



INOVAÇÃO
E TECNOLOGIA

unidade

3

Laboratório de PROGRAMAÇÃO



Desenvolvimento com PHP - III

Prezado estudante,

Estamos começando uma unidade desta disciplina. Os textos que a compõem foram organizados com cuidado e atenção, para que você tenha contato com um conteúdo completo e atualizado tanto quanto possível. Leia com dedicação, realize as atividades e tire suas dúvidas com os tutores. Dessa forma, você, com certeza, alcançará os objetivos propostos para essa disciplina.

OBJETIVO GERAL



Reconhecer e utilizar ferramentas de auxílio no desenvolvimento de códigos com PHP bem como proteger os dados mediante aos ataques de SQL Injection.

OBJETIVOS ESPECÍFICOS



- Reconhecer o uso prático da ferramenta phpMyAdmin;
- Reconhecer classes e instanciação no PHP;
- Aplicar a remoção de SQL Injection no PHP;
- Definir a instalação do framework Zend;
- Construir base de dados e tabelas com phpMyAdmin;
- Desenvolver manipulação de dados e importação e exportação de banco de dados com phpMyAdmin;
- Descrever métodos para classes em PHP;
- Definir herança e polimorfismo com PHP;
- Reconhecer criptografia no PHP;
- Descrever o padrão MVC no PHP;
- Descrever os principais comandos do framework Zend;
- Reconhecer exemplos de aplicações do framework Zend.



Parte 1

phpMyAdmin

phpMyAdmin

Objetivos de aprendizagem

Ao final deste texto, você deve apresentar os seguintes aprendizados:

- Reconhecer o uso prático da ferramenta phpMyAdmin.
- Construir base de dados e tabelas com phpMyAdmin.
- Desenvolver manipulação de dados, importação e exportação de banco de dados.

Introdução

O phpMyAdmin é uma ferramenta essencial para administração de bancos de dados MySQL pela internet. Por meio do servidor web XAMPP instalado, tarefas como criar e remover bases de dados e tabelas, inserir, editar e remover campos em tabelas e executar códigos em SQL se tornam muito fáceis, devido a sua interface intuitiva e amigável.

Neste capítulo, você irá estudar o uso prático da ferramenta phpMyAdmin, como construir bases de dados e tabelas com o phpMyAdmin e, também, como manipular dados, importar e exportar bancos de dados com o phpMyAdmin.

Reconhecer o uso prático da ferramenta phpMyAdmin

O phpMyAdmin é um aplicativo para web de código aberto, estável e flexível, que foi desenvolvido em PHP. É uma das ferramentas mais conhecidas do PHP, utilizada para fazer a administração de bancos de dados MySQL pela internet, e que suporta uma grande variedade de operações no MySQL.

As operações mais utilizadas do phpMyAdmin envolvendo gerenciamento de bancos de dados, tabelas, campos, relacionamentos, índices, permissões de usuários, entre outras, podem ser realizadas por uma interface de usuário, que permite a execução direta de códigos em SQL.

Com o aplicativo phpMyAdmin é possível (PHPMYADMIN, c2003-2018):

- criar e remover bases de dados;
- criar, alterar e remover tabelas;
- inserir, editar e remover campos;
- executar códigos em SQL;
- executar manipulação de campos-chave em tabelas.

O aplicativo phpMyAdmin vem acompanhado de uma detalhada documentação, e os usuários estão liberados para atualizar as páginas Wiki do site oficial, no qual podem compartilhar suas ideias e procedimentos utilizados para executar uma variedade de operações. Além disso, existe uma equipe de consultoria que tenta ajudar os usuários que enfrentam problemas com a ferramenta, além de outros canais de suporte nos quais pode ser obtido auxílio.

Algumas das características mais importantes do phpMyAdmin são (PHPMYADMIN, c2003-2018):

- interface web intuitiva;
- suporte para a maioria dos recursos do MySQL;
- possibilidade de criar e arrastar bancos de dados, tabelas, visualizações, campos e índices;
- possibilidade de criar, copiar, arrastar, renomear e alterar bancos de dados, tabelas, campos e índices;
- executa, cria e edita qualquer declaração SQL, inclusive queries em lote;
- gerencia contas e privilégios de usuário e privilégios do MySQL;
- gerencia procedimentos e disparadores armazenados;
- permite importação de dados de vários formatos, como CSV e SQL;
- possibilita exportar dados para vários formatos (CSV, SQL, XML, PDF, ISO / IEC 26300 - OpenDocument Text and Spreadsheet, Word, LATEX);
- abre documentos de texto e planilhas, como Microsoft Word e Excel;
- permite administrar vários servidores;
- possibilita criação gráfica do layout do banco de dados em vários formatos, inclusive PDF;
- permite a criação de consultas complexas utilizando o Query-By-Example (QBE);
- facilita a pesquisa global em um banco de dados ou um subconjunto dele;
- permite ativação de consultas de monitor para processos;
- possibilita transformar dados armazenados em qualquer formato usando um conjunto de funções predefinidas, como exibir dados BLOB como imagem ou download-link.

O phpMyAdmin é muito utilizado pelos programadores web, devido à facilidade com a qual permite a manipulação das bases de dados. Ele é visto como uma ferramenta quase que obrigatória na maioria dos sites de hospedagem para web, além de estar incluído em pacotes off-line, como o EasyPHP, o PHP Triad e o WAMOServer.

Não é necessário fazer a instalação do phpMyAdmin, principalmente quando o sistema operacional é o Microsoft Windows, pois geralmente esse software vem incluído dentro de pacotes de aplicativos, como é o caso do servidor web XAMPP. Apesar disso, você pode fazer o download do phpMyAdmin diretamente do site oficial e utilizá-lo como um software independente (PHPMYADMIN, c2003-2018).

Dessa forma, para executar o phpMyAdmin será necessário abrir o painel de controle do XAMPP e clicar em ADMIN no serviço MySQL, conforme demonstrado na Figura 1.

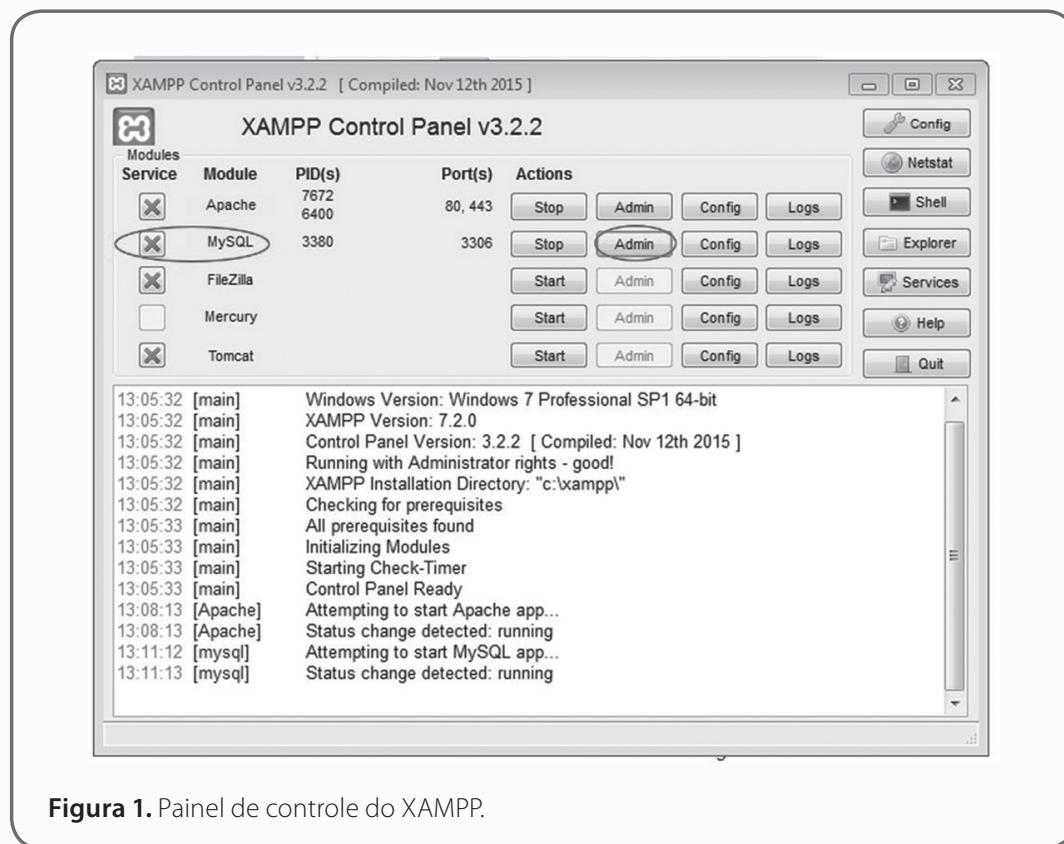


Figura 1. Painel de controle do XAMPP.

Esse comando abrirá o navegador web e exibirá a tela principal do phpMyAdmin, semelhante a que é mostrada na Figura 2. Independentemente do pacote de aplicativos escolhido para obter o phpMyAdmin, a tela principal será sempre essa, pois é a tela padrão do aplicativo.

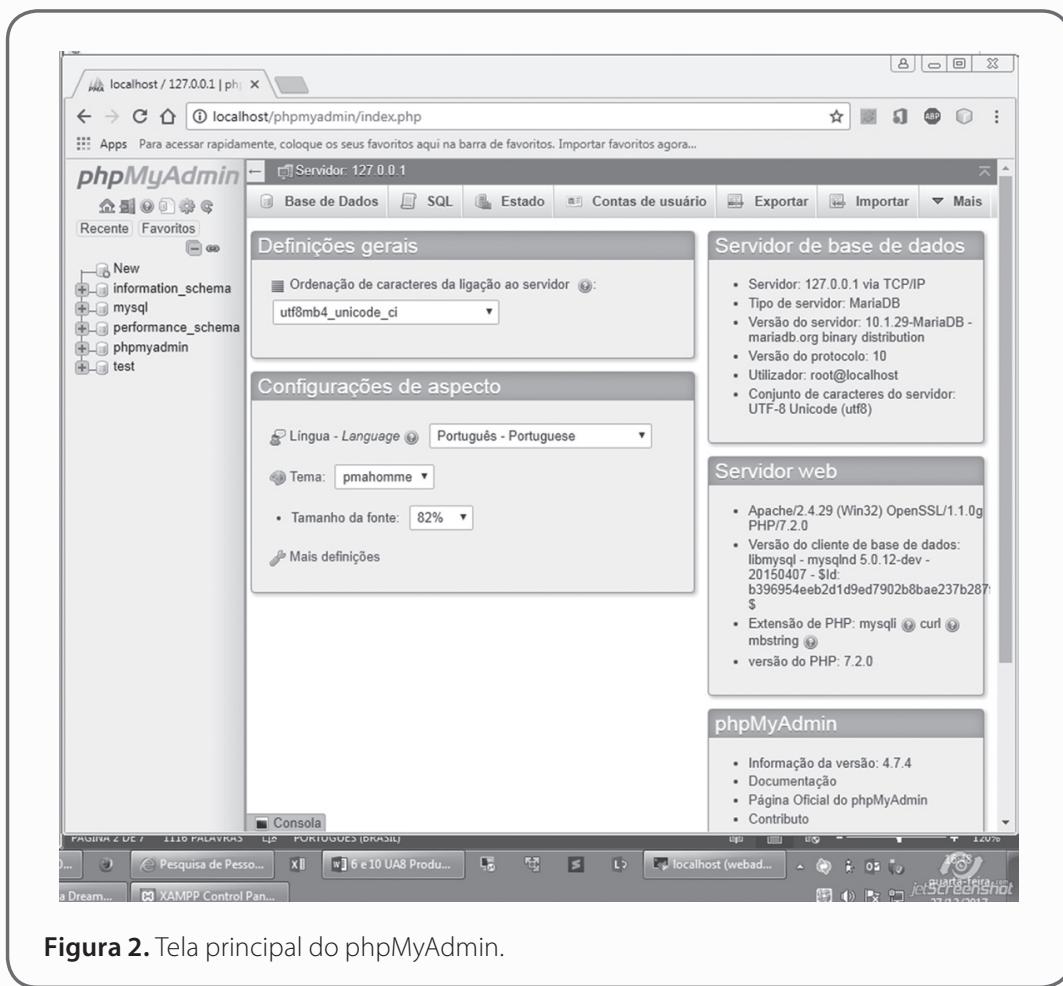


Figura 2. Tela principal do phpMyAdmin.

Na parte superior da tela principal do phpMyAdmin está o menu principal, cujos botões mais relevantes são os apresentados na Figura 3 e detalhados a seguir (PHPMYADMIN, c2003-2018):

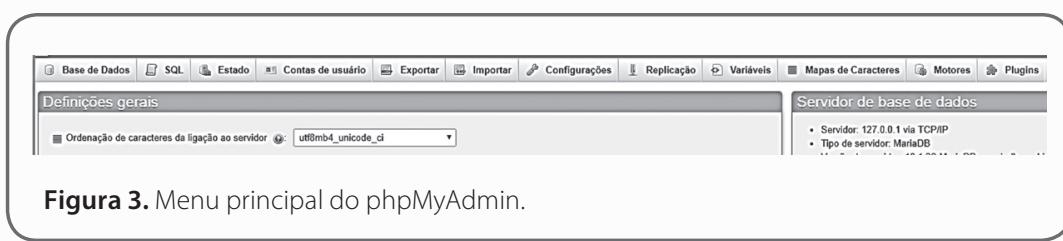


Figura 3. Menu principal do phpMyAdmin.

- **Base de dados:** área que contém os bancos de dados. Possui as mesmas opções do menu lateral esquerdo.
- **SQL:** área em que pode ser digitado um código SQL para execução.
- **Estado:** área que possui informações sobre o *status* dos bancos de dados que estão sendo utilizados, como conflitos gerados, questões de segurança, etc.

- **Exportar:** área que permite exportar o banco de dados de maneira simples e rápida para qualquer local do computador.
- **Importar:** área que permite importar arquivos, de forma total ou parcial, e escolher o formato desejado para o arquivo no servidor.
- **Variáveis:** área que contém informações sobre as variáveis do servidor e suas configurações.
- **Mapa de caracteres:** área que contém o nome de cada *collation* e qual o idioma usado para cada um.

Construir base de dados e tabelas com phpMyAdmin

A criação de bancos de dados e de tabelas se torna muito mais fácil utilizando a ferramenta de administração phpMyAdmin. Para dar início a essa atividade, abra o phpMyAdmin pelo painel de controle do XAMPP (APACHE FRIENDS, c2018).

Depois de aberta a tela principal do phpMyAdmin, você deve clicar no botão **base de dados** do menu principal, ou na opção **new** do menu lateral. Isso fará aparecer a área de criação de base de dados, como você pode ver na Figura 4. Nesse momento, você deve inserir um nome para a base de dados e clicar no botão **criar**. Para esse exemplo, o nome escolhido para a base de dados foi “base_teste”.

| Base de Dados | Agrupamento (Collation) | Acções |
|--------------------|-------------------------|---------------------------------------|
| information_schema | utf8_general_ci | Verificar Privilégios |
| mysql | latin1_swedish_ci | Verificar Privilégios |
| performance_schema | utf8_general_ci | Verificar Privilégios |
| phpmyadmin | utf8_bin | Verificar Privilégios |
| test | latin1_swedish_ci | Verificar Privilégios |
| Total: 5 | latin1_swedish_ci | |

Check all Com os selecionados: Elimina

Nota: Activar as estatísticas aqui pode causar um grande volume de tráfego entre os servidores web e MySQL.

Enable statistics

Figura 4. Área de criação de base de dados.

O nome da base de dados criada aparecerá no menu lateral, demonstrado na Figura 5, para que o banco de dados possa ser administrado.

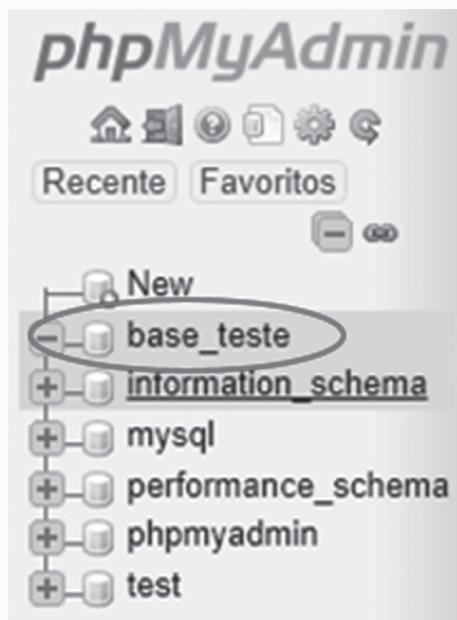


Figura 5. Menu lateral com banco de dados incluído.

Para criar tabelas dentro de um banco de dados, você pode clicar sobre o nome do banco no menu lateral, ou clicar sobre o botão **base de dados** do menu principal. Na nova janela, insira um nome para a tabela, a quantidade de colunas que ela deverá ter e clique no botão **executar**, conforme ilustra a Figura 6.

A screenshot of a 'Criar tabela' (Create Table) dialog box. At the top, a message says 'Nenhuma tabela encontrada na base de dados.' (No tables found in the database). Below that is a button labeled 'Criar tabela'. Underneath, there are two input fields: 'Nome:' containing 'tabela_nova' and 'Número de colunas:' containing '5'. At the bottom right of the dialog is a button labeled 'Executar'.

Figura 6. Criação de tabelas.

Na janela a seguir, haverá espaço para detalhar a quantidade de colunas informadas na janela anterior (veja Figura 7). Deverão ser inseridos, para cada coluna: nome, tipo de dados, tamanho e valores possíveis, valor predefinido, agrupamento, atributo, nulidade, tipo de índice, entre outros.

Nesse exemplo, foi criada a tabela **TABELA_NOVA**, com 5 campos, conforme abaixo:

- cod_cliente: integer, chave primária
- nome_cliente: text
- idade_cliente: integer
- apelido_cliente: text
- cod_cidade: integer

The screenshot shows the 'Estrutura' (Structure) tab of the MySQL Workbench Table Editor. The table name is set to 'tabela_nova'. The 'Estrutura' tab displays the following columns:

| Nome | Tipo | Tamanho/Valores* | Predefinido | Agrupamento (Collation) | Atributos | Nulo | Índice | A_J |
|-----------------|------|------------------|-------------|-------------------------|-----------|-------------------------------------|---------|-----|
| cod_cliente | INT | | None | | | <input checked="" type="checkbox"/> | PRIMARY | |
| nome_cliente | TEXT | | None | | | <input checked="" type="checkbox"/> | --- | |
| idade_cliente | INT | | None | | | <input checked="" type="checkbox"/> | --- | |
| apelido_cliente | TEXT | | None | | | <input checked="" type="checkbox"/> | --- | |
| cod_cidade | INT | | None | | | <input checked="" type="checkbox"/> | --- | |

Below the table structure, there are fields for 'Comentários da tabela:', 'Collation:', and 'Motor de armazenamento:' (InnoDB). The 'PARTITION definition:' section is also visible. At the bottom right, there are buttons for 'Pré-visualizar SQL' and 'Guarda'.

