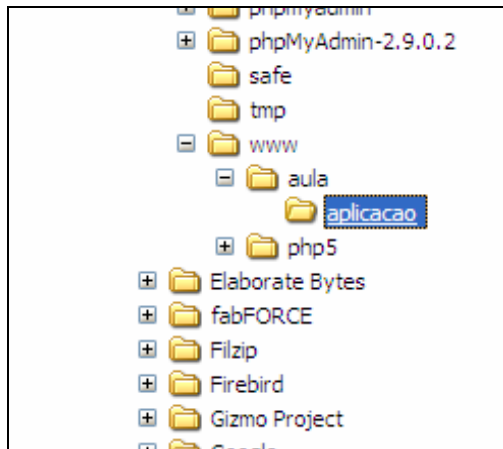
Exluindo o aluno de Matrícula 10005

```
DELETE FROM alunos
WHERE matricula="10001"
```

OBS: indique sempre a condição para evitar possíveis desastres. ☺

## 8. Fazendo o PHP se comunicar com o MySQL

Vamos criar agora os scripts que farão a comunicação com o banco de dados. Primeiramente crie uma subpasta dentro da pasta aula chamada **aplicacao**.



### 8.1. Páginas HTML do Sistema de Cadastros

Nosso objetivo é criar um pequeno sistema de gerenciamento de escola, o ponto inicial será a criação de um arquivo tipo html que será a página principal da aplicação que chamaremos de index.html.

**Baixe a figura do logo no endereço:**

<http://www.maromo.pro.br/modules.php?name=Downloads&op=getit&lid=32>

**Ou**

<http://www.maromo.pro.br/modules/Downloads/cursos/sislogo.gif>

1) No **PHP Editor** inicie um novo projeto HTML, digite o código abaixo:

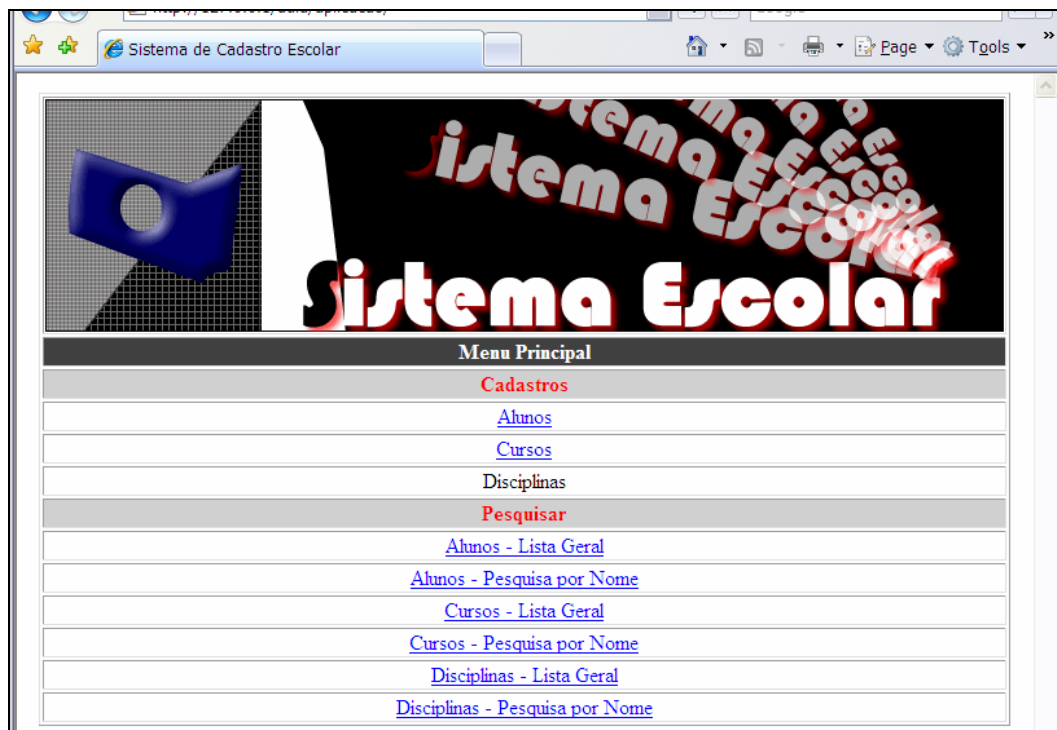
```
<HTML>
<HEAD>
  <TITLE>Sistema de Cadastro Escolar</TITLE>
</HEAD>
<BODY>
<p align="center"><table border="1" width=740>
<tr>
  <td><p align="center"></p></td>
</tr>
<tr>
  <td bgcolor=#404040><p align="center"><b><font color=#FFFFFF>Menu
Principal</font></b></p></td>
</tr>
<tr>
  <td bgcolor=#D0D0D0><p align="center"><b><font
color="#FF0000">Cadastros</font></b></p></td>
</tr>
<tr>
  <td><p align="center"><a href="cadaluno.html">Alunos</a></p></td>
```

```

</tr>
<tr>
  <td><p align="center"><a href="cadcurso.html">Cursos</a></p></td>
</tr>
<tr>
  <td><p align="center"><a href="caddisiplina.html">Disciplinas</a></p></td>
</tr>
<tr>
  <td bgcolor=#D0D0D0><p align="center"><b><font
color="#FF0000">Pesquisar</font></b></p></td>
</tr>
<tr>
  <td><p align="center"><a href="pesqalu_geral.php">Alunos - Lista
Geral</a></p></td>
</tr>
<tr>
  <td><p align="center"><a href="pesqalu_nome.html">Alunos - Pesquisa por
Nome</a></p></td>
</tr>
<tr>
  <td><p align="center"><a href="pesqcur_geral.php">Cursos - Lista
Geral</a></p></td>
</tr>
<tr>
  <td><p align="center"><a href="pesqcur_nome.html">Cursos - Pesquisa por
Nome</a></p></td>
</tr>
<tr>
  <td><p align="center"><a href="pesqdisc_geral.php">Disciplinas - Lista
Geral</a></p></td>
</tr>
<tr>
  <td><p align="center"><a href="pesqdisc_nome.html">Disciplinas - Pesquisa por
Nome</a></p></td>
</tr>
</table></p>
</BODY>
</HTML>

