

# DESENVOLVIMENTO DE SISTEMAS COM PHP

Maurício de Oliveira Saraiva



SOLUÇÕES  
EDUCACIONAIS  
INTEGRADAS



# Linguagem PHP com formulários

## Objetivos de aprendizagem

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

- Desenvolver dados de formulário com PHP.
- Usar os métodos GET e POST com formulários com PHP.
- Aplicar tratamento de dados em formulários com PHP.

## Introdução

Formulários web são utilizados em páginas dinâmicas para enviar informações que serão recebidas e processadas por um servidor de aplicação web. Por meio desses formulários, os usuários podem interagir com sistemas web com o objetivo de realizar diversas tarefas, como acessar *internet banking*, efetuar compras on-line ou simplesmente pesquisar informações em sites de busca na internet. Os formulários web podem conter diversos elementos que permitem a edição de dados, como campos para edição de texto, caixas de seleção de opções, listas de seleção de itens, botões de alternativas, entre outros.

Neste capítulo, você irá estudar a integração de formulários com PHP, o uso dos métodos GET e POST e o tratamento de dados em formulários com PHP.

## Desenvolvimento de dados de formulário com PHP

Uma aplicação web é um sistema que coleta e transmite informações por meio de uma conexão entre o navegador do usuário e um servidor de aplicação web. Essa conexão ocorre por meio de requisições e respostas, em que o usuário realiza uma requisição e o servidor responde com a informação solicitada (MILANI, 2016).

No início da internet, as páginas web eram estáticas, isto é, seu conteúdo era fixo e entregue de igual forma a todos os usuários. Com o avanço da tecnologia, surgiram as páginas dinâmicas, que são processadas por um servidor de aplicação de acordo com a solicitação de cada requisição, permitindo que os usuários recebam dados personalizados.

As páginas dinâmicas são suportadas por servidores de aplicação específicos, que atendem a determinadas linguagens de programação, como PHP, Java, Perl, C#, entre outros. Esses servidores são responsáveis por receber as solicitações dos usuários, processar e devolver o conteúdo específico (MILLETTO; BERTAGNOLLI, 2014).

A requisição dos usuários em um sistema web é realizada por meio de formulários linguagem de marcação de hipertexto, do inglês HyperText Markup Language (HTML). Os formulários, por sua vez, são compostos por campos ou atributos de diversos tipos, como edição de texto, listas, caixas de seleção, botões, entre outros, que são apresentados pelo navegador web (SOARES, 2013).

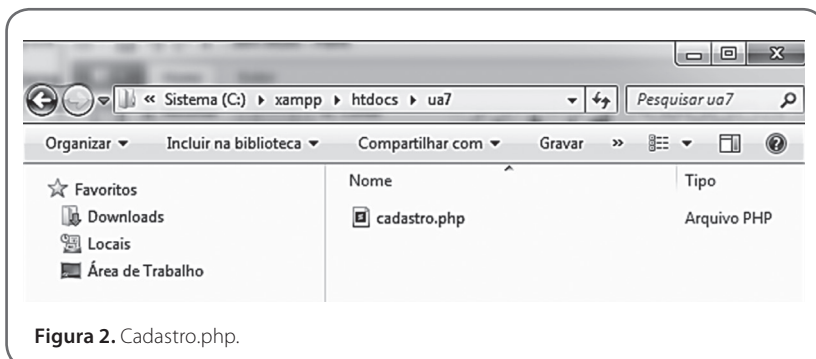
É partir dos formulários que os usuários interagem com um sistema web, e essa interação se dá de diversas formas, por exemplo, para realizar a autenticação por meio de usuário e senha em um webmail, para consultar o extrato da conta corrente no *internet banking*, em uma página para pesquisar detalhes de produtos de um site e-commerce, entre outros.

Formulários web estão por toda a parte na internet. É comum vermos páginas que apresentam formulários para padronizar a entrada e a saída de dados, conforme o objetivo do site. O acesso a um *internet banking*, como você pode ver no exemplo da Figura 1, solicita dados específicos como a agência, o número da conta e a senha do cliente, e exibe os dados de um extrato de forma linear, contendo lançamentos, como data, descrição e valor.



Como dito anteriormente, nas páginas web, os formulários são implementados em linguagem HTML. Essa linguagem permite a criação de elementos de diversos tipos, que atendem as mais variadas finalidades de formulários, desde simples cadastros até a apresentação de layouts de sistemas complexos.

No entanto, para trabalhar com páginas dinâmicas no PHP, você precisa implementar a integração dessas páginas HTML com scripts na linguagem PHP. Para isso, cria-se um arquivo chamado `cadastro.php` e o coloca em uma subpasta dentro da pasta `htdocs`, no local de instalação do servidor de aplicação PHP – XAMPP, conforme representado na Figura 2.