Figura 7. Criação das colunas.

Para visualizar o código SQL utilizado para a criação da tabela e de todas as suas colunas, basta clicar no botão **Pré-visualizar SQL**, que fica no canto inferior direito.

Depois que as colunas da tabela estiverem todas definidas, basta clicar em **Guarda**, para que ela seja salva no banco de dados.

Desenvolver manipulação de dados, importação e exportação de banco de dados

Para fazer a manipulação dos dados que estão armazenados nas tabelas de um banco de dados MySQL, são usados comandos como o **Insert**, usado para adicionar um registro novo; o **Update**, utilizado para alterar os dados de um registro gravado; e o **Delete**, usado para apagar um determinado registro da tabela (PHPMYADMIN, c2003-2018).



Fique atento

A linguagem SQL é dividida em três tipos, conforme a funcionalidade que os seus comandos oferecem:

- A **data definition language (DDL)**, ou linguagem de definição de dados, faz a interação com os objetos do banco. Os comandos mais usados na DDL são o Create, o Alter e o Drop.
- A **data manipulation language (DML)**, ou linguagem de manipulação de dados, faz a interação com os dados armazenados dentro das tabelas do banco. Os comandos mais usados na DML são o Insert, o Update e o Delete.
- A **data query language (DQL)**, ou linguagem de consulta de dados, executa diversos tipos de consulta aos dados que estão inseridos nas tabelas do banco. O comando mais usado na DQL é o Select.

Para iniciar a operação de manipulação de dados nas tabelas de um banco com o phpMyAdmin, é necessário escolher uma base de dados no menu lateral à esquerda, clicando uma vez sobre ela. Depois disso, você deve clicar no botão **SQL**, do menu principal. Isso fará aparecer uma área em branco na janela, que possibilitará a digitação do código SQL e a sua execução, como demonstrado na Figura 8.

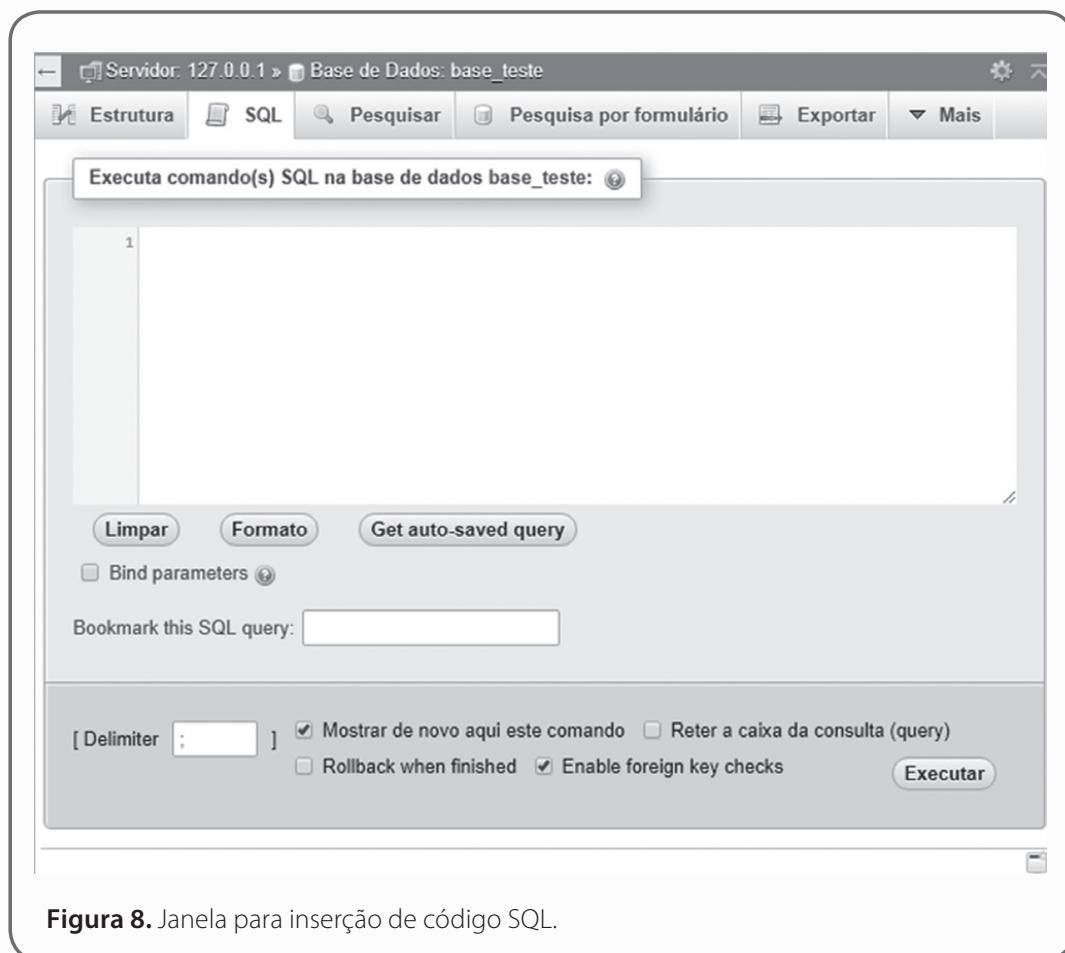


Figura 8. Janela para inserção de código SQL.

O comando SQL que faz a inserção de dados em uma tabela no banco, é o **Insert**. Sua sintaxe é a seguinte:

INSERT INTO nome _ tabela (lista de campos que vão receber os valores separada por vírgula)

VALUES (lista de valores que cada campo vai receber separada por vírgula)

No caso da tabela do exemplo, chamada TABELA_NOVA, o comando a seguir deve ser digitado para que seja incluído um registro novo, e, para executá-lo, deve-se clicar no botão **Executar**, conforme você pode observar na Figura 9.

```
INSERT INTO tabela _ nova (cod _ cliente,nome _ cliente,idade _ cliente,apelido _ cliente,cod _ cidade)
VALUES (1,'Aluno Novo',23,'Aluninho',5)
```

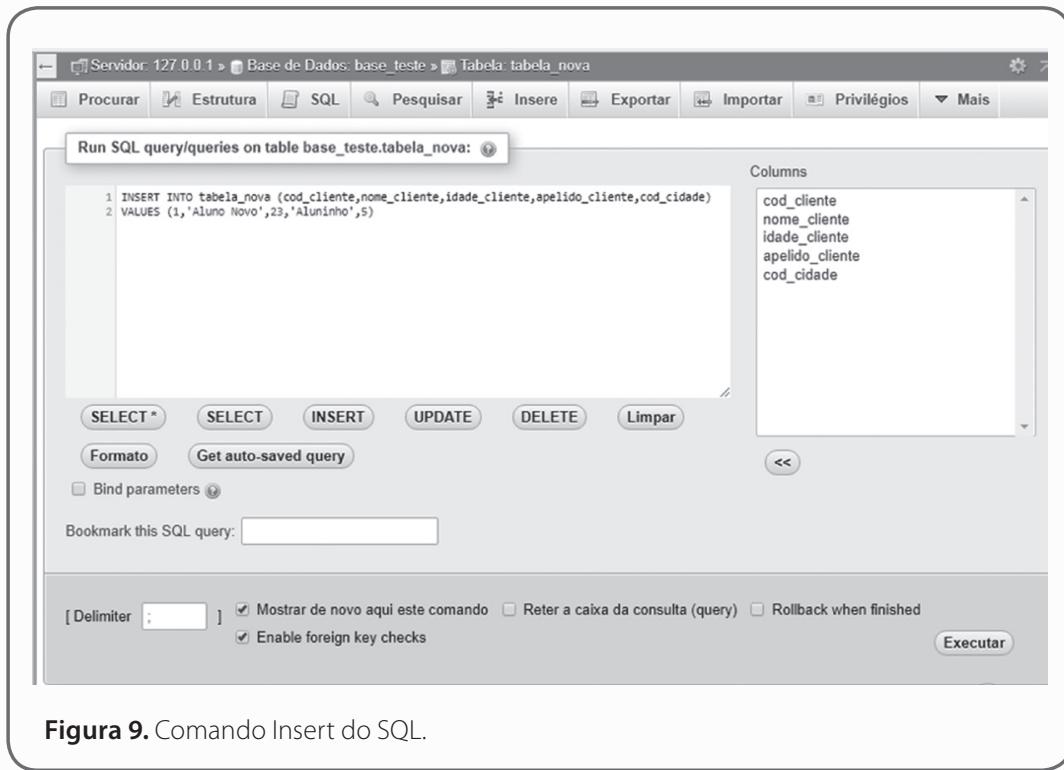


Figura 9. Comando Insert do SQL.

Se o comando **Insert** tiver sido executado de forma correta, o phpMyAdmin vai apresentar uma mensagem informando a quantidade de linhas inserida na tabela.

O comando SQL que faz a alteração de dados em uma tabela no banco, é o **Update**. Sua sintaxe é a seguinte:

```
UPDATE nome _ tabela
SET campo = novo _ valor
WHERE condição
```

É importante observar que o comando **Update** só fará alterações nos campos informados da tabela quando eles obedecerem à condição informada no comando. No caso da tabela do exemplo, chamada **TABELA_NOVA**, o comando a seguir deve ser digitado para que seja alterado o apelido do registro novo recém incluído, que contém o campo **cod _ cliente** igual a 1, como condição. Para executá-lo, deve-se clicar no botão **Executar**, como demonstra a Figura 10.

```
UPDATE tabela _ nova
SET apelido _ cliente = 'Alunão'
WHERE cod _ cliente = 1
```

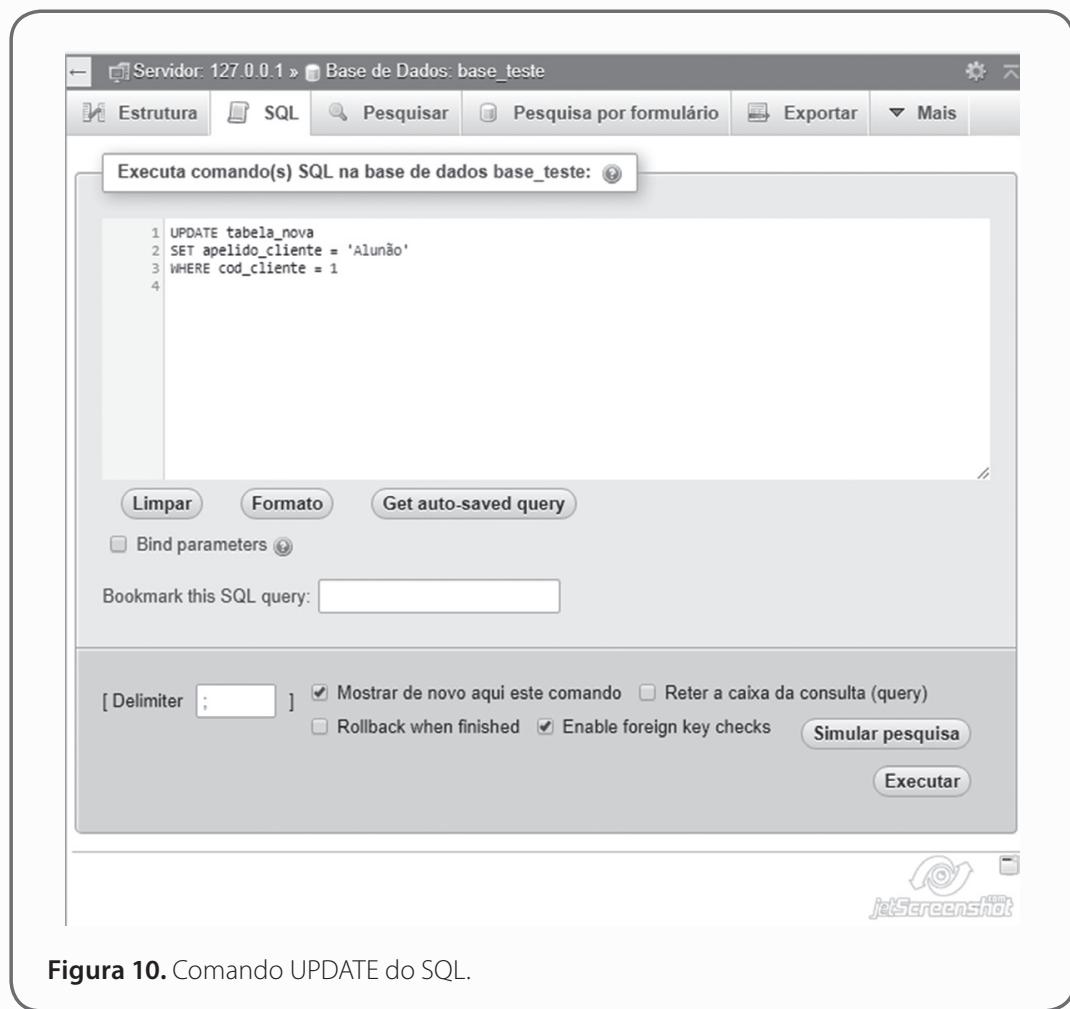


Figura 10. Comando UPDATE do SQL.

Se o comando Update tiver sido executado de forma correta, o phpMyAdmin vai apresentar uma mensagem informando a quantidade de linhas da tabela que foram afetadas pela alteração de dados.

O comando SQL que faz a exclusão de dados em uma tabela no banco, é o **Delete**, e sua sintaxe é a seguinte:

```
DELETE FROM nome _ tabela
WHERE condição
```

Da mesma forma que o comando Update, é importante observar que o comando **Delete** também obedece à condição informada para fazer a exclusão dos registros da tabela.

No caso da tabela do exemplo, chamada TABELA_NOVA, o comando a seguir deve ser digitado para que seja excluído o registro que foi recentemente alterado, que contém o campo cod _ cliente igual a 1, como condição. Para executá-lo, deve-se clicar no botão **Executar**, como você pode ver na Figura 11.

```
DELETE FROM tabela _ nova
WHERE cod _ cliente = 1
```

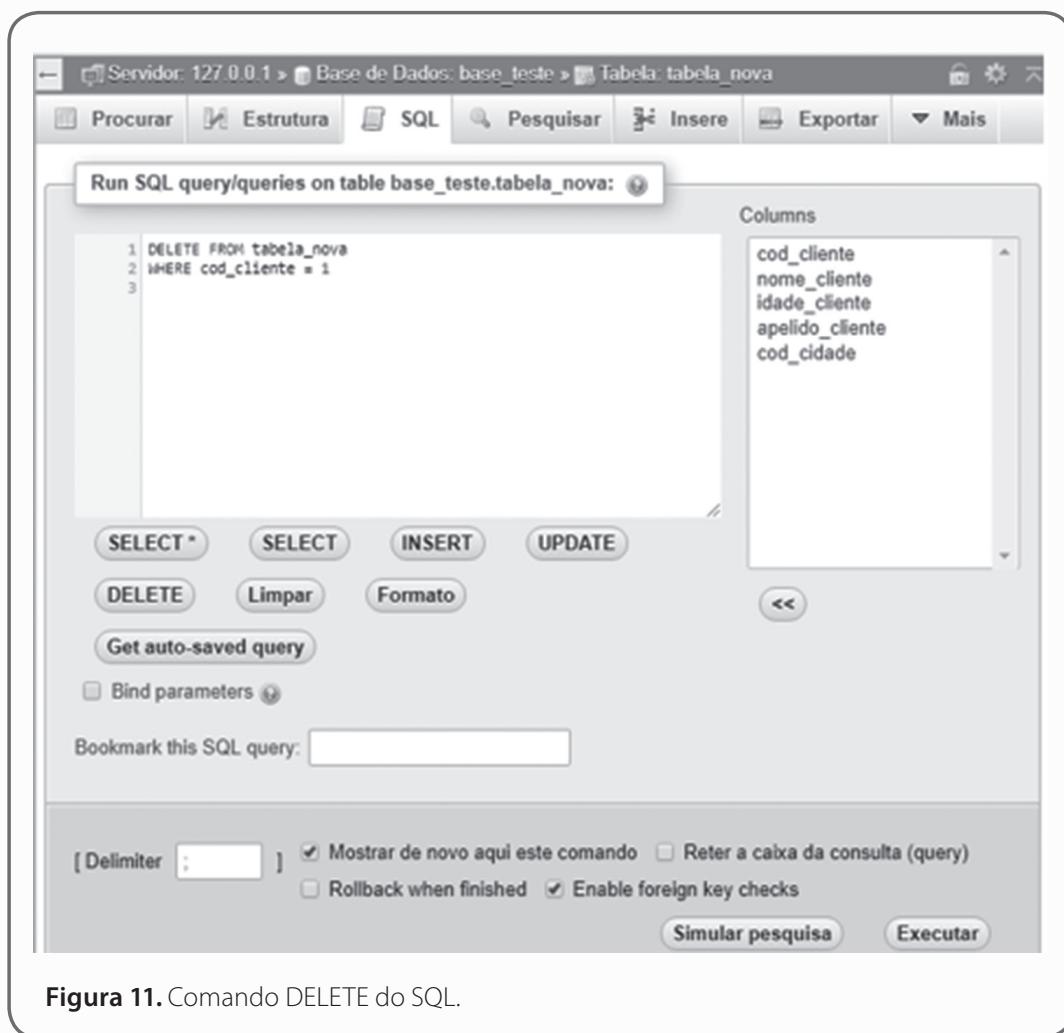


Figura 11. Comando DELETE do SQL.

Se o comando **Delete** tiver sido executado de forma correta, o phpMyAdmin vai apresentar uma mensagem informando a quantidade de linhas da tabela que foram afetadas pela exclusão de registros.

Além da manipulação de dados, outras funcionalidades muito importantes do phpMyAdmin são a importação e a exportação de bancos de dados, pois ele permite que essas tarefas sejam feitas utilizando inúmeros formatos de arquivo (PHPMYADMIN, c2003-2018).

O exemplo da Figura 12 mostra como fazer a importação de um banco de dados, chamado de UA8.xml para o servidor que está sendo utilizado no momento. Para iniciar a atividade, você deve clicar no botão **Importar** do menu principal do phpMyAdmin, fazendo aparecer a janela que contém as opções de importação.

Fazendo importação para o servidor atual

File to import:

O ficheiro pode ser comprimido (gzip, bzip2, zip) ou descomprimido.
O nome de um ficheiro comprimido tem que acabar em **.[format].[compression]**. Exemplo: **.sql.zip**

Procurar no seu computador: Nenhum arquivo selecionado (Tamanho máximo: 2,048KB)
You may also drag and drop a file on any page.