```

2) Salve o arquivo como index.html – Nossa página principal deverá ficar igual a figura abaixo:



Agora vamos criar as páginas html restantes para os cadastros.

Vamos começar pelo cadastro de alunos. Página **cadaluno.html**

3) No **PHP Editor** inicie um novo projeto HTML, digite o código abaixo:

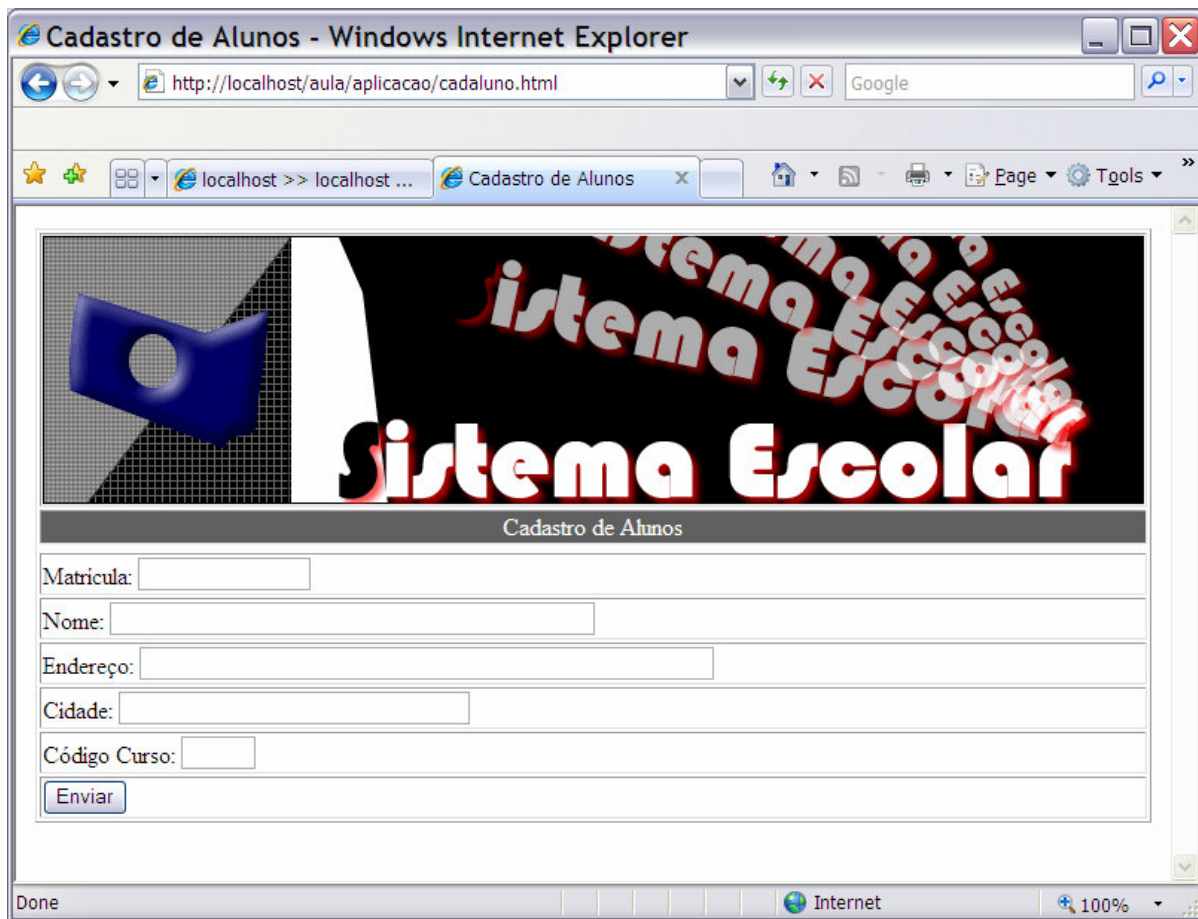
```
<HTML>
<HEAD>
  <TITLE>Cadastro de Alunos</TITLE>
</HEAD>
<BODY>
<p align="center"><table border="1" width="740">
<tr>
  <td></td>
</tr>
<tr>
  <td bgcolor=#606060><p align="center"><font color="#FFFFFF">Cadastro de
Alunos</font></p></td>
</tr>
<tr>
  <td><form action="cadaluno.php" method="POST">
    <tr>
      <td>Matrícula: <input type="text" size="15" name="txtMatricula"></td>
</tr>
    <tr>
      <td>Nome: <input type="text" size="50" name="txtNome"></td>
</tr>
    <tr>
      <td>Endereço: <input type="text" size="60" name="txtEndereco"></td>
</tr>
    <tr>
      <td>Cidade: <input type="text" size="35" name="txtCidade"></td>
</tr>
    <tr>
      <td>Código Curso: <input type="text" size="04" name="txtCodcurso"></td>
```

```

</tr>
<tr>
    <td><input type="submit" value="Enviar"></td>
</tr>
</form></td>
</tr>
</table></p>
</BODY>
</HTML>