Apesar de ser nomeado com a extensão php, inicialmente o arquivo de cadastro terá apenas instruções HTML, que irão montar os dados do formulário de cadastro para o envio ao servidor. A parte do programa que trabalha com PHP para receber os dados do formulário será tratada na próxima seção deste capítulo. O código-fonte do formulário HTML está apresentado a seguir:

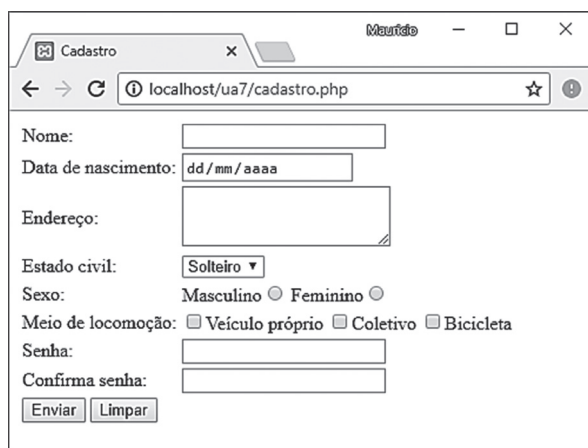
```

1  <!DOCTYPE html>
2  <html>
3  <head>
4  <title>Cadastro</title>
5  <meta charset="utf-8">
6  </head>
7
8  <body>
9      <form name="cadastro" method="get" action="cadastro.php">
10         <table>
11             <tr>
12                 <td><label>Nome:</label></td>
13                 <td><input type="text" name="nome" size="20"></td>
14             </tr>
15
16             <tr>
17                 <td>Data de nascimento:</td>
18                 <td><input type="date" name="dataNascimento"></td>
19             </tr>
20
21             <tr>
22                 <td><label>Endereço:</label></td>
23                 <td><textarea name="endereco" rows="3"
24 cols="22"></textarea></td>
25             </tr>
26
27             <tr>
28                 <td><label>Estado civil:</label></td>
29                 <td>
30                     <select name="estadoCivil">
31                         <option value="S">Solteiro</option>
32                         <option value="C">Casado</option>
33                     </select>
34                 </td>
35             </tr>
36
37             <tr>
38                 <td><label>Sexo:</label></td>
39                 <td>
40                     Masculino<input type="radio" name="sexo" value="M">
41                     Feminino<input type="radio" name="sexo" value="F">
42                 </td>
43             </tr>

```

```
44 <tr>
45 <td><label>Meio de locomoção:</label></td>
46 <td>
47 <input type="checkbox" name="locomocao[]" value="M">Moto
48 <input type="checkbox" name="locomocao[]" value="C">Coletivo
49 <input type="checkbox" name="locomocao[]" value="B">Bicicleta
50 </td>
51 </tr>
52
53 <tr>
54 <td><label>Senha:</label></td>
55 <td><input type="password" name="senha" size="20"></td>
56 </tr>
57
58 <tr>
59 <td><label>Confirma senha:</label></td>
60 <td><input type="password" name="confirmaSenha" size="20"></td>
61 </tr>
62
63 <tr>
64 <td colspan="2">
65 <input type="submit" name="enviar" value="Enviar">
66 <input type="reset" name="limpar" value="Limpar">
67 </td>
68 </tr>
69
70 </table>
71 </body>
72 </html>
```

O resultado do formulário de cadastro está apresentado na Figura 3.



Cadastro

localhost/ua7/cadastro.php

Nome:

Data de nascimento:

Endereço:

Estado civil:

Sexo: ☒ Masculino ☐ Feminino

Meio de locomoção: ☐ Veículo próprio ☐ Coletivo ☐ Bicicleta

Senha:

Confirma senha:

Figura 3. Exibição da tela de cadastro.php.



## Link

Conheça melhor os elementos HTML no link a seguir (RIBEIRO, c2018).



<https://goo.gl/5YHfbs>

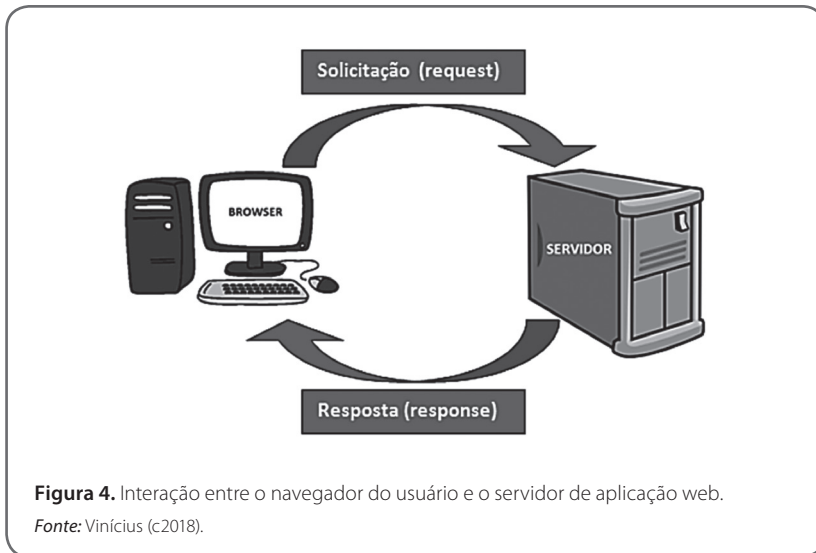
Além dos elementos que representam os campos do formulário, as seguintes instruções definem as características e o comportamento do formulário (MILETTO; BERTAGNOLLI, 2014).