Configurar o Mapa de Caracteres do ficheiro:

Partial import:

Permite a interrupção de uma importação quando o script deteta que está próximo do limite de tempo do PHP. (Esta é uma boa forma de importar ficheiros grandes, contudo pode interromper transações.)
Skip this number of queries (for SQL) starting from the first one:

Other options:

Enable foreign key checks

Formato:

Opções específicas do formato:

SQL compatibility mode:
 Do not use AUTO_INCREMENT for zero values



Figura 12. Importar banco de dados no phpMyAdmin.

Para importar o banco, você deve clicar no botão **Escolher arquivo**. Na próxima janela, você encontrará o local do computador em que o banco a ser importado está armazenado, deve clicar nele e, depois, no botão **Abrir**. Nesse momento, o campo Formato da tela já deve ter se modificado para XML, devido ao fato de estarmos importando um banco de dados que se chama UA8.xml. Para finalizar a importação, basta clicar no botão **Executar**. Acompanhe o exemplo na Figura 13.

Caso a importação tenha sido bem-sucedida, o phpMyAdmin mostrará uma janela contendo a mensagem “Importação terminou com sucesso”.

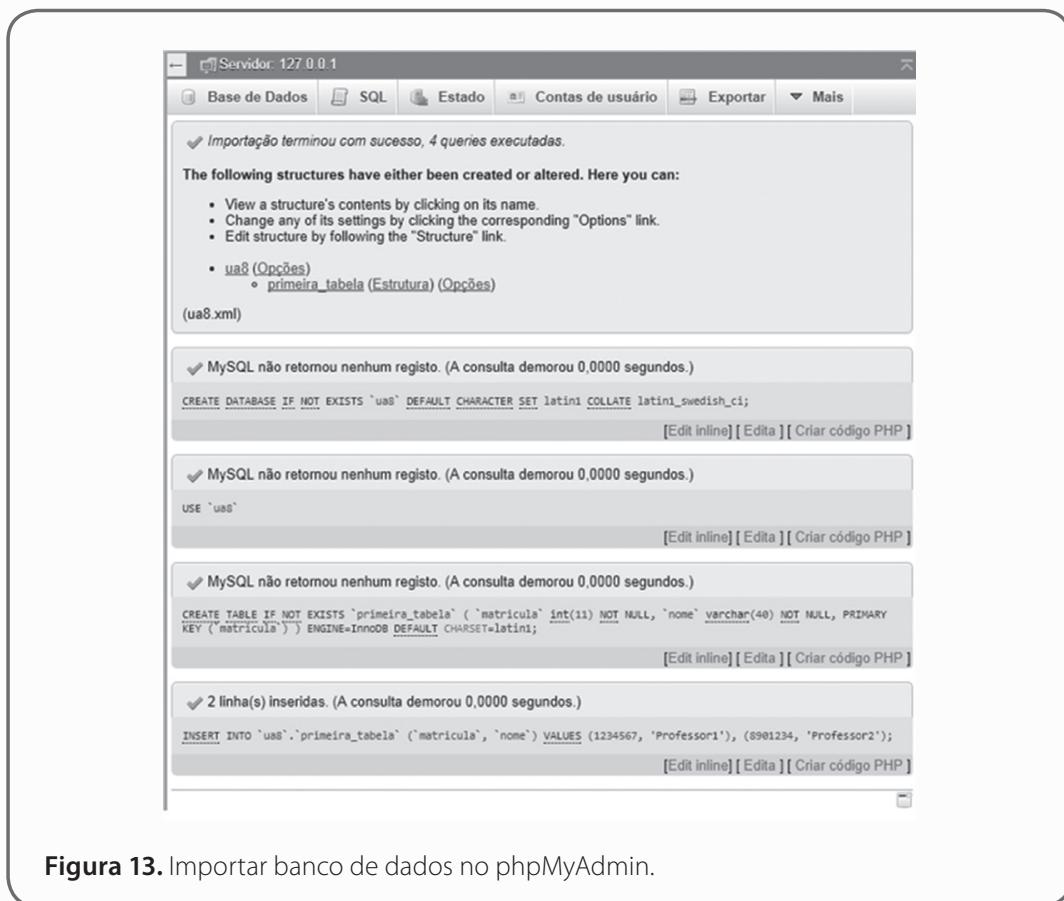


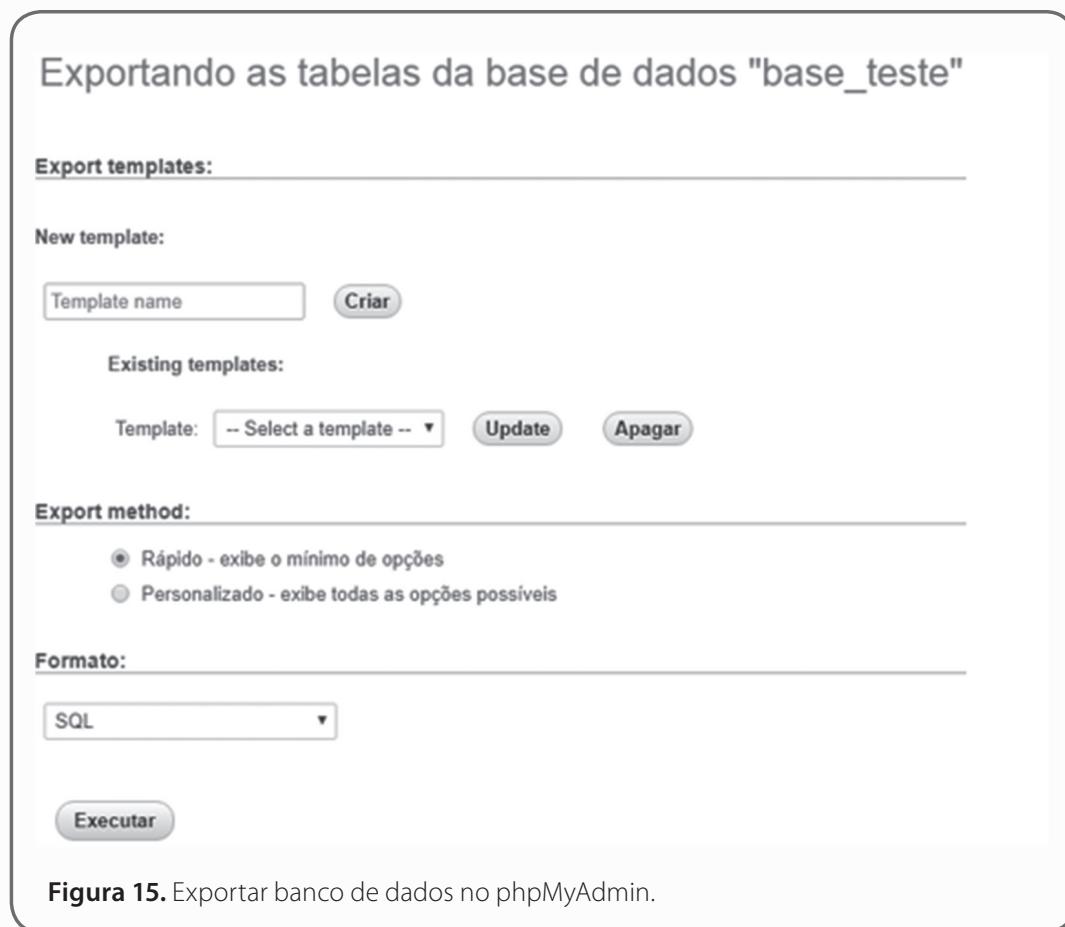
Figura 13. Importar banco de dados no phpMyAdmin.

Veja, na Figura 14, como é possível verificar que o banco de dados importado, UA8.xml, já aparece no menu lateral, com sua tabela e os registros incluídos.



Figura 14. Importar banco de dados no phpMyAdmin.

Para exemplificar a exportação de um banco de dados do servidor atual, será utilizado o banco de dados chamado `base_teste`, que será exportado com a extensão `.CSV`, como você pode observar na Figura 15. Para iniciar a atividade, você deve clicar sobre o nome do banco de dados a ser exportado, no menu lateral. Logo depois, deve clicar no botão **Exportar** do menu principal do phpMyAdmin. Isso fará aparecer a janela que contém as opções de exportação.



Para exportar o banco, deve-se escolher o formato desejado para a exportação, na área Formato. São várias as opções de extensão disponíveis, entre elas estão SQL, XML, CSV, PDF e Microsoft Word. Para finalizar a exportação, basta clicar no botão **Executar**. Por fim, será aberta uma janela para que seja indicado o local do computador em que o arquivo com o banco de dados deve ser armazenado.



Saiba mais

Visualizando dados: instrução SELECT

Não faria sentido armazenar dados que não pudessem ser visualizados quando necessário, não é? Confira agora como visualizar esses dados. O comando a ser utilizado é o SELECT:

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

- SELECT: comando para chamar a visualização de registros (campos) da tabela, para referência da visualização dos registros.
- FROM: chamada para indicar a tabela (ou as 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. Atenção: O * (asterisco) terá um papel importante neste comando.



Referências

APACHE FRIENDS. XAMPP Apache + MariaDB + PHP + Perl. [S.I.], c2018. Disponível em: <https://www.apachefriends.org/pt_br/index.html>. Acesso em: 20 dez. 2017.

PHPMYADMIN. About. [S.I.], c2003-2018. Disponível em: <https://www.phpmyadmin.net>. Acesso em: 17 dez. 2017.



PREZADO ESTUDANTE

**ENCERRA AQUI O TRECHO DO LIVRO DISPONIBILIZADO
PELA SAGAH PARA ESTA PARTE DA UNIDADE.**



Parte 2

Linguagem PHP Orientada a Objetos

Linguagem PHP orientada a objetos

Objetivos de aprendizagem

Ao final deste texto, você deve apresentar os seguintes aprendizados:

- Reconhecer classes e instanciação no PHP.
- Descrever métodos para classes em PHP.
- Definir herança e polimorfismo com PHP.

Introdução

A princípio o PHP não era uma linguagem voltada à orientação a objetos, esse tipo de programação só começou a ser suportado a partir de sua versão 3, mas vem sendo aprimorado a cada nova versão. Atualmente, o PHP é uma linguagem de programação com alta performance e inúmeras funcionalidades, o que lhe garante um dos melhores desempenhos na orientação a objetos.

Neste capítulo, você estudará as classes e instanciação, os métodos para as classes e, também, a herança e o polimorfismo utilizando PHP.

Classes e instanciação no PHP

As linguagens de programação orientadas a objetos sempre permitem a implementação de classes, pois elas são elementos fundamentais nesse paradigma de programação. Uma classe é uma descrição de algo que faz a abstração de um conjunto de objetos possuindo características semelhantes. Pode-se dizer que uma classe é o encapsulamento de abstrações de algo que descreve o conteúdo e o comportamento de elementos do mundo real, representadas por objetos (PHP, c2001-2018a).

De forma mais simplificada, uma classe pode ser a definição de uma descrição de propriedades ou estados possíveis para um conjunto de objetos, ou para o comportamento e para as ações que podem ser aplicadas a eles.

Uma subclasse, ou classe filha, é uma outra classe, que herda características da sua superclasse, ou classe mãe.

É importante lembrar que toda classe (ENGHOLM, 2010):

- deve iniciar a definição com a palavra `class`;
- deve possuir um nome, que deve começar com uma letra ou um sublinhado, seguido de qualquer sequência de letras, números e sublinhados;
- deve ter um limitador ou modificador de acesso, que pode ser `public`, `private` ou `protected`;
- deve possuir elementos como atributos e métodos;
- deve ser definida antes de ser instanciada.

Um objeto, ou instância de uma classe, é o elemento capaz de armazenar estados, por meio de seus atributos, e executar ações de acordo com as chamadas enviadas a ele, bem como relacionar e enviar chamadas para outros objetos.

Como boa prática, recomenda-se salvar cada classe em um arquivo separado, nomeado com o mesmo nome da classe. No exemplo apresentado a seguir, a classe Cachorro foi salva como `Cachorro.php`.

```
1 <?php
2     class Cachorro {
3         // DECLARAÇÃO DE ATRIBUTO
4         public $raca;
5
6         // DECLARAÇÃO DE MÉTODO
7         public function getRaca() {
8             return $this->raca;
9         }
10    }
11 ?>
```

A classe Cachorro apresentada possui um atributo público chamado `raca` e um método público chamado `getRaca`. Para criar um objeto ou uma instância dessa classe no PHP, é preciso utilizar a instrução `new`, conforme apresentado a seguir.

```
1 <?php
2 require_once "Cachorro.php";
3
4 $cachorro1 = new Cachorro();
5 $cachorro1->raca = "Yorkshire";
6 echo "O objeto cachorro1 é da raça: " . $cachorro1->getRaca();
7
8 $cachorro2 = new Cachorro();
9 $cachorro2->raca = "Labrador";
10 echo "O objeto cachorro2 é da raça: " . $cachorro2->getRaca();
11 ?>
```

Como a classe Cachorro está implementada em um arquivo separado, é preciso incluir a classe pela instrução `require __once` apresentada na linha 2. Isso torna possível realizar a instância dessa classe em outros locais do programa, com o objetivo de acessar seus atributos e métodos.

Na linha 4, criamos um objeto nomeado `$cachorro1` e na linha 5 definimos “Yorkshire” ao atributo `raca` desse objeto. Na linha 6, realizamos a chamada do método `getRaca`, que retorna a raça do objeto `$cachorro1` e a apresenta na tela pela instrução `echo`.

Na linha 8, criamos outro objeto da classe Cachorro chamado `$cachorro2` e definimos “Labrador” ao atributo `raca` na linha 9. Isso mostra que é possível instanciar diversas vezes uma classe e, assim, criar vários objetos dessa classe com valores diferentes.

De maneira geral, uma classe faz a definição do estado e do comportamento do objeto implementando atributos e métodos para ele. Os atributos são os elementos que definem uma classe e que podem também serem chamados de campos ou de propriedades. Eles são as características de um objeto e indicam as informações que podem ser armazenadas por um objeto que pertence a uma classe, representando o estado do objeto e a estrutura da classe.

Os métodos também podem ser referenciados como operações ou serviços de um objeto, sendo responsáveis pelos procedimentos, que formam as ações e os comportamentos que o objeto de uma classe poderá oferecer, representando a alteração do estado ou o fornecimento de informações acerca do objeto. Em poucas palavras, os métodos são as ações que os objetos podem executar quando forem chamados.

O Quadro 1 apresenta exemplos de atributos, métodos e objetos da classe Cachorro.

Quadro 1. Esquema de classe, atributos, método e objeto.

| Classe cachorro | | |
|---|---|---|
|  |  |  |
| Atributos: \$nome, \$raca, \$idade, \$peso | | |
| Métodos: pegarOsso(), latir(), lamber(), morder() | | |
| Objeto1 \$nome = Billy \$raca = Linguiça \$idade = 1 ano \$peso = 2,5kg | Objeto2 \$nome = Pretinho \$raca = Vira-lata \$idade = 1 ano \$peso = 3kg | Objeto3 \$nome = Lambão \$raca = Maltês \$idade = 2 anos \$peso = 3,7kg |
| <i>Fonte:</i> drawkman/Shutterstock.com, Linza/Shutterstock.com e Pushkin/Shutterstock.com. | | |

No PHP, as classes podem ser do tipo concreta, abstrata e final:

- As classes concretas fazem a implementação de todos os seus métodos e permitem a criação de instâncias. Esse tipo de classe não possui métodos abstratos e, se forem utilizadas em um cenário como esse, certamente serão subclasses de uma classe abstrata. As classes concretas são as mais comuns, e utilizam a seguinte sintaxe:

```
1 class NomeDaClasse {  
2     //atributos  
3     //métodos  
4 }
```

- As classes abstratas são desenvolvidas para fazer a representação de entidades e de conceitos abstratos, e são sempre superclasses que não possuem instâncias. Elas definem um modelo de funcionalidade de forma genérica, como uma implementação incompleta, que é compartilhada por um grupo de subclasses. Cada subclasse deverá complementar a funcionalidade da classe abstrata, adicionando um comportamento específico a ela. Os métodos das classes abstratas normalmente são métodos abstratos, implementados nas subclasses concretas, a fim de definir um comportamento específico. O método abstrato não contém código, ele define apenas uma assinatura para o método. Para criar uma classe abstrata, utiliza-se a seguinte sintaxe:

```
1 abstract class NomeDaClasse{  
2     //atributos  
3     //métodos  
4 }
```

- As classes do tipo final não permitem que outras classes realizem herança delas, isto é, elas não podem ser herdadas, sendo, portanto, a classe final de utilização da instância. Para criar uma classe final, utiliza-se a seguinte sintaxe:

```
1 final class NomeDaClasse{  
2     //atributos  
3     //métodos  
4 }
```



Link

Confira as classes e instâncias de classe no link ou código a seguir (PHP, c2001-2018c).

<https://goo.gl/YuQGH9>



Métodos para classes em PHP

Na orientação a objetos, os métodos são os elementos que determinam o comportamento dos objetos de uma classe, sendo considerados semelhantes ou análogos às funções ou aos procedimentos da programação estruturada, uma vez que a chamada de um método, assim como a chamada de uma função, pode alterar o estado de um objeto ou de uma variável (PRESSMAN, 2011).

As ações só acontecem quando os métodos são chamados por meio dos objetos e, durante a execução do programa, um método deve afetar somente um objeto.

Em geral, os métodos são definidos dentro da sua classe. Eles definem o comportamento das instâncias da sua classe, enquanto o programa estiver sendo executado, e conseguem acessar dados armazenados na instância da classe a qual estão associados, além de serem capazes de controlar o estado de uma instância da classe.

Método construtor

Existem métodos, em orientação a objetos, que se chamam construtores. Esse tipo de método é chamado automaticamente no momento em que uma nova instância de uma classe for criada. O método construtor é responsável por alocar os recursos necessários para o funcionamento do objeto, além da definição inicial dos atributos (ENGHOLM, 2010):

```
1  <?php
2      class Funcionario {
3          public $nome;
4          public $idade;
5          public function __construct($nome='', $idade=0) {
6              $this->nome = $nome;
7              $this->idade = $idade;
8          }
9          public function imprimir(){
10             echo "Nome: " . $this->nome . "<br />Idade: " .
11                 $this->idade . "<br />";
12         }
13     }
14 ?>
```

No código apresentado, a linha 5 contém o construtor que inicializa os atributos da classe Funcionario. Quando é feita a chamada do construtor, é possível fazer a instância da classe no mesmo momento em que é feita a passagem dos valores de todos os atributos da classe.