```

4) Salve o arquivo como cadaluno.html – Nossa página de cadastro de alunos deverá ficar igual a figura abaixo:



Vamos agora criar a página de cadastro de curso.

5) No **PHP Editor** inicie um novo projeto HTML, digite o código abaixo:

```

<HTML>
<HEAD>
    <TITLE>Cadastro de Cursos</TITLE>
</HEAD>
<BODY>
<p align="center"><table border="1" width="740">
<tr>
    <td></td>
</tr>
<tr>

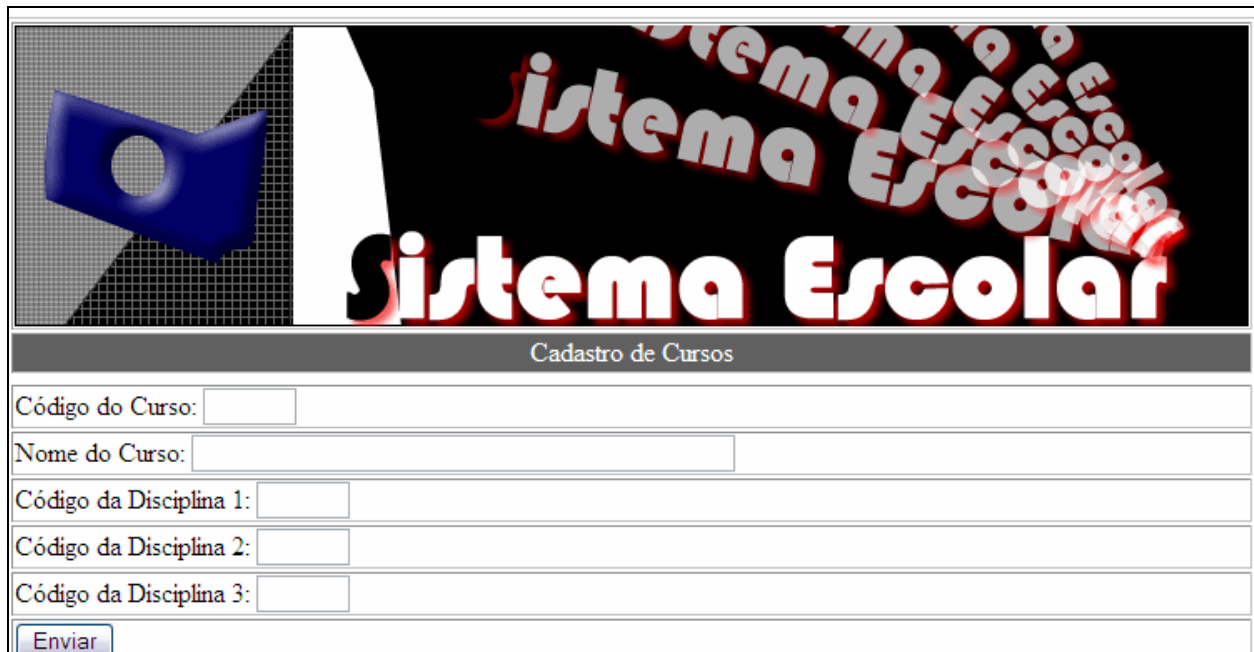
```

```

<td bgcolor=#606060><p align="center"><font color="#FFFFFF">Cadastro de
Cursos</font></p></td>
</tr>
<tr>
<td><form action="cadcurso.php" method="POST">
<tr>
<td>Código do Curso: <input type="text" size="05"
name="txtCodcurso"></td>
</tr>
<tr>
<td>Nome do Curso: <input type="text" size="50" name="txtNome"></td>
</tr>
<tr>
<td>Código da Disciplina 1: <input type="text" size="05"
name="txtCoddisc1"></td>
</tr>
<tr>
<td>Código da Disciplina 2: <input type="text" size="05"
name="txtCoddisc2"></td>
</tr>
<tr>
<td>Código da Disciplina 3: <input type="text" size="05"
name="txtCoddisc3"></td>
</tr>
<tr>
<td><input type="submit" value="Enviar"></td>
</tr>
</form></td>
</tr>
</table></p>
</BODY>
</HTML>