O exemplo apresentado a seguir cria dois objetos a partir da classe Funcionario. O primeiro objeto (funcionario1) não faz uso do construtor e requer a definição dos atributos nome e idade de maneira independente (linhas 5, 6 e 7). Já o segundo objeto (funcionario2) é criado pelo construtor e os atributos são passados por parâmetro na instanciação da classe Funcionario (linha 11):

```
1 <?php
2 require_once "Funcionario.php";
3
4 //Criação de um objeto sem o uso do construtor
5 $funcionario1 = new Funcionario();
6 $funcionario1->nome = "Jeanine";
7 $funcionario1->idade = "43";
8 $funcionario1->imprimir();
9
10 //Criação de um objeto com o uso do construtor
11 $funcionario2 = new Funcionario('Mauricio', 41);
12 $funcionario2->imprimir();
13 ?>
```



Fique atento

O PHP não exige que toda classe tenha um método construtor, deixando a cargo do programador decidir quando precisa implementá-lo.

Modificadores de acesso

É possível fazer o encapsulamento do estado de um objeto por meio de limitação ou modificação do acesso aos atributos de uma classe pelos seus métodos. Para que isso seja possível, as linguagens de programação orientadas a objetos oferecem modificadores de acesso para cada membro de uma classe.

Os modificadores de acesso são muito importantes na programação orientada a objetos, traduzindo-se em palavras reservadas da própria linguagem que determinam de que forma será feito o acesso aos atributos e/ou métodos de uma classe. É por meio do modificador de acesso que é determinada a visibilidade de um atributo ou de um método pertencente a uma classe, ou seja, o modificador de acesso define se ele vai ou não poder ser acessado de fora daquela classe que o declarou.

Para que a visibilidade de um atributo ou de um método seja modificada, é preciso preceder a sua declaração com o modificador, conforme segue:

```
1 modificador $nomeDoAtributo;  
2 modificador function nomeDoMetodo() {  
3 }
```

Sommerville (2011) cita que cada linguagem de programação pode definir os seus próprios modificadores de acesso por meio de palavras reservadas e criadas especificamente para isso. No entanto, o PHP fornece os modificadores de acesso listados a seguir e apresentados no Quadro 2.

- Private (privado): é o modificador que define o tipo de acesso mais restrito. Ele indica que o atributo e/ou o método da classe pode ser acessado apenas pela sua própria classe, ou seja, nenhuma outra parte do código, nem mesmo as suas subclasses, podem acessá-los.
- Protected (protegido): é o modificador que indica que o atributo e/ou método da classe pode ser acessado apenas pela sua própria classe e pelas suas subclasses, que são aquelas que herdam da classe.
- Public (público): é o modificador que define o tipo de acesso mais liberado possível. Ele indica que o atributo e/ou o método da classe pode ser acessado por qualquer classe existente e de qualquer outro ponto do código do programa. Os membros que são definidos como públicos em uma classe definem a interface que ela terá com as demais classes.

Quadro 2. Comparação entre os modificadores de acesso.

| Modificador | Classe | Subclasse | Global |
|-------------|--------|-----------|--------|
| Private | X | | |
| Protected | X | X | |
| Public | X | X | X |



Fique atento

Por padrão, o tipo privado é o modificador de acesso mais utilizado para a definição dos atributos, ao passo que o tipo público é mais utilizado para a definição dos métodos das classes. No entanto, caso não sejam especificados, o PHP definirá o tipo público, tanto para os métodos como para os atributos.

Encapsulamento

Os atributos e os métodos de uma classe podem ser definidos como públicos, protegidos ou privados, mas tudo que precisar ser conhecido externamente a respeito de uma classe deverá ser declarado como público. Somente os elementos que são membros da classe é que podem acessar seus métodos e atributos privados, garantindo que não existam ações inadequadas com os seus valores.

Nesse sentido, a única forma de consultar ou alterar atributos de um objeto é por meio de seus métodos, o que chamamos de encapsulamento. O encapsulamento é importante também para que o desenvolvedor não necessite conhecer como os métodos trabalham com os atributos, ele só precisa saber que eles existem, para então acessá-los.

Uma boa prática de encapsulamento é definir os métodos **getters** e **setters** de todos os atributos de uma classe. Isso servirá para, quando um atributo precisar receber um valor e tiver que passar seu valor para outro lugar do código, isso possa ser feito por esses métodos.

No exemplo a seguir, definimos os atributos como privados e criamos os métodos públicos de acesso para cada um deles (getters e setters). Dessa forma, implementamos o encapsulamento dos atributos e protegemos seu acesso direto.

```
1  <?php
2      class Funcionario {
3          private $nome;
4          private $idade;
5
6          public function getNome() {
7              return $this->nome;
8          }
9
10         public function setNome($nome) {
11             $this->nome = $nome;
12         }
13
14         public function getIdade() {
15             return $this->idade;
16         }
17
18         public function setIdade($idade) {
19             $this->idade = $idade;
20         }
21     }
22 ?>
```

A partir dessa implementação, devemos utilizar os métodos getters e setters para acessar os atributos da classe Funcionario, conforme segue.

```
1  <?php
2  require_once "Funcionario.php";
3
4  $funcionario1 = new Funcionario();
5  $funcionario1->setNome("Jeanine");
6  $funcionario1->setIdade(43);
7
8  echo "Nome: " . $funcionario1->getNome() . "<br />";
9  echo "Idade: " . $funcionario1->getIdade() . "<br />";
10 ?>
```

É importante notar que é definido um método `get` e um método `set` para cada um dos atributos da classe, e é isso que significa a expressão getters e setters. Algumas Integrated Development Environment (IDE), como o Eclipse e o NetBeans, possuem ferramentas de auxílio ao programador que escrevem automaticamente os getters e os setters dos atributos da classe.



Fique atento

Tentar acessar diretamente um atributo privado de uma classe, sem o uso dos métodos de acesso da própria classe, produzirá um erro de restrição de acesso. O mesmo acontecerá com um método declarado como privado, mas que seja chamado a partir de outra classe.

Herança e polimorfismo com PHP

Herança entre classes

Na programação orientada a objetos, a herança é o relacionamento pelo qual uma classe, que pode ser chamada de classe filha ou subclasse, recebe por herança todos os estados e todos os comportamentos possíveis da classe mãe ou superclasse, ou seja, é por meio da herança que as classes compartilham seus atributos e seus métodos (PHP, c2001-2018b).

Esse tipo de relacionamento também pode ser chamado de extensão, pois a subclasse estende os estados e comportamentos da superclasse. A extensão acontece quando se adicionam novos membros a uma subclasse, como novos atributos e novos métodos.

No PHP, uma classe pode utilizar os atributos e métodos de uma outra classe quando for utilizada a instrução `extends` na declaração da classe. É importante frisar que não é possível herdar atributos e métodos de múltiplas classes, pois uma subclasse só pode herdar de uma única superclasse.

No exemplo a seguir, a subclasse `Estagiario` acessa os atributos e os métodos da sua superclasse `Funcionario`, utilizada anteriormente, contando, ainda, com um atributo novo chamado `fimContrato`, observe:

```
1  <?php
2  require_once "Funcionario.php";
3
4  class Estagiario extends Funcionario {
5      private $fimContrato;
6
7      public function getFimContrato() {
8          return $this->fimContrato;
9      }
10
11     public function setFimContrato($fimContrato) {
12         $this->fimContrato = $fimContrato;
13     }
14
15 }
16 ?>
```

Note que os métodos `setNome`, `getNome`, `setIdade` e `getIdade`, acessados pela classe `Estagiario` (filha), pertencem à classe `Funcionario` (pai). Isso é possível pela herança, indicada pela instrução `class Estagiario extends Funcionario`.

```
1  <?php
2  require_once "Estagiario.php";
3
4  $estagiario = new Estagiario();
5  $estagiario->setFimContrato("2018-12-31");
6  $estagiario->setNome("Mauricio");
7  $estagiario->setIdade(41);
8
9  echo "Fim contrato: " . $estagiario->getFimContrato() . "<br />";
10 echo "Nome: " . $estagiario->getNome() . "<br />";
11 echo "Idade: " . $estagiario->getIdade() . "<br />";
12 ?>
```

Por meio da herança, é possível que uma subclasse compartilhe o código de programa de uma superclasse, aproveitando os métodos e atributos já definidos. A herança apresenta como grande vantagem o fato de, durante o desenvolvimento do programa, serem evitados pedaços de código desnecessários e duplicados, o que, entre outros benefícios, pode reduzir de maneira drástica o tempo gasto para a programação.

A herança pode ser vista de duas formas, como uma generalização ou uma especialização, conforme detalhado a seguir e esquematizado na Figura 1 (PRESSMAN, 2011):

- **Generalização:** é o processo da herança em que é criada uma superclasse a partir de subclasses que já existem. A generalização vai de algo mais detalhado, para algo mais amplo.
- **Especialização:** é o processo da herança em que são criadas subclasses a partir de uma superclasse que já existe. A especialização vai de algo mais amplo, para algo mais detalhado.

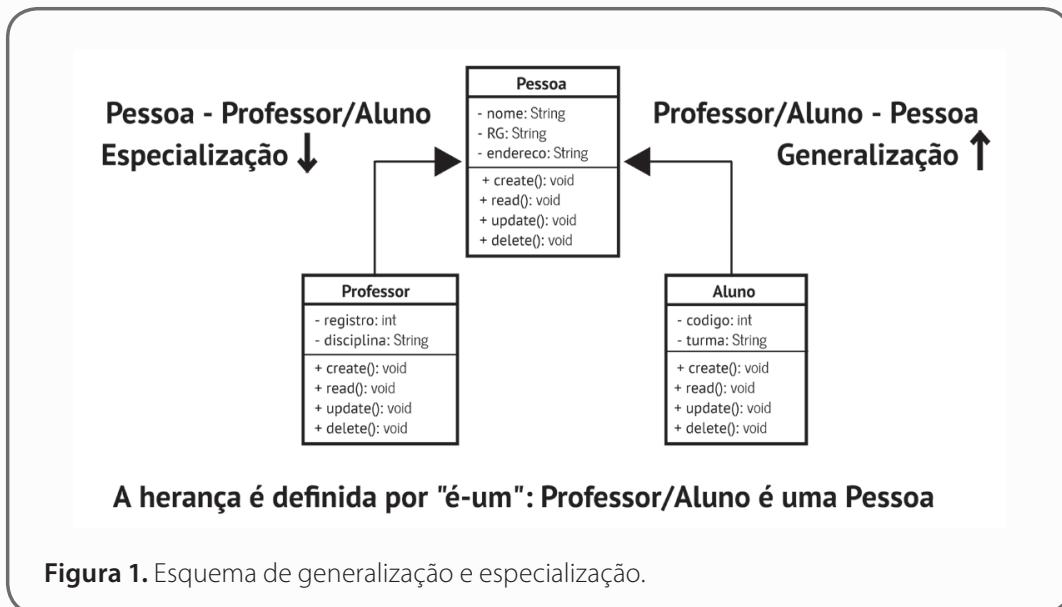


Figura 1. Esquema de generalização e especialização.



Fique atento

Tentar herdar uma classe declarada com o tipo final produzirá um erro em tempo de execução, apresentando a seguinte mensagem: **Fatal error: Class Estagiario may not inherit from final class (Funcionario)**

```

1 //Classe Funcionario declarada como final
2 final class Funcionario {
3 ...
4 }
5
6 //Classe Estagiario tentando fazer herança
7 require _ once "Funcionario.php";
8 class Estagiario extends Funcionario {
9 ...
10 }
```

Poliformismo

O polimorfismo é o que possibilita às classes abstratas fazerem a representação do comportamento das classes concretas que as referenciam. Dessa forma, um mesmo método pode apresentar diversas formas, de acordo com o contexto em que está inserido (PHP, c2001-2018b).

É pelo polimorfismo que duas ou mais subclasses de uma mesma superclasse são capazes de chamar métodos que têm a mesma identidade ou assinatura, mas possuem comportamentos distintos, que são especializados dependendo de cada classe. Também pelo polimorfismo, é possível que a subclassa faça a alteração de estados e comportamentos para a superclasse. Quando isso acontece, a subclassa sobrescreve os métodos da superclasse.

Em outras palavras, o polimorfismo é a capacidade que uma classe possui de sobrepor elementos da classe abstrata, geralmente seus métodos, modificando e sobrescrevendo-os. Para isso, é preciso que o nome e a assinatura do método sejam os mesmos, isto é, que eles tenham os mesmos tipos e quantidades de parâmetros.

Para exemplificar o polimorfismo vamos apresentar uma situação em que temos três classes: Empregado, Diretor e Gerente. A classe Empregado é a superclasse e as classes Diretor e Gerente herdam o atributo \$salario e o método calculaBonus.

```
1  <?php
2  class Empregado {
3      private $salario;
4
5      public function getsalario() {
6          return $this->salario;
7      }
8
9      public function setsalario($salario) {
10         $this->salario = $salario;
11     }
12
13     public function calculaBonus() {
14         return $this->salario * 1.1;
15     }
16 }
17 ?>
```

```

1  <?php
2  require_once "Empregado.php";
3
4  class Diretor extends Empregado {
5      public function calculaBonus() {
6          return $this->getSalario() * 2;
7      }
8  }
9 ?>

1  <?php
2  require_once "Empregado.php";
3
4  class Gerente extends Empregado {
5      public function calculaBonus() {
6          return $this->getSalario() * 1.5;
7      }
8  }
9 ?>
```

Após a implementação das classes, criamos o código apresentado na sequência para acessá-las. Declaramos uma variável `$salario` com valor 1000, que é passada para cada método calcular o valor do bônus conforme o objeto, ou seja, se o objeto for da classe `Empregado` o bônus será de 10%, se for da classe `Gerente`, será de 50%, e da classe `Diretor`, será 100%.

Note que em todas as classes o método possui o mesmo nome (`calculaBonus`). Isso significa que os métodos das classes filhas (`Diretor` e `Gerente`) estão reescrevendo o método da classe pai (`Empregado`) de acordo com a determinada regra de negócio.

```

1  <?php
2  require_once "Diretor.php";
3  require_once "Gerente.php";
4  require_once "Empregado.php";
5
6  $salarioAtual = 1000;
7
8  $diretor = new Diretor();
9  $diretor->setSalario($salarioAtual);
10 echo "Diretor: " . $diretor->calculaBonus() . "<br />";
11
12 $gerente = new Gerente();
13 $gerente->setSalario($salarioAtual);
14 echo "Gerente: " . $gerente->calculaBonus() . "<br />";
15
16 $empregado = new Empregado();
17 $empregado->setSalario($salarioAtual);
18 echo "Empregado: " . $empregado->calculaBonus() . "<br />";
19 ?>
```

A execução do código implementado apresenta o resultado ilustrado na Figura 2.

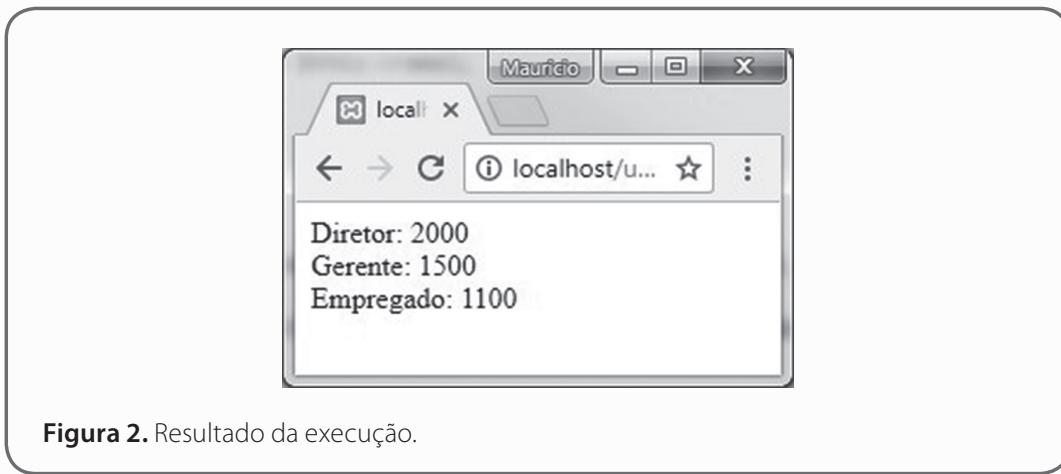


Figura 2. Resultado da execução.



Referências

DRAWKMAN. *Ilustração em vetor dos desenhos animados de cachorro fofo Dachshund de raça pura*. [S.I.]: Depositphotos, c2009-2018. Disponível em: <<https://br.depositphotos.com/112460892/stock-illustration-cartoon-vector-illustration-of-cute.html>>. Acesso em: 21 jan. 2018.

DVARG. *Cartoon puppy*. Budapest: Stockfresh, 2011. Disponível em: <<https://br.stockfresh.com/image/735048/cartoon-puppy>>. Acesso em: 21 jan. 2018.

ENGHOLM, H. J. *Engenharia de software na prática*. São Paulo: Novatec, 2010.

PHP. *Classes e objetos*. [S.I.]: The PHP Group, c2001-2018a. Disponível em: <http://php.net/manual/pt_BR/language.oop5.php>. Acesso em: 21 jan. 2018.

PHP. *Herança de objetos*. [S.I.]: The PHP Group, c2001-2018b. Disponível em: <http://php.net/manual/pt_BR/language.oop5.inheritance.php>. Acesso em: 21 jan. 2018.

PHP. *O básico*. [S.I.]: The PHP Group, c2001-2018c. Disponível em: <http://php.net/manual/pt_BR/language.oop5.basic.php>. Acesso em: 21 jan. 2018.

PRESSMAN, R. S. *Engenharia de software*. São Paulo: MacGraw-Hill, 2011.

ROCKSANA. *Cartoon smiling dark brown spotty puppy*. [S.I.]: Getty Images, c1999-2018. Disponível em: <https://www.gettyimages.com/detail/illustration/cartoon-smiling-dark-brown-spotty-puppy-royalty-free-illustration/473323964?esource=SEO_GIS_CDN_Redirect>. Acesso em: 21 jan. 2018.

SOMMERVILLE, I. *Engenharia de software*. São Paulo: Pearson, 2011.



PREZADO ESTUDANTE

**ENCERRA AQUI O TRECHO DO LIVRO DISPONIBILIZADO
PELA SAGAH PARA ESTA PARTE DA UNIDADE.**



Parte 3

Linguagem PHP Seguro

Linguagem PHP seguro

Objetivos de aprendizagem

Ao final deste texto, você deve apresentar os seguintes aprendizados:

- Aplicar a remoção de SQL Injection no PHP.
- Reconhecer criptografia no PHP.
- Descrever o padrão MVC no PHP.

Introdução

Segurança de dados é preocupação de qualquer administrador de sistemas, desenvolvedor de software ou gestor funcional. No desenvolvimento de sistemas com PHP, os profissionais envolvidos devem ficar atentos às melhores práticas existentes, visando aplicar os recursos que a linguagem fornece para desenvolver aplicações eficientes e seguras. Ao utilizar os recursos que se baseiam nos padrões consolidados no mercado, os sistemas desenvolvidos serão capazes de lidar com as principais formas de ataque, além de promover diversos benefícios que facilitam e agilizam o desenvolvimento de sistemas.

Neste capítulo, você estudará sobre remoção de SQL Injection, criptografia e o padrão de arquitetura de software model view controller (MVC).

SQL Injection

A expressão SQL Injection, ou simplesmente injeção de SQL, em livre tradução, é um tipo de ataque em que o invasor insere instruções SQL em campos de formulário que realizam operações em bancos de dados, por exemplo, consultas em telas de login ou inserção em formulários de fale conosco de sistemas (PHP, c2001-2018f).

O ataque é bem-sucedido quando o sistema apresenta vulnerabilidades que facilitam a execução de scripts pelo invasor, que normalmente ataca com as seguintes finalidades:

- **Acessar sistemas:** instruções específicas permitem entrar em sistemas desprotegidos sem ter cadastro para acessá-los. Ao acessar um sistema, o invasor pode extrair informações confidenciais ou até executar ações específicas, como autorizar pedidos, renovar contratos, etc.
- **Quebrar sistemas:** por meio de determinadas instruções, o invasor consegue excluir ou modificar registros na base de dados do sistema invadido, podendo proporcionar falhas ou até tirar o sistema do ar.

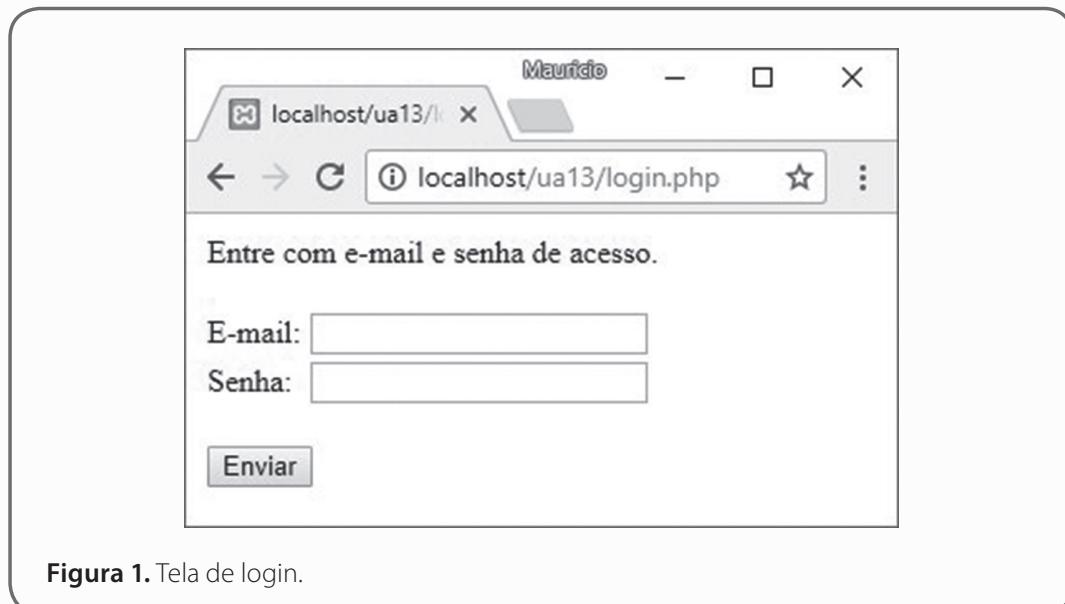
Ao pegar como exemplo a tela de login de um sistema qualquer desenvolvido em PHP, composta pelos campos e-mail e senha, considera-se que a expressão SQL a seguir, ou algo semelhante, seja executada para validar os dados do usuário para acessar o sistema.

```
1 SELECT nome
2   FROM usuario
3 WHERE email = '$email'
4   AND senha = '$senha'
```

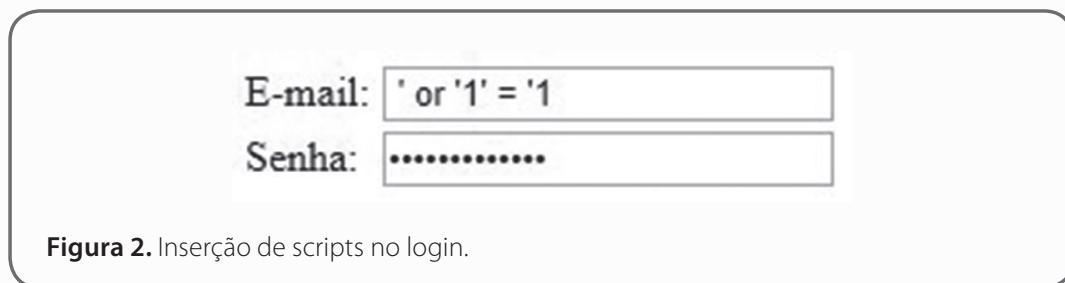
O conteúdo das variáveis \$email e \$senha é, então, enviado ao banco de dados para realizar a validação, verificando se o usuário informado está cadastrado e se a sua senha está correta. No entanto, um usuário mal-intencionado pode enviar instruções SQL em vez de os dados de usuário e senha, que serão executados no banco de dados.

Ao receber as instruções SQL, em vez de dados de usuário e senha, um sistema em PHP vulnerável pode executar essas instruções e comprometer a confidencialidade e a confiabilidade das informações armazenadas no banco de dados.

Para representar os casos de invasão pela tela de login de um sistema vulnerável, apresentamos a tela da Figura 1, que contém os campos usuário e senha do tipo string.

**Figura 1.** Tela de login.

Para realizar o acesso em um sistema vulnerável, sem possuir um usuário cadastrado, o invasor pode inserir scripts que burlem as instruções SQL do formulário, conforme o modelo apresentado na Figura 2.

**Figura 2.** Inserção de scripts no login.

Nesse caso, a instrução SQL executada pelo aplicativo passa a ser:

```

1 SELECT nome
2 FROM usuario
3 WHERE email = '' or '1' = '1'
4 AND senha = '' or '1' = '1'

```

Dessa forma, a instrução SQL retorna todos os registros da tabela `usuario`, pois as condições verificadas para `email` e `senha` são válidas, isto é, `email` e `senha` são vazios ou 1 é igual a 1. Assim, o invasor consegue acessar o sistema pelo primeiro usuário que for trazido pela consulta no banco de dados.

O prejuízo pode ser ainda maior se o invasor souber o e-mail de um usuário que seja administrador do sistema, assim ele passa esse e-mail no campo email e insere a expressão SQL apenas na senha, conseguindo acesso privilegiado ao sistema.

```
1 SELECT nome
2   FROM usuario
3 WHERE email = 'administrador@site.com.br'
4   AND senha = '' or '1' = '1'
```

Além disso, dependendo da forma como a execução do programa PHP realiza os comandos de acesso ao banco de dados, instruções SQL mais perigosas podem ser executadas, como excluir tabelas, retirar privilégios, etc.

Na Figura 3, você pode observar o encerramento da instrução SELECT, com o caractere ponto e vírgula, e o início outra instrução SQL, que exclui a tabela usuario do banco de dados. Os caracteres “--” representam um comentário em SQL e indicam que o restante da instrução SQL deve ser ignorado.

E-mail:

Senha:

Figura 3. Injeção de SQL no formulário de login.



Fique atento

É importante que o administrador de banco de dados (DBA) crie usuários de banco de dados específicos para rodar as instruções SQL da aplicação, com privilégios limitados apenas às necessidades de cada funcionalidade. No caso de injeção de SQL pelo formulário de login de um sistema vulnerável, o invasor conseguiria rodar apenas instruções que consultam registros, sem privilégios para modificar os registros ou a estrutura das tabelas.

Evitar SQL Injection no PHP

O PHP fornece diversas alternativas para evitar a injeção de scripts SQL nas páginas web. O tratamento deve ser realizado nas variáveis que recebem os campos dos formulários antes de executar as instruções SQL no banco de dados, conforme será apresentado nos itens a seguir (PHP, c2001-2018f).

Declarações preparadas

Sendo uma das formas mais robustas para evitar injeção de SQL nas páginas, as variáveis enviadas ao banco de dados são tratadas internamente pelo PHP, removendo qualquer instrução que possa proporcionar risco de execução de *scripts*.

Para usar declarações preparadas nas expressões SQL no PHP, você deve utilizar as funções `bindParam` ou `bindValue`, que realizam a parametrização dos campos oriundos dos formulários, conforme modelo a seguir (W3IM, c2017):

```
1 $stmt = $conn->prepare("SELECT email, nome, senha
2                         FROM usuario
3                         WHERE email = :email
4                         AND senha = :senha");
5
6 $stmt->bindParam(':email', $email, PDO::PARAM_STR);
7 $stmt->bindParam(':senha', $senha, PDO::PARAM_STR);
8 $stmt->execute();
```

Validação de campos

As funções `filter_*` servem para validar e filtrar dados de variáveis postadas por formulários. Existem diversas opções de filtros, que podem validar tipos de dados, endereços de URL, endereços IP, endereços de e-mail, **strings**, caracteres especiais e outros (PHP, c2001-2018c). No exemplo a seguir, usaremos função `filter_var` para garantir que o endereço de e-mail fornecido pelo formulário seja válido.

```
1 if (filter_var($email, FILTER_VALIDATE_EMAIL));
2 ...
```

As funções `ctype_*` realizam a verificação de tipo de caractere das variáveis que são postadas nos formulários. Diversas opções estão disponíveis, como a verificação de caracteres alfanuméricos, alfabeticos, numéricos, de controle, imprimíveis, etc. (PHP, c2001-2018d).

No exemplo a seguir, apresentamos a função `ctype_digit` para verificar se todos os caracteres da variável `$numero` são numéricos. Essa validação é recomendada para validar campos que devem conter apenas dígitos numéricos.

```
1 $numero = '12345684001';
2 if (ctype_digit($numero))
3 ...
```

Criptografia

Em relação à segurança de dados, a criptografia ocupa um papel importante no sentido de proteger as informações tratadas pelas aplicações. Essa proteção envolve métodos e técnicas que embaralham e tornam as informações ilegíveis, impedindo o acesso para quem não está autorizado.

O PHP fornece diversas opções de criptografia que envolvem algoritmos poderosos e consagrados, que podem ser usados por meio de funções de fácil utilização. No entanto, apenas criptografar os dados não é suficiente para obter uma boa segurança, ou seja, é preciso ficar atento a outros fatores, como a escolha de uma senha forte.

Senha forte

De nada adianta implementar criptografia de uma senha fraca, ou seja, uma senha como uma sequência de números de 1 a 6 ou que contenha palavras conhecidas. Diversos programas na internet possuem um banco de senhas, com milhares de opções predefinidas para diversos tipos de criptografia.

Esses programas realizam o ataque de força bruta (*Brute Force Attack*), em que diversas opções de combinações de senhas são tentadas, incessantemente, com o objetivo de achar uma senha que coincida com a verdadeira. Esse tipo de ataque pode levar muitas horas, dias ou até semanas, dependendo do poder computacional do atacante (RICOBA, 2017).

Palavras como teste, admin, administrador, root, senha, nomes próprios, raças de animais, etc. podem ser facilmente descobertas, portanto, recomenda-se usar senhas fortes, que mesclam caracteres especiais, números e letras maiúsculas e minúsculas.



Fique atento

Um sistema web seguro regista o número de tentativas de login dos usuários e os bloqueia assim que uma quantidade de tentativas de acesso, sem sucesso, é realizada em determinado espaço de tempo. Isso serve como medida de segurança contra os ataques de força bruta.

Na criptografia considera-se importante o tempo computacional utilizado para codificar os dados. Isso significa que métodos mais rápidos tendem a produzir resultados inferiores aos métodos mais demorados, que utilizam algoritmos mais robustos.

Nesse sentido, o desenvolvedor deve avaliar o custo-benefício no momento de decidir qual método de criptografia utilizar na aplicação, isto é, um método não tão forte, mas capaz produzir um rápido resultado, ou um método mais eficaz que consome maior tempo de processamento.

Métodos de criptografia no PHP

O PHP dispõe de diversos métodos para criptografar strings, como campos de formulário, expressões de texto e palavras de um modo geral. Esses métodos implementam vários algoritmos de criptografia, entre os quais muitos são robustos, seguros e amplamente divulgados e aceitos pela comunidade de desenvolvimento de software. Veja a seguir os principais métodos de criptografia disponíveis no PHP.

Md5

O método md5 gera uma chave alfanumérica de 32 caracteres, independentemente da expressão a ser criptografada possuir um ou mais caracteres, mesmo que tenha mais que 32 caracteres. Esse método é considerado um dos mais rápidos para criptografar informações, contudo, devido ao alto

poder computacional atual, pode ser decifrado por força bruta em tempo razoável de processamento (PHP, c2001-2018g). Observe o exemplo a seguir e a Figura 4.

```
1 <?php
2     $senha= 'T25ss#+';
3     $senha_cripto= md5($senha);
4     echo "Senha: $senha <br />";
5     echo "md5: $senha_cripto";
6 ?>
```

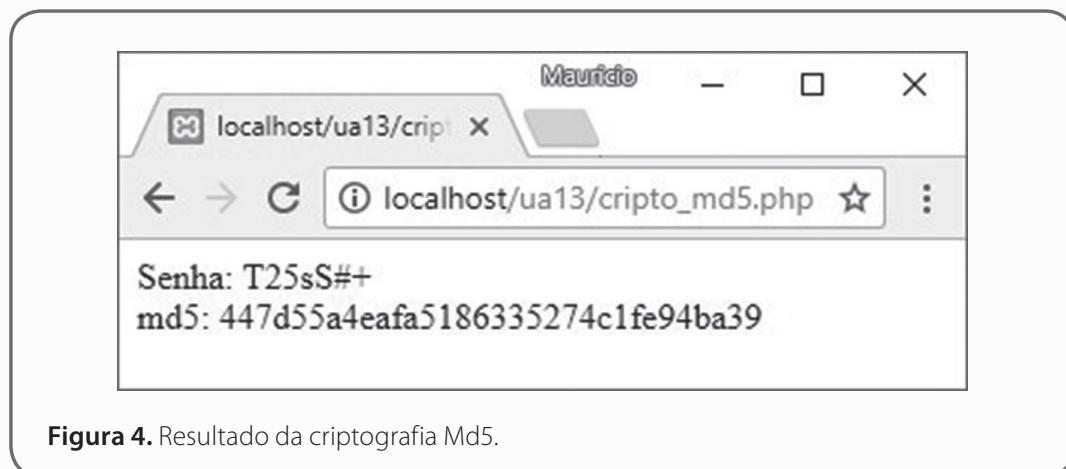


Figura 4. Resultado da criptografia Md5.

O método md5 não possui uma forma de decifrar a expressão criptografada. Dessa forma, é preciso criptografar a expressão e comparar o resultado com a expressão que está cifrada.

Sha1

Assim como o md5, o método sha1 também é rápido e de mão única, ou seja, não possui uma forma de decifrar uma expressão criptografada. O resultado desse método produz uma chave alfanumérica de 40 caracteres, conforme apresentado a seguir e ilustrado na Figura 5 (PHP, c2001-2018i).

```
1 <?php
2     $senha= 'T25ss#+';
3     $senha_cripto= sha1($senha);
4     echo "Senha: $senha <br />";
5     echo "sha1: $senha_cripto";
6 ?>
```

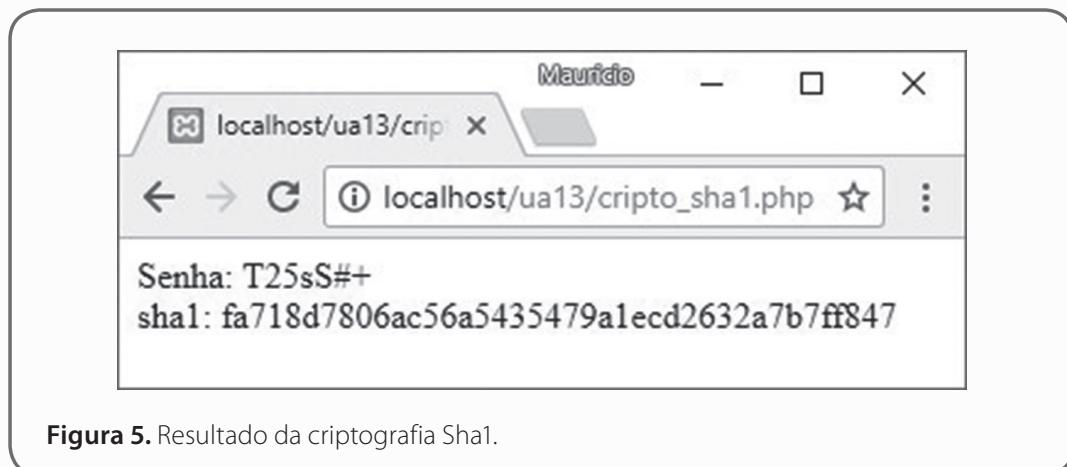


Figura 5. Resultado da criptografia Sha1.

Base64

Para base64, o PHP fornece dois métodos, um que codifica e outro que decodifica. Por isso, ele é considerado um método de mão dupla, pois consegue reproduzir a expressão original a partir da string codificada, como você pode ver a seguir e na Figura 6 (PHP, c2001-2018a).

```

1 <?php
2     $senha= 'T25sS#+';
3     $senha_cripto= base64_encode($senha);
4     echo "Senha: $senha <br />";
5     echo "base64: $senha_cripto";
6 ?>

```

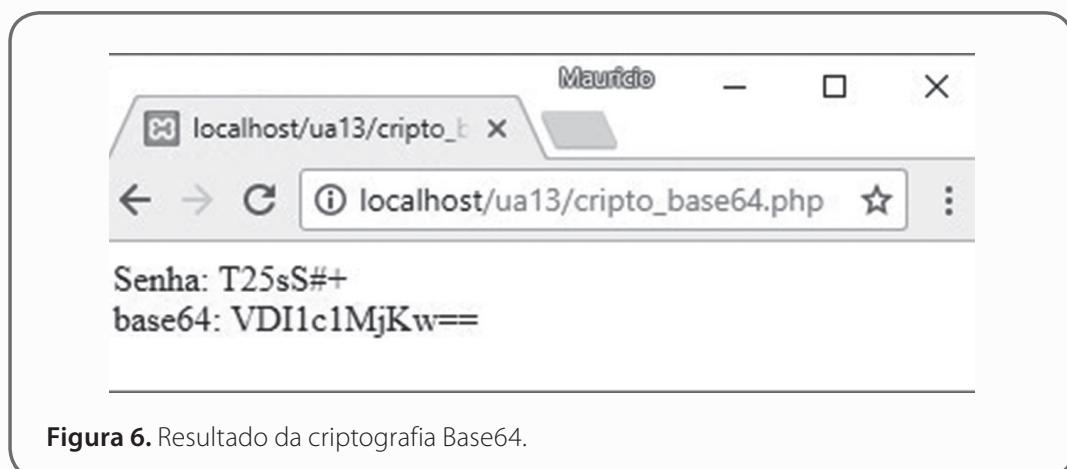


Figura 6. Resultado da criptografia Base64.