```

6) Salve o arquivo como cadcurso.html – Nossa página de cadastro de alunos deverá ficar igual a figura abaixo:



The screenshot shows a web browser window with a header banner for 'Sistema Escolar'. Below the banner is a form titled 'Cadastro de Cursos'. The form contains the following fields:

- Código do Curso:
- Nome do Curso:
- Código da Disciplina 1:
- Código da Disciplina 2:
- Código da Disciplina 3:
- Enviar:



Vamos agora criar a página de cadastro de disciplinas.

7) No **PHP Editor** inicie um novo projeto HTML, digite o código abaixo:

```
<HTML>
<HEAD>
  <TITLE>Cadastro de Disciplinas</TITLE>
</HEAD>
<BODY>
<p align="center"><table border="1" width="740">
<tr>
  <td></td>
</tr>
<tr>
  <td bgcolor=#606060><p align="center"><font color="#FFFFFF">Cadastro de
Disciplinas</font></p></td>
</tr>
<tr>
  <td><form action="caddisciplina.php" method="POST">
    <tr>
      <td>Código da Disciplina: <input type="text" size="05"
name="txtCoddisciplina"></td>
    </tr>
    <tr>
      <td>Nome da Disciplina: <input type="text" size="50"
name="txtNome"></td>
    </tr>
    <tr>
      <td><input type="submit" value="Enviar"></td>
    </tr>
  </form></td>
</tr>
</table></p>
</BODY>
</HTML>
```

8) Salve o arquivo como caddisciplina.html – Nossa página de cadastro de disciplinas deverá ficar igual a figura abaixo:



The screenshot shows a web browser window displaying a form titled 'Cadastro de Disciplinas'. The form has a header with a blue logo and the text 'Sistema Escolar'. Below the header, there are two input fields: 'Código da Disciplina:' and 'Nome da Disciplina:'. At the bottom of the form, there is a button labeled 'Enviar'.

## 8.2. Scripts PHP de configurações

Para que possamos realizar os cadastros e as pesquisas no Banco de Dados Escola, precisamos inicialmente realizar a conexão com o Banco propriamente dito.

1) Inicie um novo projeto PHP no PHP Editor e após digitar o código abaixo, salve com o nome **config.php** dentro da pasta **aplicacao**.

```
<HTML>
<HEAD>
  <TITLE>Conexão com o BD Escola</TITLE>
</HEAD>
<BODY>
<?PHP
function conectar()
{
    $hostdb = 'localhost'; //host em que se dará a transação, pode ser o nome ou o
                        //endereço ip
    $db = 'escola'; //nome do banco de dados
    $userdb = 'root'; //usuário que terá o acesso
    $passdb = ''; //senha do usuário

    if ($con = mysql_pconnect($hostdb,$userdb,$passdb))
    {
        return $con; //se a conexão for bem sucedida, será retornado a
variável $con
    }
    else
    {
        return 0; //se a conexão não ocorrer, será retornado 0
    }
}
?>
</BODY>
</HTML>
```

NOTA: Este script será utilizado pelos demais scripts que necessitaram se comunicar com o banco de dados.

Agora nosso próximo passo é criar um script com uma função que executa um comando SQL no Banco de Dados MySQL.

2) Inicie um novo projeto PHP no PHP Editor e após digitar o código abaixo, salve com o nome **mysqlexecuta.php** dentro da pasta **aplicacao**.

```
<?php
/*
$id - Ponteiro da Conexão
$sql - Cláusula SQL a executar
$erro - Especifica se a função exibe ou não(0=não, 1=sim)
$res - Resposta
*/
function mysqlexecuta($id,$sql,$erro = 1) {
    if(empty($sql) OR !($id))
        return 0; //Erro na conexão ou no comando SQL
    if (!($res = @mysql_query($sql,$id))) {
```

```

        if($erro)
            echo "Ocorreu um erro na execução do Comando SQL no banco de dados.
Favor Contactar o Administrador.";
        exit;
    }
    return $res;
}
?>

```

### 8.3. Scripts de envio de Dados dos Cadastros

Agora criaremos 03 novos script para envio de dados dos cadastros anteriormente criados, ou seja, **Alunos, Cursos e Disciplinas** ao nosso banco de dados escola.

Nota: Observe que o form action de cada um dos arquivos htmls criados para os cadastros apontam para os scripts phps que criaremos nesse instante.

1) Inicie um novo projeto PHP no PHP Editor e após digitar o código abaixo, salve com o nome **cadaluno.php** dentro da pasta **aplicacao**.

```

<HTML>
<HEAD>
    <TITLE>Recebe Dados - Insere no Banco</TITLE>
</HEAD>
<BODY>
<?php
$matricula= $_POST["txtMatricula"];
$nome =$_POST["txtNome"];
$endereco =$_POST["txtEndereco"];
$cidade =$_POST["txtCidade"];
$codcurso =$_POST["txtCodcurso"];
include 'config.php';
include 'mysqlexecuta.php'; //para executar o script no mysql
$con = conectar();
mysql_select_db('ESCOLA');
$sql = "INSERT INTO alunos(matricula, nome, endereco, cidade,
codcurso)Values('$matricula','$nome','$endereco','$cidade','$codcurso')";
$res = mysqlexecuta($con,$sql);
echo "Inclusão efetuada com sucesso!!!";
echo "<p align='center'><a href='index.html'>Menu Principal</a>";
?>
</BODY>
</HTML>

```

Alguma explicações: o comando `$_POST` representa uma variável ‘superglobal’. Significa que ela está disponível em todos os níveis (escopo) de um script. Já a instrução `include`: inclui e avalia o arquivo informado, nesse caso são dois arquivos de configuração o conexão ao Banco de Dados Escola (`config.php`) e o de execução de sentença SQL de inclusão de dados “Insert Into” no Mysql (`mysqlexecuta.php`).

2) Inicie um novo projeto PHP no PHP Editor e após digitar o código abaixo, salve com o nome **cadcurso.php** dentro da pasta **aplicacao**.

```

<HTML>

```

```

<HEAD>
  <TITLE>Recebe Dados - Insere no Banco</TITLE>
</HEAD>
<BODY>
<?php
$codcurso= $_POST["txtCodcurso"];
$nome =$_POST["txtNome"];
$disc1 =$_POST["txtCoddisc1"];
$disc2 =$_POST["txtCoddisc2"];
$disc3 =$_POST["txtCoddisc3"];
include 'config.php';
include 'mysqlexecuta.php'; //para executar o script no mysql
$con = conectar();
mysql_select_db('ESCOLA');
$sql = "INSERT INTO cursos(codcurso, nome, coddisc1, coddisc2,
coddisc3)Values('$codcurso','$nome','$disc1','$disc2','$disc3')";
$res = mysql_executa($con,$sql);
//confirmar inclusão de dados
echo "Inclusão efetuada com sucesso!!!";
echo "<p align='center'><a href='index.html'>Menu Principal</a>";
?>
</BODY>
</HTML>

```

3) Inicie um novo projeto PHP no PHP Editor e após digitar o código abaixo, salve com o nome **caddisplina.php** dentro da pasta **aplicacao**.

```

<HTML>
<HEAD>
  <TITLE>Recebe Dados - Insere no Banco</TITLE>
</HEAD>
<BODY>
<?php
$coddisc= $_POST["txtCoddisciplina"];
$nome =$_POST["txtNome"];
include 'config.php';
include 'mysqlexecuta.php'; //para executar o script no mysql
$con = conectar();
mysql_select_db('ESCOLA');
$sql = "INSERT INTO disciplinas(coddisciplina,
nomedisciplina)Values('$coddisc','$nome')";
$res = mysql_executa($con,$sql);
echo "Inclusão efetuada com sucesso!!!";
echo "<p align='center'><a href='index.html'>Menu Principal</a>";
?>
</BODY>
</HTML>

```

## 8.4. Páginas HTML do Sistema de Pesquisas

Vamos criar as páginas HTML necessárias para o nosso sistema de pesquisa por critérios de nome. Faremos isso para as três tabelas (Alunos, Cursos e Disciplinas).

1) Inicie um novo projeto HTML no PHP Editor e após digitar o código abaixo, salve com o nome **pesqalu\_nome.html** dentro da pasta **aplicacao**.

```

<HTML>
<HEAD>

```

```

<TITLE>Pesquisar Alunos</TITLE>
</HEAD>
<BODY>
<form action="pesqalu_nome.php" method="POST">
<p align="center"><table border="1" width="740">
<tr>
<td></td>
</tr>
<tr>
<td bgcolor=#FFFFC0><p align="center"><b>Pesquisa Dados de Aluno</b></p></td>
</tr>
<tr>
<td>Nome do Aluno: <input type="text" size="50" name="txtNome"></td>
</tr>
<tr>
<td><input type="submit" value="Buscar" name="btnEnviar"></td>
</tr>
</table>
</form>
</BODY>
</HTML>