Para decodificar uma expressão criptografada com base64, utilize a função `base64_decode`:

```
echo base64_decode($senha_cripto);
```

Crypt

Esse método codifica uma expressão com base em diversas opções de algoritmos de criptografia que variam de acordo com o sistema operacional em uso. O segundo parâmetro da função, que é opcional, define o `salt`. Os seguintes tipos de codificação podem ser utilizados, conforme disponibilidade do sistema operacional (PHP, c2001-2018b):

- **CRYPT_STD_DES:** codificação standard DES-based, que utiliza um `salt` de 2 caracteres.
- **CRYPT_EXT_DES:** codificação extended DES-based, que utiliza um `salt` de 9 caracteres.
- **CRYPT_MD5:** codificação MD5, que utiliza um `salt` de 12 caracteres iniciando, necessariamente, com `1`.
- **CRYPT_BLOWFISH:** codificação Blowfish que utiliza um `salt` de 16 caracteres iniciando, necessariamente, com `2`.

Se o segundo parâmetro for omitido (`salt`), o PHP irá gerar um valor aleatório fazendo, a cada execução, uma expressão codificada diferente ser gerada. Observe o exemplo a seguir e a Figura 7.

```
1 <?php
2     $senha= 'T25ss#+';
3     echo "Senha: $senha <br />";
4     for ($i= 1; $i <= 3; $i++) {
5         $senha_cripto= crypt($senha);
6         echo "crypt: $senha_cripto <br />";
7     }
8 ?>
```

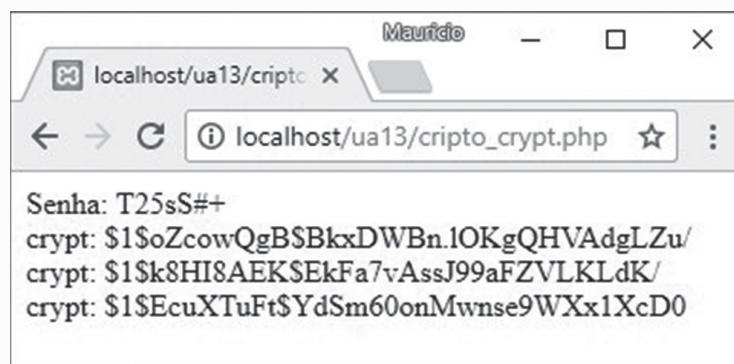


Figura 7. Resultado da criptografia Crypt.

Para conferir uma expressão criptografada, deve-se utilizar a seguinte instrução no PHP:

```

1 if (crypt($senha, '$1$K8HI8AEK$EkFa7vAssJ99aFZVLKLdK/'))
2     echo 'certo';
3 else
4     echo 'errado';

```

Password_hash

É um método do PHP compatível com o crypt, cuja expressão resultante (hash) produz 60 caracteres de comprimento a cada nova execução. Deve utilizar um dos seguintes parâmetros, que especificam os algoritmos utilizados para realizar a criptografia (PHP, c2001-2018h).

- **PASSWORD_DEFAULT**: utiliza o algoritmo bcrypt, que é o algoritmo padrão definido pelo PHP desde a versão 5.5.0.
- **PASSWORD_BCRYPT**: utiliza o algoritmo crypt _ blowfish, que aplica o identificador \$2y\$, compatível com o crypt.

Veja o exemplo a seguir e sua representação na Figura 8.

```

1 <?php
2     $senha= 'T25ss#';
3     echo "Senha: $senha <br />";
4     for ($i= 1; $i <= 3; $i++) {
5         $senha_cripto= password_hash($senha, PASSWORD_DEFAULT);
6         echo "password_hash: $senha_cripto <br />";
7     }
8 ?>

```



Figura 8. Resultado da criptografia Password_hash.

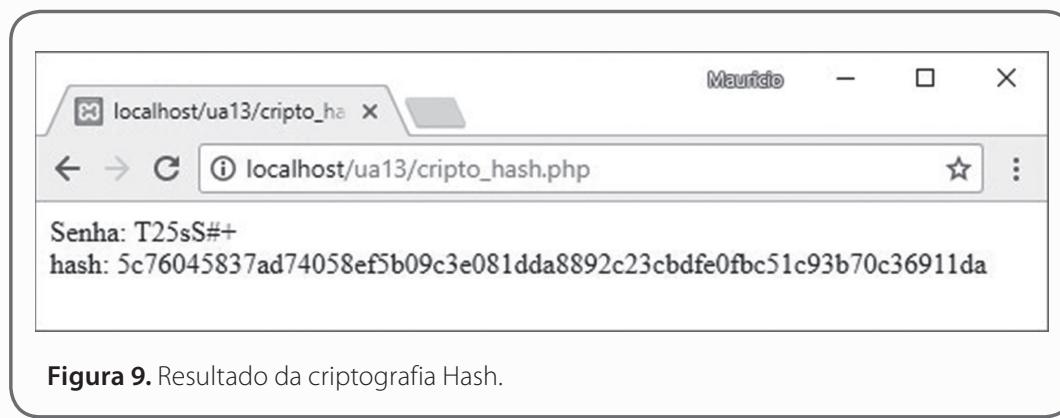
Para comparar uma string com uma expressão criptografada, utiliza-se a seguinte instrução no PHP:

```
1 if (password_verify($senha,
2     '$2y$10$GQEzKFQ68xP8apB236aUIuMGvqVtBBhk2knvL7REE4Lb5iCn56P.0'))
3     echo 'certo';
4 else
5     echo 'errado';
```

Hash

Esse método permite a seleção de um algoritmo de criptografia para gerar a expressão criptografada de uma string. Três parâmetros estão disponíveis: o primeiro define o algoritmo a ser utilizado; o segundo, a expressão original; e o terceiro, que é opcional, especifica TRUE para dados binários brutos e FALSE para hexadecimais minúsculos, sendo este último o padrão, conforme você pode observar a seguir e na Figura 9 (PHP, c2001-2018e).

```
1 <?php
2     $senha= 'T25sS#+';
3     echo "Senha: $senha <br />";
4     $senha_cripto= hash('sha256', $senha);
5     echo "hash: $senha_cripto";
6 ?>
```



Diversos algoritmos estão disponíveis para o método `hash`, como `md5`, `sha512`, `gost`, `crc32` e outros. Uma lista completa dos algoritmos suportados pode ser obtida por meio da função `hash _ algos` do PHP, conforme script a seguir.

```

1   foreach (hash_algos() as $algoritmo) {
2       echo "$algoritmo <br />";
3   }

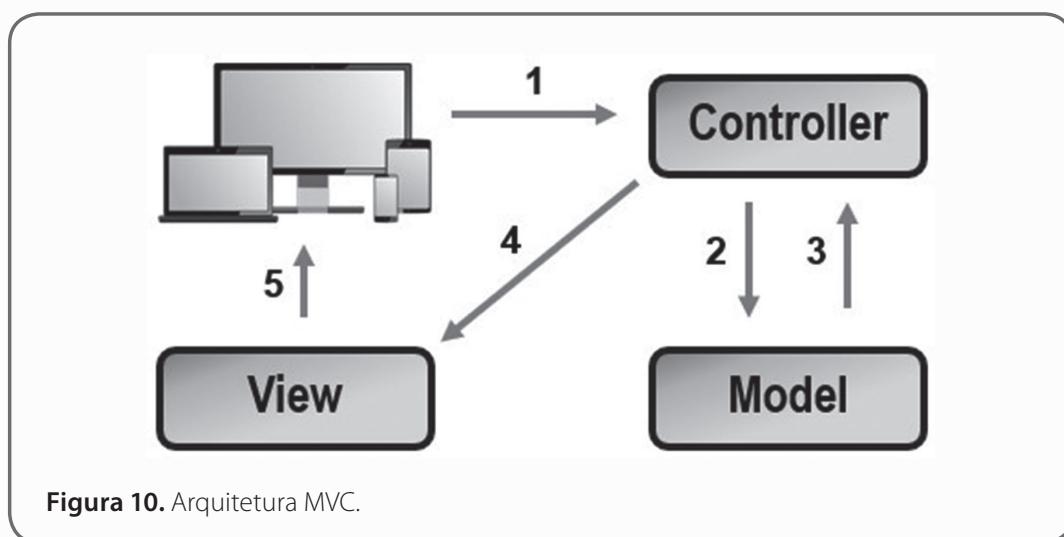
```

Padrão Model View Controller

O padrão MVC é definido por uma arquitetura de software que realiza a separação da estrutura das aplicações em três camadas que interagem entre si, sendo cada uma delas responsável por determinadas atividades.

Esse padrão foi criado em meados da década de 1970, com o objetivo de organizar os projetos de software, bem como facilitar a manutenção nos sistemas por meio da separação de responsabilidades, em que cada camada é responsável por uma ou mais atividades, como acesso a dados, execução de regras de negócio, controle das requisições, apresentação de informações ao usuário, entre outras (PITT, 2012).

As camadas do padrão MVC, model, view e controller, atuam em conjunto por meio de troca de informações, conforme ilustrado na Figura 10.



O processo inicia com uma requisição do usuário (1), em que a camada controller envia a demanda à camada model (2) que, por sua vez, processa e devolve uma resposta à camada controller (3). Na sequência, a camada controller envia essa resposta à camada view (4), que se encarrega de apresentar o resultado ao usuário (5).

Com base nessa representação, destacam-se os seguintes benefícios da arquitetura em três camadas MVC (PITT, 2012):

- **Padronização da estrutura dos sistemas:** os sistemas são desenvolvidos de forma estruturada e padronizada, fazendo todos os desenvolvedores utilizarem a mesma forma de desenvolvimento.
- **Diminuição da complexidade do código-fonte:** dividido em camadas, cada parte do sistema fica responsável por determinada tarefa, diminuindo, assim, a complexidade dos módulos desenvolvidos.
- **Facilidade de manutenção de sistemas:** ao trabalhar em determinada camada, o desenvolvedor atua especificamente em determinadas partes do sistema, sem afetar áreas que não estão envolvidas, por exemplo, é possível modificar regras de negócio sem mexer nas telas de apresentação.
- **Modularização dos sistemas:** a arquitetura em camadas facilita a modularização de sistemas. Com isso, profissionais podem se especializar em determinadas partes ou camadas, promovendo maior produtividade.
- **Independente de plataforma:** esse padrão permite desenvolver sistemas para qualquer plataforma, como sistemas web, desktop, dispositivos móveis, etc.
- **Padrão de mercado:** utilizar arquitetura em três camadas significa seguir um padrão aprovado e bastante difundido, baseado nas melhores práticas utilizadas na academia e na indústria de desenvolvimento de software.



Fique atento

O padrão arquitetural MVC, atualmente, é muito difundido entre as melhores práticas para desenvolvimento de sistemas, em especial para aplicações web desenvolvidas em PHP, e diversos frameworks o implementam, como Zend, Laravel, Symfony, CakePHP e CodeIgniter, entre outros.

Model

A camada model define a lógica da implementação do sistema, pois contém as regras de negócio, as rotinas de acesso aos dados, as classes, etc. Essa camada espera as requisições da camada controller, que demanda atividades a serem executadas, como uma pesquisa no banco de dados, por exemplo (PITT, 2012).

Assim que recebe uma requisição, a camada model processa determinada atividade e envia o resultado de volta à camada controller. Essa camada não recebe requisições diretas que não sejam da controller, nem se preocupa com a apresentação dos dados.

São exemplos de atividades da camada model:

- salvar e recuperar registros no banco de dados;
- salvar arquivos no servidor, como fotos, planilhas, entre outros;
- implementar regras de negócio, como fórmulas, cálculos e operações com dados.

View

Essa é a camada responsável pela apresentação dos dados ao usuário, ou seja, a view recebe da camada controller as informações que foram processadas pela model. Para a view, não importa de onde ou como os dados foram processados, sua responsabilidade está apenas em apresentar as informações (PITT, 2012).

A apresentação dos dados não se limita a páginas HTML que exibem informações como relatórios, listas, tabelas e grades tabuladas, essa apresentação pode ser representada de diferentes modos e formatos, por exemplo, planilhas, arquivos PDF, vídeos ou qualquer outro meio que disponibilize informações.

Em sistemas PHP, a camada view pode apresentar:

- uma relação de registros filtrados por meio de uma pesquisa;
- uma lista de arquivos disponíveis para download;
- um formulário para entrada de dados de um cadastro;
- um relatório de movimentação financeira.

Controller

A camada controller atua na recepção das requisições dos usuários e gerencia a troca de informações entre as camadas model e view. A camada controller identifica cada tipo de requisição do usuário e envia diretamente aos processos responsáveis pela execução na model (PITT, 2012).

Após o processamento da camada model, a controller recebe o dado trabalhado e o distribui diretamente para o processo responsável pela apresentação ao usuário na camada view. Assim, a camada controller atua como gerenciadora da comunicação entre as camadas da aplicação, delegando responsabilidades.

São exemplos de atividades da camada controller em um sistema PHP:

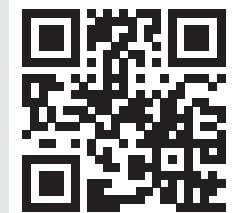
- receber cada requisição e identificar o processo responsável por sua execução;
- controlar a validade da sessão do usuário dentro da aplicação;
- impedir o acesso direto às páginas do sistema.



Link

Confira o padrão arquitetural MVC no link ou código a seguir (MEDEIROS, c2018).

<https://goo.gl/1CV5an>





Referências

MEDEIROS, H. *Introdução ao padrão MVC*. Rio de Janeiro: DevMedia, c2018. Disponível em: <<https://www.devmedia.com.br/introducao-ao-padroao-mvc/29308>>. Acesso em: 03 jan. 2018.

PHP. *base64_encode*. [S.I.]: The PHP Group, c2001-2018a. Disponível em: <http://br.php.net/base64_encode>. Acesso em: 21 jan. 2018.

PHP. *crypt*. [S.I.]: The PHP Group, c2001-2018b. Disponível em: <https://secure.php.net/manual/pt_BR/function.crypt.php>. Acesso em: 21 jan. 2018.

PHP. *Filtragem de dados*. [S.I.]: The PHP Group, c2001-2018c. Disponível em: <https://secure.php.net/manual/pt_BR/book.filter.php>. Acesso em: 21 jan. 2018.

PHP. *Funções pra verificação de tipo de caractere*. [S.I.]: The PHP Group, c2001-2018d. Disponível em: <https://secure.php.net/manual/pt_BR/book ctype.php>. Acesso em: 21 jan. 2018.

PHP. *hash*. [S.I.]: The PHP Group, c2001-2018e. Disponível em: <https://secure.php.net/manual/pt_BR/function.hash.php>. Acesso em: 21 jan. 2018.

PHP. *Injeção de SQL*. [S.I.]: The PHP Group, c2001-2018f. Disponível em: <http://php.net/manual/pt_BR/security.database.sql-injection.php>. Acesso em: 21 jan. 2018.

PHP. *md5*. [S.I.]: The PHP Group, c2001-2018g. Disponível em: <<http://br.php.net/md5>>. Acesso em: 21 jan. 2018.

PHP. *password_hash*. [S.I.]: The PHP Group, c2001-2018h. Disponível em: <http://php.net/manual/pt_BR/function.password-hash.php>. Acesso em: 21 jan. 2018.

PHP. *sha1*. [S.I.]: The PHP Group, c2001-2018i. Disponível em: <<http://br.php.net/sha1>>. Acesso em: 21 jan. 2018.

PITT, C. *Pro PHP MVC: everything you need to know about using MVC with PHP in a single reference*. New York: Apress, 2012.

RICOBA. *Ataque de força bruta*. [S.I.]: MyCyberSecurity, 2017. Disponível em: <<http://www.mycybersecurity.com.br/glossario/ataque-de-forca-bruta/>>. Acesso em: 20 jan. 2018.

W3IM. *PHP declarações preparadas*. [S.I.], c2017. Disponível em: <http://www.w3im.com/pt/php/php_mysql_prepared_statements.html>. Acesso em: 18 jan. 2018.



PREZADO ESTUDANTE

**ENCERRA AQUI O TRECHO DO LIVRO DISPONIBILIZADO
PELA SAGAH PARA ESTA PARTE DA UNIDADE.**



Parte 4

Linguagem PHP com Framework

Linguagem PHP com framework

Objetivos de aprendizagem

Ao final deste texto, você deve apresentar os seguintes aprendizados:

- Definir a instalação do framework Zend.
- Descrever os principais comandos do framework Zend.
- Reconhecer exemplos de aplicações do framework Zend.

Introdução

O desenvolvimento de sistemas web é uma atividade desafiadora que demanda inúmeros recursos, principalmente o tempo de implementação, que tem relação direta com o custo de produção. Além disso, não são raras as vezes em que analistas e programadores implementam rotinas sem atender às boas práticas do mercado, produzindo sistemas sem padronização e sujeitos a falhas, erros e atrasos. Para atuar nessa problemática, organizações de diversos tipos criaram frameworks para desenvolvimento de sistemas, que são pacotes de software que atendem a determinados domínios de aplicação, com o objetivo de organizar a estrutura dos sistemas e proporcionar agilidade e ganho de produtividade no desenvolvimento de software.

Neste capítulo, você estudará sobre a instalação do framework Zend, seus principais comandos e exemplos de aplicações.

Framework Zend

Zend é um **framework** para desenvolvimento de sistemas e serviços em PHP que possui código aberto e 100% orientado a objetos. É disponibilizado sob a licença New BSD License, e o principal patrocinador é a organização Zend. Contudo, várias outras empresas já contribuíram com componentes ou recursos

importantes, como Google, Microsoft e Strikelron (ZEND FRAMEWORK, c2006-2018a).

Zend é compatível com a versão 5.6 ou superior do PHP e pode ser instalado em qualquer pacote ou distribuição do PHP, como XAMPP, WAMP, ZWAMP, EASYPHP e outros, tanto para Windows como para Linux, Mac OSX, Solaris, etc. (ZEND FRAMEWORK, c2006-2018a).



Fique atento

No desenvolvimento de software, um **framework** compreende um conjunto de classes e/ou funções, implementadas em uma ou mais linguagens de programação, criadas para atender determinados domínios de aplicação de forma padronizada, proporcionando maior produtividade e agilidade às equipes de desenvolvimento de sistemas (BIERER; HUSSAIN; JONES, 2016).

Instalação

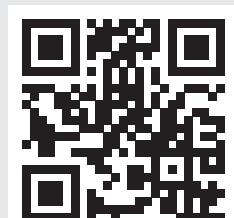
Neste capítulo, vamos utilizar o framework Zend 3 em conjunto ao XAMPP para Windows com PHP 7. A instalação do Zend deve ser realizada com o apoio da ferramenta Composer, que gerencia as dependências do PHP, isto é, o Composer baixa pacotes, descompacta e instala os arquivos necessários do framework.



Link

Para instalar o Composer, acesse o site oficial da ferramenta e efetue o download para o seu sistema operacional. Após realizar o download, faça a instalação seguindo os passos solicitados. A ferramenta Composer está disponível no seguinte link:

<https://goo.gl/u1HxYa>



Não é necessário mudar nenhuma configuração das opções apresentadas durante a instalação. Siga os passos clicando em **Next** e informe o caminho da instalação do PHP quando solicitado, conforme apresentado na Figura 1.

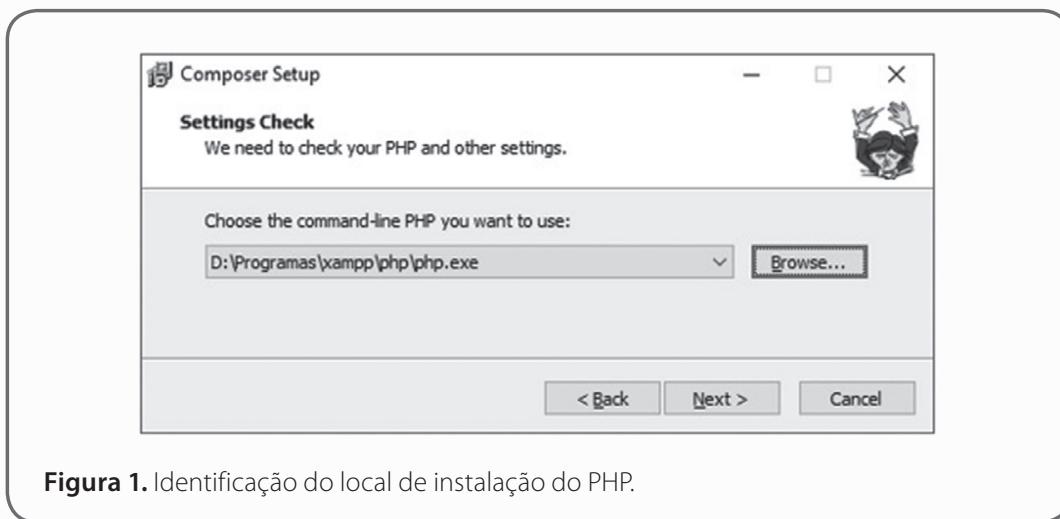


Figura 1. Identificação do local de instalação do PHP.

Confirme as configurações selecionadas e clique em **Install** para concluir a instalação, conforme ilustrado na Figura 2.

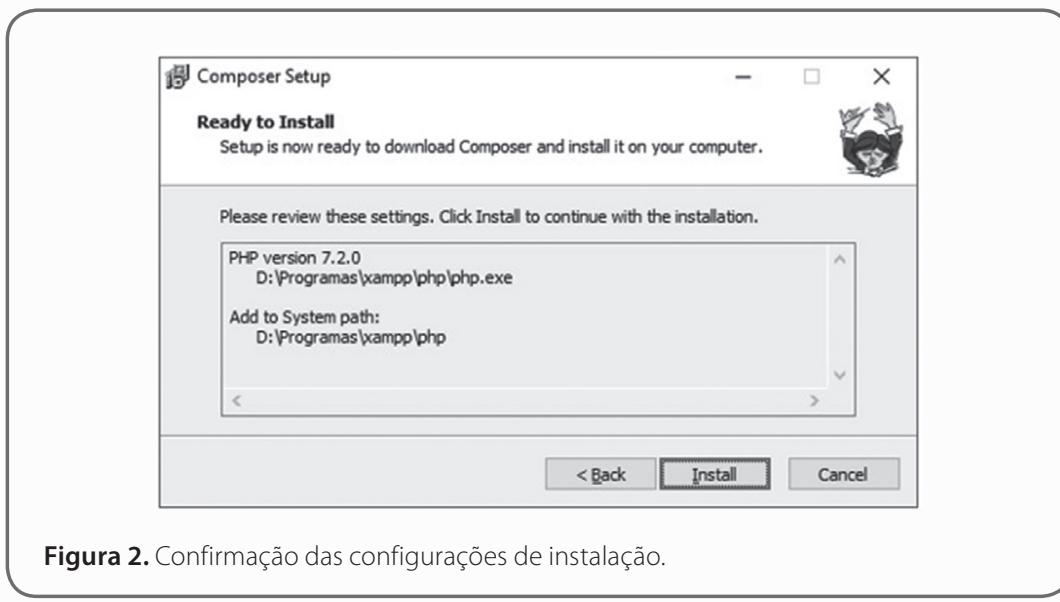


Figura 2. Confirmação das configurações de instalação.

Após concluir a instalação, abra uma janela **prompt** para executar linhas de comando no seu computador. No Windows, clique no botão **Iniciar** e selecione o programa **Prompt de comando** ou clique em **Iniciar** e selecione **Executar** para digitar o comando **cmd**.

Digite `composer` na linha de comando do *prompt* para confirmar se o Composer foi instalado corretamente. Em versões anteriores ao PHP 7 ou ao Windows 10, pode ser necessário realizar configurações manualmente no arquivo `php.ini` e nas variáveis de ambiente do Windows. Nesse caso, consulte a documentação da ferramenta para maiores informações. A Figura 3 ilustra a instalação do Composer.



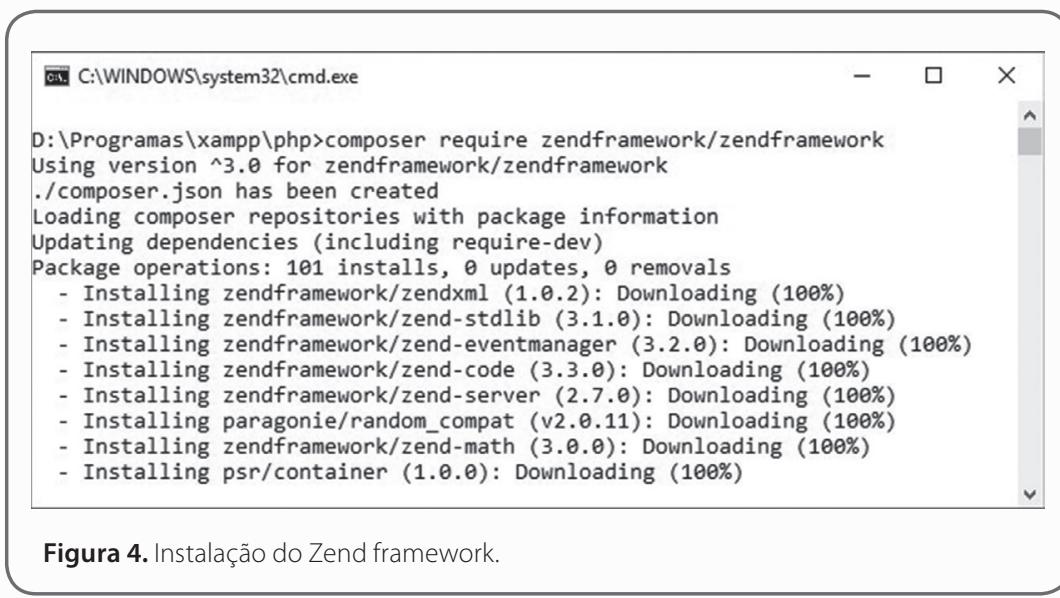
A screenshot of a Windows Command Prompt window titled "C:\WINDOWS\system32\cmd.exe". The window shows the command "D:\Programas\xampp\php>composer" followed by a decorative loading graphic consisting of a grid of diagonal lines forming a diamond shape. At the bottom of the window, the text "Composer version 1.6.2 2018-01-05 15:28:41" is displayed.

Figura 3. Confirmação da instalação do Composer.

No Prompt de comando, digite a seguinte expressão para realizar a instalação completa do *framework* Zend 3, composta de todos os seus pacotes:

`D:\Programas\xampp\php>composer require zendframework/zendframework`

A Figura 4 ilustra a instalação do Zend *framework*.



A screenshot of a Windows Command Prompt window titled "C:\WINDOWS\system32\cmd.exe". The window shows the command "D:\Programas\xampp\php>composer require zendframework/zendframework". The output includes: "Using version ^3.0 for zendframework/zendframework", "./composer.json has been created", "Loading composer repositories with package information", "Updating dependencies (including require-dev)", "Package operations: 101 installs, 0 updates, 0 removals", and a list of 101 packages being installed, each with a progress bar indicating 100% download.

Figura 4. Instalação do Zend framework.



Fique atento

É possível instalar pacotes do Zend separadamente. A instalação de pacotes independentes pode ser realizada tanto pelo Composer como pelo GitHub.

Após a execução da linha de comando, o Composer realiza a instalação do framework Zend 3, composto por 61 pacotes, descritos a seguir.

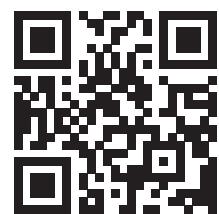
- Zend-authentication: ferramenta para autenticação de usuários.
- Zend-barcode: renderização de códigos de barras em imagens ou arquivos PDF.
- Zend-captcha: geração e validação de Captcha para formulários.
- Zend-crypt: ferramentas robustas para criptografia.
- Zend-db: camada de abstração de acesso a banco de dados.
- Zend-mvc: implementa arquitetura model view controller (MVC) no desenvolvimento de aplicações.



Link

Saiba mais sobre os pacotes que compõem o framework Zend no seguinte link (ZEND FRAMEWORK, c2006-2018b):

<https://goo.gl/1SJT Xt>



Principais comandos do framework Zend

Após a instalação do framework, realizada na seção anterior, iniciaremos a criação de um projeto. O Zend 3 fornece o Skeleton Application, que serve como modelo ou esqueleto para um projeto qualquer. A criação desse projeto deve ser realizada pelo seguinte comando (ZF TUTORIALS, 2006-2017):

```
Composer create-project -s dev zendframework/skeleton-application
d:\programas\xampp\htdocs\ua15
```

Esse comando realiza a instalação do esqueleto do projeto no diretório apresentado a seguir e representado na Figura 5. No entanto, no seu computador, você pode escolher uma outra pasta qualquer.

```
d:\programas\xampp\htdocs\ua15
```



```
C:\WINDOWS\system32\cmd.exe - Composer create-project -s dev zendframework/skeleton-application... - □ X
D:\Programas\xampp\php>Composer create-project -s dev zendframework/skeleton-application
d:\programas\xampp\htdocs\ua15
Installing zendframework/skeleton-application (dev-master 7581703d5979c090b74905c5c7f5e7f
2c527c980)
- Installing zendframework/skeleton-application (dev-master master): Cloning master
```

Figura 5. Execução do comando para criar o projeto Skeleton.

Assim que o esqueleto do projeto for instalado, a ferramenta inicia uma série de perguntas a respeito do restante da configuração. Nessa instalação, vamos escolher algumas opções, contudo, você pode selecionar opções diferentes, caso já tenha mais familiaridade com o framework:

Do you want a minimal install (no optional packages)? Y/n

Deseja uma instalação mínima (sem pacotes opcionais)?

Selecione “n” para não instalar o mínimo.

Would you like to install the developer toolbar? y/N

Deseja instalar a barra de ferramentas do desenvolvedor?

Selecione “y” para instalar.

Would you like to install caching support? y/N

Deseja instalar suporte ao cache?

Selecione “n” para não instalar.

Would you like to install database support (installs zend-db)? y/N

Deseja instalar suporte ao banco de dados?

Selecione “y” para instalar.

Would you like to install forms support? y/N

Deseja instalar suporte a formulários?

Selecione “y” para instalar.

Would you like to install JSON de/serialization support? y/N

Deseja instalar suporte à serialização JSON?

Selecione “n” para não instalar.

Would you like to install logging support? y/N

Deseja instalar suporte ao log?

Selecione “n” para não instalar.

Would you like to install MVC-based console support? (We recommend migrating to zf-console, symfony/console, or Aura.CLI) y/N

Deseja instalar suporte ao console baseado em MVC?

Selecione “n” para não instalar.

Would you like to install i18n support? y/N

Deseja instalar suporte a i18n?

Selecione “n” para não instalar.

Would you like to install the official MVC plugins, including PRG support, identity, and flash messages? y/N

Deseja instalar os plug-ins MVC oficiais, incluindo suporte a PRG, identidade e mensagens instantâneas?

Selecione “n” para não instalar.

Would you like to use the PSR-7 middleware dispatcher? y/N

Deseja usar o dispatcher PSR-7?

Selecione “n” para não usar.

Would you like to install sessions support? y/N

Deseja instalar suporte a sessões?

Selecione “n” para não instalar.

Would you like to install MVC testing support? y/N

Deseja instalar suporte de teste MVC?

Selecione “n” para não instalar.

```
Would you like to install the zend-di integration  
for zend-servicemanager? y/N
```

Deseja instalar a integração zend-di para zend-servicemanager?

Selecione “n” para não instalar.

Assim que encerrada a fase de inventário apresentada, a ferramenta solicita a seleção da atualização da configuração da aplicação.

```
Please select which config file you wish to inject  
'ZendDeveloperTools' into:
```

- [0] Do not inject
- [1] config/modules.config.php
- [2] config/development.config.php.dist

```
Make your selection (default is 0):
```

Selecione qual arquivo de configuração você deseja injetar para a ferramenta do desenvolvedor.

Selecione “1” para o módulo de configuração.

```
Remember this option for other packages of the  
same type? (y/N)
```

Relemburar estas opções para outros pacotes do mesmo tipo?

Selecione “n” para não relembrar.

```
Please select which config file you wish to inject  
'Zend/Db' into:
```

- [0] Do not inject
- [1] config/modules.config.php
- [2] config/development.config.php.dist

```
Make your selection (default is 0):
```

Selecione qual arquivo de configuração você deseja injetar para o banco de dados?

Selecione “1” para o módulo de configuração.

```
Remember this option for other packages of the  
same type? (y/N)
```

Relemburar estas opções para outros pacotes do mesmo tipo?

Selecione “n” para não relembrar.

Please select which config file you wish to inject
 'Zend/Form' into:

- [0] Do not inject
- [1] config/modules.config.php
- [2] config/development.config.php.dist

Make your selection (default is 0):

Selecione qual arquivo de configuração você deseja injetar para os formulários:

Selecione “1” para o módulo de configuração.

Remember this option for other packages of the same type? (y/N)

Relembrar estas opções para outros pacotes do mesmo tipo?

Selecione “n” para não relembrar.

Para testar se os comandos do projeto foram executados corretamente, abra uma guia no seu navegador e digite o endereço a seguir. Deve aparecer a mensagem de boas-vindas ao framework Zend na versão 3, conforme ilustrado pela Figura 6.

<http://localhost/ua15/public/>

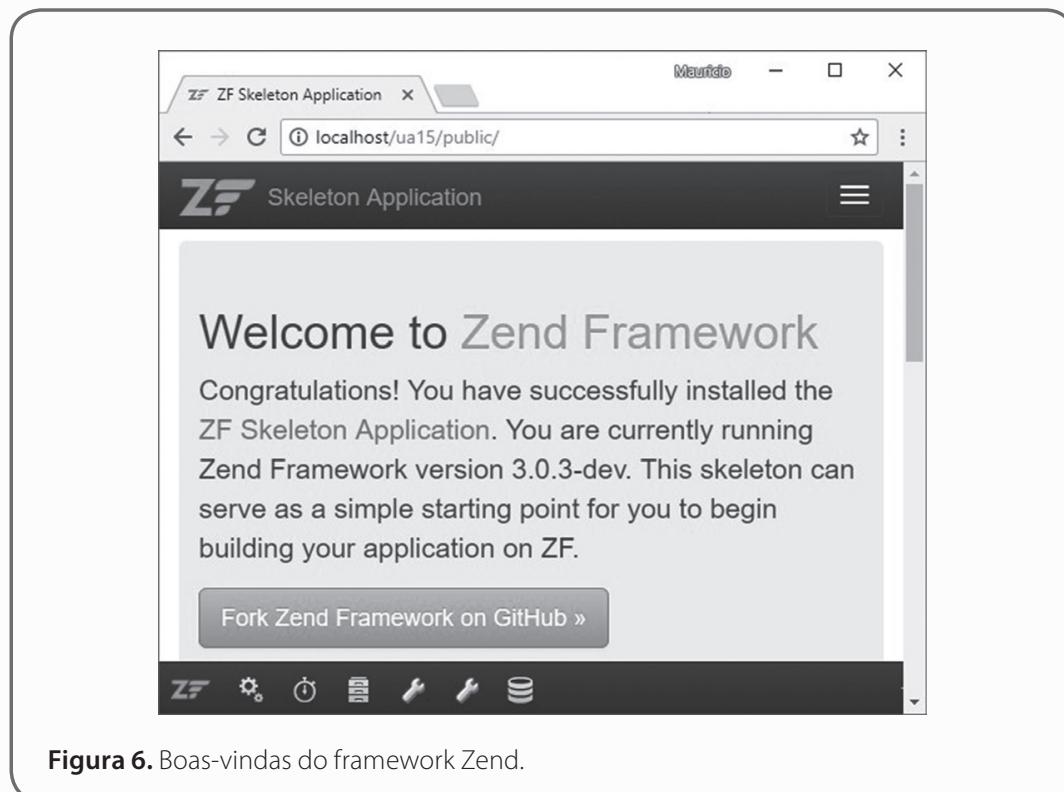


Figura 6. Boas-vindas do framework Zend.

Exemplos de aplicações do framework Zend

O framework Zend pode ser usado para desenvolver aplicações dos mais diversos tipos. Baseado em padrões de software, com foco na reutilização de componentes, permite configurar aplicações seguras e robustas, fundamentadas nas melhores práticas do desenvolvimento de sistemas PHP (LISBOA, 2009).

Aplicações desenvolvidas com frameworks normalmente são compostas por inúmeros arquivos e requerem diversas configurações para o seu correto funcionamento. De modo geral, isso impossibilita apresentar todos os detalhes de uma aplicação nesse trabalho.

Nesse sentido, para ilustrar exemplos de aplicações com o framework Zend 3, baixe dois exemplares disponíveis no GitHub: um que apresenta um modelo básico de aplicação e utiliza o padrão MVC; e outro que acessa um banco de dados MySQL.



Link

Você pode baixar os exemplos do GitHub usando o seguinte link:

<https://goo.gl/iuJuXP>



Após baixar e descompactar o arquivo que contém os exemplos (`using-zf3-book-samples-master.zip`), você deve criar uma pasta chamada “book”, dentro da pasta “`htdocs`” do PHP e transferir para lá duas pastas do arquivo que foi baixado: “`blog`” e “`helloworld`”, conforme apresentado na Figura 7.