```

2) Inicie um novo projeto HTML no PHP Editor e após digitar o código abaixo, salve com o nome **pesqcur\_nome.html** dentro da pasta **aplicacao**.

```

<HTML>
<HEAD>
<TITLE>Pesquisar Cursos</TITLE>
</HEAD>
<BODY>
<form action="pesqcur_nome.php" method="POST">
<p align="center"><table border="1" width="740">
<tr>
<td></td>
</tr>
<tr>
<td bgcolor=#FFFFC0><p align="center"><b>Pesquisa Dados de Cursos</b></p></td>
</tr>
<tr>
<td>Nome do Curso: <input type="text" size="50" name="txtNome"></td>
</tr>
<tr>
<td><input type="submit" value="Buscar" name="btnEnviar"></td>
</tr>
</table>
</form>
</BODY>
</HTML>

```

3) Inicie um novo projeto HTML no PHP Editor e após digitar o código abaixo, salve com o nome **pesqdisc\_nome.html** dentro da pasta **aplicacao**.

```

<HTML>
<HEAD>
<TITLE>Listar Dados das Disciplinas</TITLE>
</HEAD>
<BODY>
<?php

```

```

include 'config.php'; //para logar no Banco de Dados
include 'mysqlexecuta.php'; //para executar o script no mysql
$con = conectar();
mysql_select_db('ESCOLA');
$nome= $_POST["txtNome"];
$sql="SELECT * FROM disciplinas where nomedisciplina like '%$nome%' order by
nomedisciplina";
$res = mysql_executa($con,$sql);
?>
<p align="center"><table border="1" width="740">
<tr>
    <td></td>
</tr>
<tr>
    <td bgcolor=#FFFFC0><p align="center"><b>Disciplinas em Ordem
Alfabética</b></p></td>
</tr>
<table width=750 cellpadding=0 cellspacing=0 border=1>
<?php
    //Exibe as linhas encontradas na consulta
    while ($row = mysql_fetch_array($res)) {
?>

    <tr>
        <td><?php echo $row['coddisciplina'];?></td>
        <td><?php echo $row['nomedisciplina'];?></td>
    </tr>

<?php
}
?>
</table>
<p> <p>
<a href='index.html'><p align="center">Voltar para Menu Principal</p></a>
</BODY>
</HTML>

```

## 8.5. Script PHP do Sistema de Pesquisa

Abaixo os 06 scripts necessários para a realização de todas as consultas.

1) Começamos com os dois scripts para pesquisar dados de alunos.

1-a) Inicie um novo projeto PHP no PHP Editor, digite o código abaixo e salve como **pesqalu\_nome.php**.

```

<HTML>
<HEAD>
    <TITLE>Listar Dados dos Alunos</TITLE>
</HEAD>
<BODY>
<?php
include 'config.php'; //para logar no Banco de Dados
include 'mysqlexecuta.php'; //para executar o script no mysql
$con = conectar();
mysql_select_db('ESCOLA');

```

```

$nome= $_POST["txtNome"];
$sql="SELECT * FROM alunos where nome like '%$nome%' order by nome";
$res = mysql_executa($con,$sql);
?>
<p align="center"><table border="1" width="740">
<tr>
<td></td>
</tr>
<tr>
<td bgcolor=#FFFFC0><p align="center"><b>Alunos em Ordem
Alfabética</b></p></td>
</tr>
<table width=750 cellpadding=0 cellspacing=0 border=1>
<tr>
<td><b>Matrícula</b></td>
<td><b>Nome do Aluno</b></td>
<td><b>Endereço</b></td>
<td><b>Cidade</b></td>
<td><b>Cód. Curso</b></td>
</tr>
<?php
//Exibe as linhas encontradas na consulta
while ($row = mysql_fetch_array($res)) {
?>

<tr>
<td><?php echo $row['matricula'];?></td>
<td><?php echo $row['nome'];?></td>
<td><?php echo $row['endereco'];?></td>
<td><?php echo $row['cidade'];?></td>
<td><?php echo $row['codcurso'];?></td>
</tr>

<?php
}
?>
</table>
<p> <p>
<a href='index.html'><p align="center">Voltar para Menu Principal</p></a>
</BODY>
</HTML>

```

O script acima recebe o resultado da pesquisa de dados de aluno enviada pelo arquivo `pesq_alu.html`.

**NOTA:** O comando "LIKE '%\$nome%'" faz uma busca sobre qualquer parte do nome do aluno, sendo que neste caso ele irá retornar os registros cujo nome case com a string digitada.

A variável "\$res", por sua vez, irá receber todos os dados encontrados nesta conexão.

A função "mysql\_fetch\_array", nativa do PHP, retorna uma matriz que corresponde a linha buscada, ou FALSE se não houver linhas.

A função "mysql\_num\_rows" informa quantos registros foram encontrados pela pesquisa realizada.

1-b) Inicie um novo projeto PHP no PHP Editor, digite o código abaixo e salve como `pesqalu_geral.php`

```
<HTML>
<HEAD>
  <TITLE>Listar Dados dos Alunos</TITLE>
</HEAD>
<BODY>
<?php
include 'config.php'; //para logar no Banco de Dados
include 'mysqlexecuta.php'; //para executar o script no mysql
$con = conectar();
mysql_select_db('ESCOLA');
$sql="SELECT * FROM alunos order by nome";
$res = mysqlexecuta($con,$sql);
?>
<p align="center"><table border="1" width="740">
<tr>
  <td></td>
</tr>
<tr>
  <td bgcolor=#FFFFC0><p align="center"><b>Alunos em Ordem
Alfabética</b></p></td>
</tr>
<table width=750 cellpadding=0 cellspacing=0 border=1>
<tr>
  <td><b>Matrícula</b></td>
  <td><b>Nome do Aluno</b></td>
  <td><b>Endereço</b></td>
  <td><b>Cidade</b></td>
  <td><b>Cód. Curso</b></td>
</tr>

<?php
  //Exibe as linhas encontradas na consulta
  while ($row = mysql_fetch_array($res)) {
    ?>

    <tr>
      <td><?php echo $row['matricula'];?></td>
      <td><?php echo $row['nome'];?></td>
      <td><?php echo $row['endereco'];?></td>
      <td><?php echo $row['cidade'];?></td>
      <td><?php echo $row['codcurso'];?></td>
    </tr>

<?php
  }
?>
</table>
<p> <p>
<a href='index.html'><p align="center">Voltar para Menu Principal</p></a>
</BODY>
</HTML>
```

O script acima exibe todos os dados da tabela de aluno sem filtro.

2) Agora vamos digitar os dois scripts para pesquisar dados de cursos.



2-a) Inicie um novo projeto PHP no PHP Editor, digite o código abaixo e salve como **pesqcur\_nome.php**.

```
<HTML>
<HEAD>
  <TITLE>Listar Dados dos Cursos</TITLE>
</HEAD>
<BODY>
<?php
include 'config.php'; //para logar no Banco de Dados
include 'mysqlexecuta.php'; //para executar o script no mysql
$con = conectar();
mysql_select_db('ESCOLA');
$nome= $_POST["txtNome"];
$sql="SELECT * FROM cursos where nome like '%$nome%' order by nome";
$res = mysql_executa($con,$sql);
?>
<p align="center"><table border="1" width="740">
<tr>
  <td></td>
</tr>
<tr>
  <td bgcolor=#FFFFC0><p align="center"><b>Cursos em Ordem
Alfabética</b></p></td>
</tr>
<table width=750 cellpadding=0 cellspacing=0 border=1>
<tr>
  <td><b>Cód. Curso</b></td>
  <td><b>Nome do Curso</b></td>
  <td><b>Disc 1</b></td>
  <td><b>Disc 2</b></td>
  <td><b>Disc 3</b></td>
</tr>
<?php
  //Exibe as linhas encontradas na consulta
  while ($row = mysql_fetch_array($res)) {
?>

    <tr>
      <td><?php echo $row['codcurso'];?></td>
      <td><?php echo $row['nome'];?></td>
      <td><?php echo $row['coddisc1'];?></td>
      <td><?php echo $row['coddisc2'];?></td>
      <td><?php echo $row['coddisc3'];?></td>
    </tr>

  <?php
  }
?>
</table>
<p> <p>
<a href='index.html'><p align="center">Voltar para Menu Principal</p></a>
</BODY>
</HTML>
```

2-b) Inicie um novo projeto PHP no PHP Editor, digite o código abaixo e salve como **pesqcur\_geral.php**

```
<HTML>
<HEAD>
```

```

<TITLE>Listar Dados dos Cursos</TITLE>
</HEAD>
<BODY>
<?php
include 'config.php'; //para logar no Banco de Dados
include 'mysqlexecuta.php'; //para executar o script no mysql
$con = conectar();
mysql_select_db('ESCOLA');
$sql="SELECT * FROM cursos order by nome";
$res = mysql_executa($con,$sql);
?>
<p align="center"><table border="1" width="740">
<tr>
<td></td>
</tr>
<tr>
<td bgcolor=#FFFFC0><p align="center"><b>Cursos em Ordem
Alfabética</b></p></td>
</tr>
<table width=750 cellpadding=0 cellspacing=0 border=1>
<tr>
<td><b>Cód. Curso</b></td>
<td><b>Nome do Curso</b></td>
<td><b>Disc 1</b></td>
<td><b>Disc 2</b></td>
<td><b>Disc 3</b></td>
</tr>
<?php
//Exibe as linhas encontradas na consulta
while ($row = mysql_fetch_array($res)) {
?>

<tr>
<td><?php echo $row['codcurso'];?></td>
<td><?php echo $row['nome'];?></td>
<td><?php echo $row['coddisc1'];?></td>
<td><?php echo $row['coddisc2'];?></td>
<td><?php echo $row['coddisc3'];?></td>
</tr>

<?php
}
?>
</table>
<p> <p>
<a href='index.html'><p align="center">Voltar para Menu Principal</p></a>
</BODY>
</HTML>

```

3) Agora para encerrar, vamos digitar os dois últimos scripts, para a pesquisa específica e geral de disciplinas cadastradas.