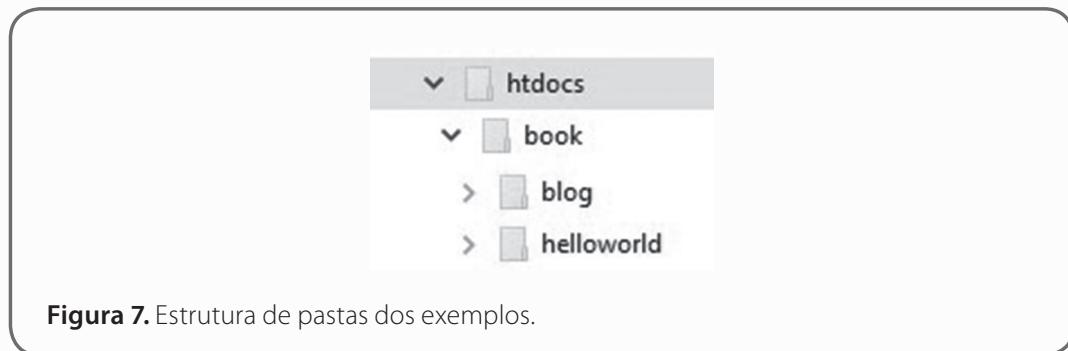
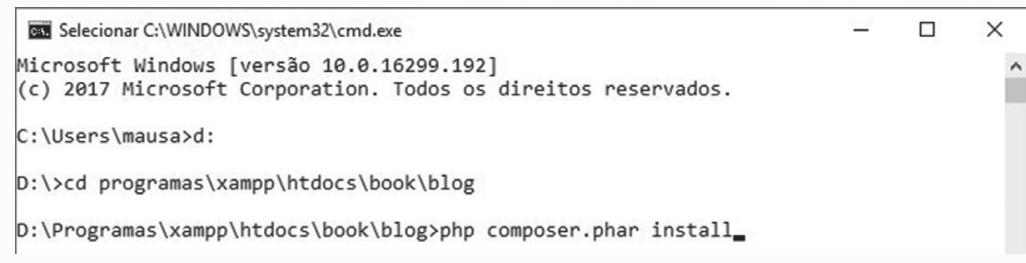


Figura 7. Estrutura de pastas dos exemplos.

Na sequência, você deve realizar a instalação de cada uma das aplicações pela ferramenta Composer, de acordo com a pasta de instalação do XAMPP, conforme segue e é demonstrado na Figura 8.

```
D:\Programas\xampp\htdocs\book\blog>php composer.phar install  
D:\Programas\xampp\htdocs\book\helloworld>php composer.phar install
```



The screenshot shows a Windows Command Prompt window titled "Selecionar C:\WINDOWS\system32\cmd.exe". The window displays the following text:

```
Microsoft Windows [versão 10.0.16299.192]  
(c) 2017 Microsoft Corporation. Todos os direitos reservados.  
C:\Users\mausa>d:  
D:>cd programas\xampp\htdocs\book\blog  
D:\Programas\xampp\htdocs\book\blog>php composer.phar install
```

Figura 8. Execução dos comandos de instalação.

Exemplo 1 — HelloWorld

Para executar a aplicação HelloWorld, acessamos o seguinte endereço no navegador web:

<http://localhost/book/helloworld/public/>

Observe, na Figura 9, a tela da aplicação.

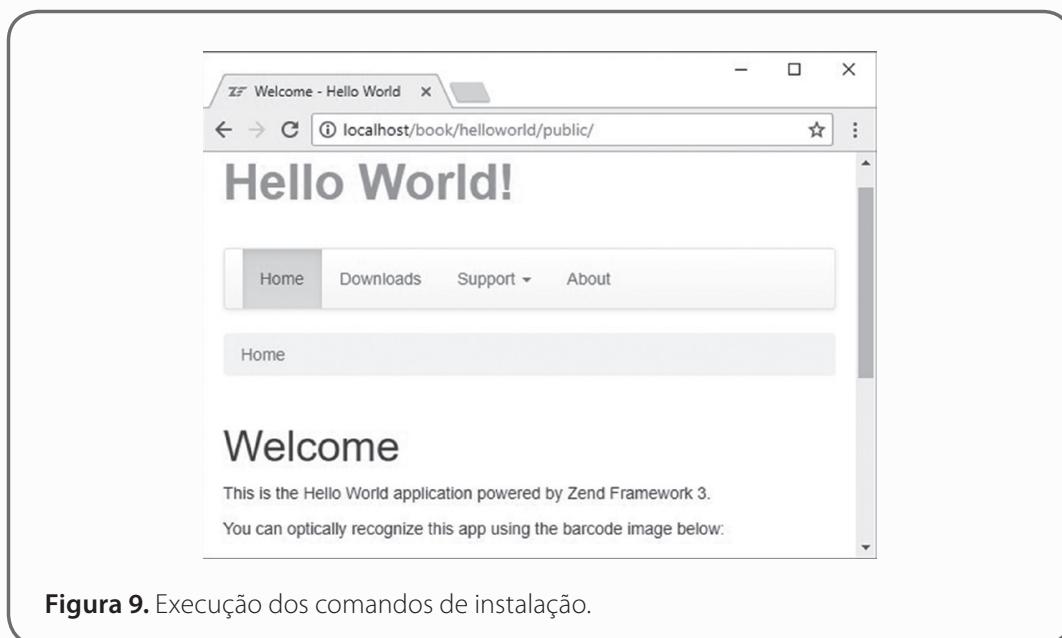


Figura 9. Execução dos comandos de instalação.

O objetivo desse exemplo é apresentar um caso básico da arquitetura MVC. Os links do menu do aplicativo Home, Downloads, Support e About realizam a chamada do **controller** da aplicação, passando na URL o que deve ser tratado. A controller identifica e repassa à **model** que, por sua vez, processa e devolve o resultado. Então a controller encaminha à **view** o que deve ser apresentado. Um resumo da estrutura de pastas da aplicação HelloWorld é apresentado na Figura 10.

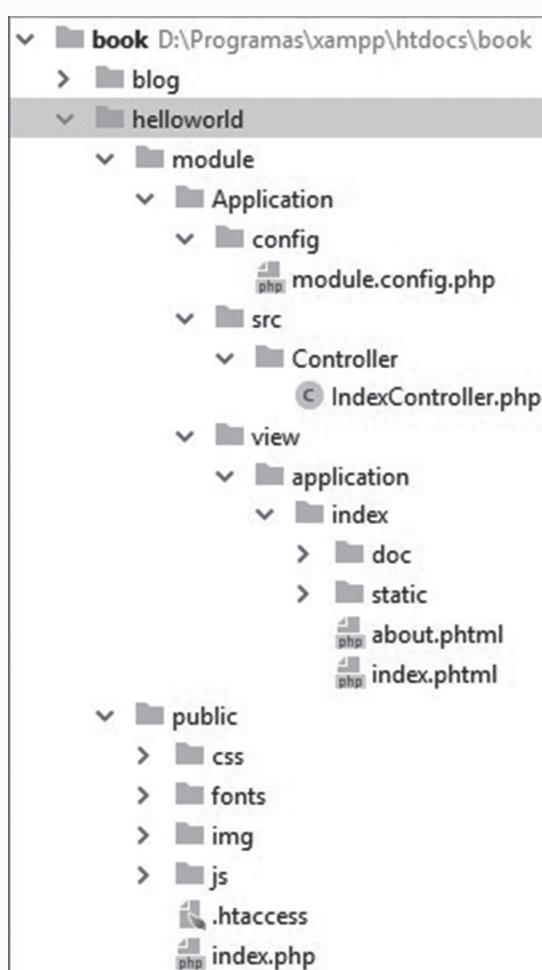


Figura 10. Resumo da estrutura de pastas da aplicação.

Arquivo module.config.php

Esse arquivo trata cada uma das requisições que são passadas na URL. O trecho de código a seguir apresenta uma rota para “/about” que a controller recebe e repassa a ação à model.

```

39     'about' => [
40         'type' => Literal::class,
41         'options' => [
42             'route' => '/about',
43             'defaults' => [
44                 'controller' => Controller\IndexController::class,
45                 'action' => 'about',
46             ],
47         ],
48     ],

```

Arquivo indexController.php

Esse arquivo contém as funções que são executadas pela controller que acessam à model. A função a seguir define os dados que são repassados à página “**about.phtml**”.

```

29     public function about Action()
30     {
31         $panama = 'Hello World';
32         $appDescription = 'Hello World sample application for
Using Zend Framework 3 book';
33
34         // Return variables to view script with the help of
35         // ViewObject variable container
36         return new ViewModel([
37             'panama' => $panama,
38             'appDescription' => $appDescription
39         ]);
40     }

```

Arquivo about.phtml

Esse arquivo pertence à camada view e apresenta os dados processados pela model e disponibilizados pela controller. No framework Zend, os arquivos de visualização possuem a extensão **phtml**.

```

1 <?php
2 $this->headTitle('About');
3 $this->mainMenu()->setActiveItemId('about');
4 $this->pageBreadcrumbs()->setItems([
5     'Home'=>$this->url('home'),
6     'About'=>$this->url('about'),
7 ]);
8 $this->layout()->setTemplate('layout/layout2');
9
10 ?>
11 <h1>About</h1>

```

```
12 <p>
13     Application name: <?= $this->escapeHtml($panama) ?>
14 </p>
15 <p>
16     Application description:
17         <?= $this->escapeHtml($appDescription) ?>
18 </p>
```

Exemplo 2 — Blog

Esse exemplo também utiliza a arquitetura MVC, no entanto, se diferencia do exemplo HelloWorld por realizar conexão com banco de dados MySQL. Nesse sentido, vamos focar nos detalhes de acesso ao banco, uma vez que o restante da estrutura se assemelha ao apresentado no exemplo 1.

Antes de abrir a aplicação, é preciso realizar algumas configurações de conexão com o banco de dados MySQL, uma vez que esse exemplo apresenta na tela informações que estão armazenadas em algumas tabelas.

Conexão com banco de dados

A conexão com o banco de dados MySQL está configurada no arquivo “local.php”. Esse arquivo define a utilização do driver PHP Data Objects (PDO) e configura os parâmetros de acesso, como servidor, usuário, senha e o nome do banco de dados, conforme segue.

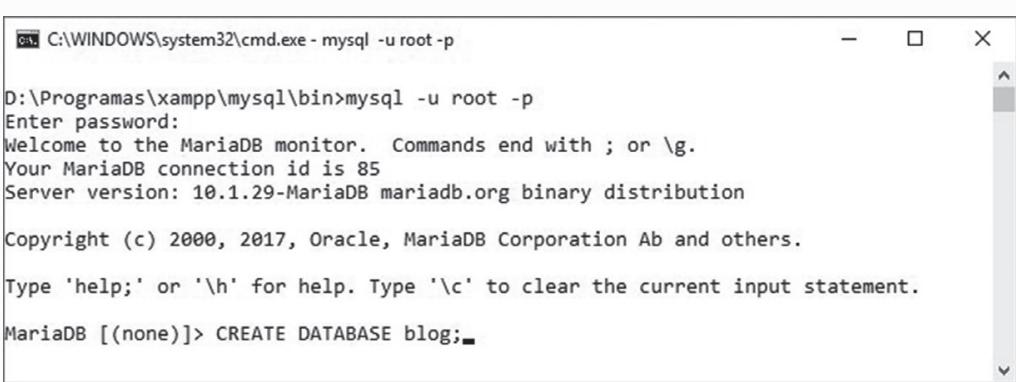
blog/config/autoload/local.php

```
14 use Doctrine\DBAL\Driver\PDOMySql\Driver as PDOMySqlDriver;
15
16 return [
17     'doctrine' => [
18         'connection' => [
19             'orm_default' => [
20                 'driverClass' => PDOMySqlDriver::class,
21                 'params' => [
22                     'host'      => 'localhost',
23                     'user'      => 'root',
24                     'password'  => '',
25                     'dbname'   => 'blog',
26                 ],
27             ],
28         ],
29     ],
30 ];
```

Scripts de criação do banco de dados, das tabelas e inclusão de registros

Via Prompt de comando, você pode executar os seguintes comandos para acessar o MySQL e criar as estruturas necessárias para o funcionamento da aplicação (veja as ilustrações nas Figuras 11 e 12).

```
1 mysql -u root -p
2 create database blog;
```



```
C:\WINDOWS\system32\cmd.exe - mysql -u root -p
D:\Programas\xampp\mysql\bin>mysql -u root -p
Enter password:
Welcome to the MariaDB monitor. Commands end with ; or \g.
Your MariaDB connection id is 85
Server version: 10.1.29-MariaDB mariadb.org binary distribution

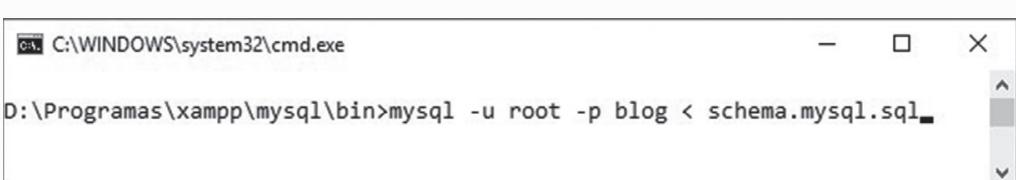
Copyright (c) 2000, 2017, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MariaDB [(none)]> CREATE DATABASE blog;
```

Figura 11. Criação do banco de dados.

```
1 mysql -u root -p blog < schema.mysql.sql
```



```
C:\WINDOWS\system32\cmd.exe
D:\Programas\xampp\mysql\bin>mysql -u root -p blog < schema.mysql.sql
```

Figura 12. Criação das tabelas e inclusão de registros.

Uma vez que a estrutura do aplicativo já está configurada, podemos rodar o Blog acessando o seguinte endereço no navegador web:

<http://localhost/book/blog/public/>

A Figura 13 apresenta a tela de abertura do blog.

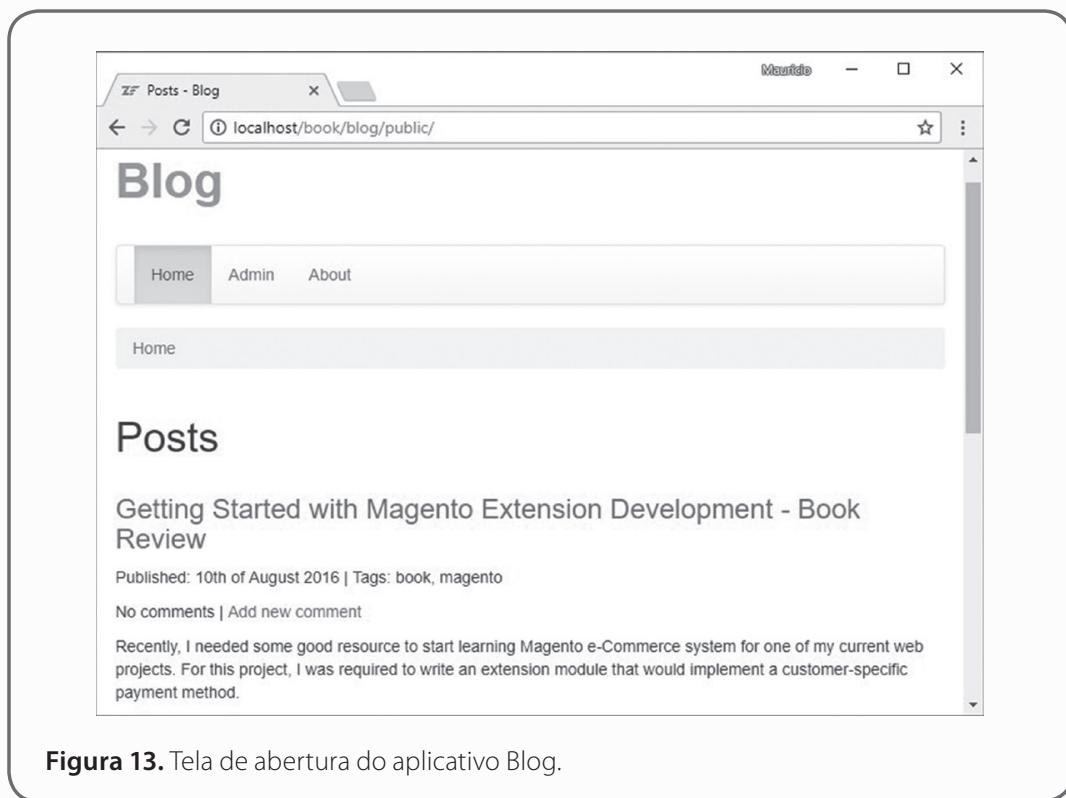


Figura 13. Tela de abertura do aplicativo Blog.

Mapeamento dos atributos

Cada tabela do banco de dados é mapeada em uma classe que pertence ao módulo **blog/modulo/Application/src/Entity**. Essas classes descrevem os atributos de cada tabela, bem como os métodos de acesso **getters** e **setters**.

Os atributos dos formulários para inclusão e alteração de registros estão definidos nos arquivos “CommentForm.php” e “PostForm.php”. Esses arquivos estão localizados na pasta **blog/modulo/Application/src/Form** e apresentam a especificação de cada elemento HTML, como tipo, nome, identificador, etiqueta e outros.

Os formulários que apresentam os dados estão localizados na pasta **blog/module/Application/view/application/post**. Eles renderizam os campos na tela para visualização dos dados.



Referências

BIERER, D.; HUSSAIN, A.; JONES, P. *PHP 7: real world application development*. Birmingham: Packt, 2016.

LISBOA, F. G. S. *Zend framework: componentes poderosos para PHP*. São Paulo: Novatec, 2009.

ZEND FRAMEWORK. *About: overview*. [S.I.], c2006-2018a. Disponível em: <<https://framework.zend.com/about>>. Acesso em: 25 jan. 2018.

ZEND FRAMEWORK. *Documentation: Zend Framework 3*. [S.I.], c2006-2018b. Disponível em: <<https://framework.zend.com/learn>>. Acesso em: 25 jan. 2018.

ZF TUTORIALS. *Getting started: a skeleton application*. [S.I.], 2006-2017. Disponível em: <<https://docs.zendframework.com/tutorials/getting-started/skeleton-application/>>. Acesso em: 25 jan. 2018.



PREZADO ESTUDANTE

**ENCERRA AQUI O TRECHO DO LIVRO DISPONIBILIZADO
PELA SAGAH PARA ESTA PARTE DA UNIDADE.**



CONTRIBUA COM A QUALIDADE DO SEU CURSO

Se você encontrar algum problema nesse material, entre em contato pelo email eadproducao@unilasalle.edu.br. Descreva o que você encontrou e indique a página.

Lembre-se: a boa educação se faz com a contribuição de todos!



Av. Victor Barreto, 2288
Canoas - RS
CEP: 92010-000 | 0800 541 8500
ead@unilasalle.edu.br