3-a) Inicie um novo projeto PHP no PHP Editor, digite o código abaixo e salve como **pesqdisc\_nome.php**

```

<HTML>
<HEAD>
<TITLE>Listar Dados das Disciplinas</TITLE>
</HEAD>
<BODY>

```

```

<?php
include 'config.php'; //para logar no Banco de Dados
include 'mysqlexecuta.php'; //para executar o script no mysql
$con = conectar();
mysql_select_db('ESCOLA');
$nome= $_POST["txtNome"];
$sql="SELECT * FROM disciplinas where nomedisciplina like '%$nome%' order by
nomedisciplina";
$res = mysql_executa($con,$sql);
?>
<p align="center"><table border="1" width="740">
<tr>
<td></td>
</tr>
<tr>
<td bgcolor=#FFFFC0><p align="center"><b>Disciplinas em Ordem
Alfabética</b></p></td>
</tr>
<table width=750 cellpadding=0 cellspacing=0 border=1>
<tr>
<td><b>Cód. Disc.</b></td>
<td><b>Nome da Disciplina</b></td>
</tr>
<?php
//Exibe as linhas encontradas na consulta
while ($row = mysql_fetch_array($res)) {
?>

<tr>
<td><?php echo $row['coddisciplina'];?></td>
<td><?php echo $row['nomedisciplina'];?></td>
</tr>

<?php
}
?>
</table>
<p> <p>
<a href='index.html'><p align="center">Voltar para Menu Principal</p></a>
</BODY>
</HTML>

```

3-b) Inicie um novo projeto PHP no PHP Editor, digite o código abaixo e salve como **pesqdisc\_geral.php**

```

<HTML>
<HEAD>
<TITLE>Listar Dados das Disciplinas</TITLE>
</HEAD>
<BODY>
<?php
include 'config.php'; //para logar no Banco de Dados
include 'mysqlexecuta.php'; //para executar o script no mysql
$con = conectar();
mysql_select_db('ESCOLA');
$sql="SELECT * FROM disciplinas order by nomedisciplina";
$res = mysql_executa($con,$sql);
?>
<p align="center"><table border="1" width="740">
<tr>
<td></td>

```

```

</tr>
<tr>
  <td bgcolor=#FFFFC0><p align="center"><b>Disciplinas em Ordem
Alfabética</b></p></td>
</tr>
<table width=750 cellpadding=0 cellspacing=0 border=1>
<tr>
  <td><b>Cód. Disc.</b></td>
  <td><b>Nome da Disciplina</b></td>
</tr>
<?php
  //Exibe as linhas encontradas na consulta
  while ($row = mysql_fetch_array($res)) {
?>

    <tr>
      <td><?php echo $row['coddisciplina'];?></td>
      <td><?php echo $row['nomedisciplina'];?></td>
    </tr>

<?php
  }
?>
</table>
<p> <p>
<a href='index.html'><p align="center">Voltar para Menu Principal</p></a>
</BODY>
</HTML>

```

Quero agradecer a todos os alunos que acessaram o <http://www.maromo.pro.br>, baixaram os artigos e materiais do site e que enviaram um “*feedback*” por terem aprendido um pouco com o que passei. Por essa razão desejo toda a sorte e felicidade a vocês.

Por favor, reporte qualquer problema encontrado nesse material para que eu possa melhorá-lo.

Obrigado,

Marcos Roberto de Moraes  
Administrador de Sistemas de Informação

[professormoraes@gmail.com](mailto:professormoraes@gmail.com)

## 9. Referências Bibliográficas

**BATTISTI**, Júlio, web-site: <http://www.juliobattisti.com.br> acessado em <NOV/2006>.

**CRIARWEB**, web-site: Fonte: <http://www.criarweb.com/artigos/537.php> acessado em <Nov/2006>.

**EASYPHP**, EasyPhp, web-site: <http://www.easuphp.org> acessado em <NOV/2006>.

**MORAES**, Marcos Roberto de Moraes, web-site: <http://www.maromo.pro.br> acessado em <DEZ/2006>.

**NIEDERAUER**, Juliano. Livro: Desvendando WebSites com PHP, Editora: Novatec Editora , Ano: 2004.

**PHP Tutorial**, web-site: <http://www.php.net/tut.php> site do projeto PHP acessado em <NOV/2006>.

**PHPEditor**, PHPEditor, web-site: [http://paginas.terra.com.br/informatica/php\\_editor/](http://paginas.terra.com.br/informatica/php_editor/) acessado em <JAN/2007>.

**TIMASTER**, web-site: <http://www.timaster.com.br> acessado em <DEZ/2006>.

**WIKIPEDIA**, Enciclopédia Wikipedia, web-site: <http://www.wikipedia.org> acessado em <NOV/2006>.