- **Form:** define a abertura de um formulário HTML. Todos os elementos que se encontrarem entre as tags `<form>` e `</form>` compreendem o conjunto de itens do formulário.
- **Name:** especifica o nome de um formulário da página web. É possível que uma página web contenha mais de um formulário, cada qual com seus respectivos campos. Por isso, esse atributo identifica cada um dos possíveis formulários existentes.
- **Method:** define o método protocolo de transferência de hipertexto, do inglês hypertext transfer protocol (HTTP) que o navegador vai usar para enviar os dados do formulário. As duas opções de envio são: GET e POST;
- **Action:** especifica a URL de destino que irá processar as informações enviadas pelo formulário. Se esse atributo não for definido, o navegador enviará os dados para a própria página HTML.
- **Submit:** caracterizado por um botão, é a instrução que realiza a submissão dos dados do formulário.

## Utilização dos métodos GET e POST com formulários com PHP

Uma vez que o formulário para o envio dos dados esteja pronto, conforme apresentado na seção anterior, é possível realizar a submissão dos dados para o servidor de aplicação web. O servidor, por sua vez, recebe os dados do for-

mulário, processa e envia uma resposta de volta ao navegador do usuário por meio de operações conhecidas como request/response (BIERER; HUSSAIN; JONES, 2016). A Figura 4 ilustra essas operações.



Você viu, na seção anterior, que existem duas formas de enviar os dados de um formulário web para um servidor de aplicação web, por meio da propriedade método da instrução `<form>`.

- **get:** neste método, os dados são anexados na URL da página de resposta. Portanto, não é um método seguro, porque os dados são passados abertos, podendo ser visualizados ou modificados pelo usuário na barra de endereços do navegador, por fora do formulário.
- **post:** neste método, os dados são enviados pelo corpo do formulário de modo transparente ao usuário da página web que realiza a submissão. Essa é a forma mais segura de enviar os dados, já que o usuário não consegue visualizá-los nem modificá-los por fora do formulário.

Para tratar o recebimento desses dados no servidor de aplicação web, o PHP fornece dois métodos específicos, conhecidos por `$_GET` e `$_POST`, que atuam respectivamente sobre os atributos `get` e `post` da instrução `<form>` do formulário HTML (BIERER; HUSSAIN; JONES, 2016).



Tanto o método `$_GET` como o `$_POST`, do PHP, criam arrays que contêm todos os elementos e seus respectivos valores enviados pelo formulário. Com isso, é possível montar uma estrutura de repetição `foreach` para ler cada um desses elementos.

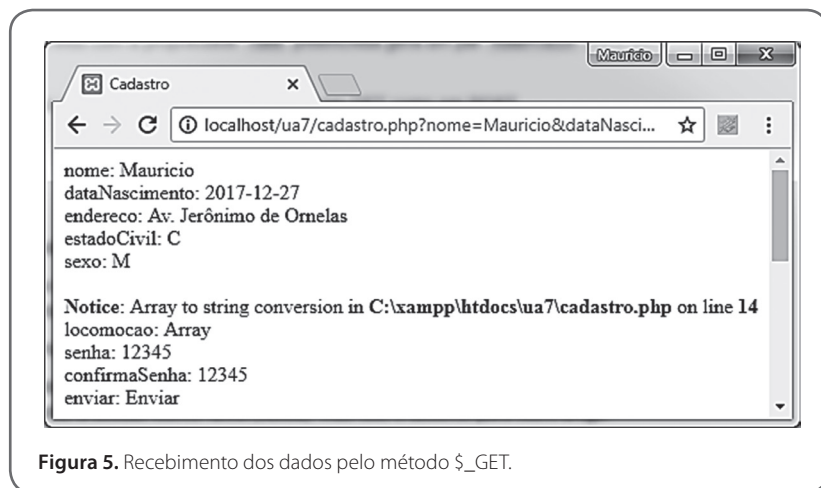
Para exemplificar o recebimento dos dados pelo método `$_GET` do PHP, você deve conferir se o conteúdo da propriedade `method` da instrução `<form>`, do formulário que foi implementado, está conforme segue:

```
1 <form name="cadastro" method="get" action="cadastro.php">
```

Após, acrescente as seguintes linhas no arquivo `cadastro.php`, entre as instruções `<body>` e `<form...>`

```
1 <?php
2     foreach ($_GET as $key => $value) {
3         echo "$key: $value <br>";
4     }
5 >?>
```

Execute a página no navegador, preencha os dados e clique em enviar. O resultado é apresentado na Figura 5.



**Figura 5.** Recebimento dos dados pelo método `$_GET`.

Note que as variáveis e seus respectivos valores são passados na barra de endereço do navegador. Nessa barra, a primeira variável é precedida do

caractere “?”, e as próximas são precedidas pelo caractere “&”, que indica o início de cada elemento da página, incluindo o botão “Enviar”.

```
http://localhost/ua7/cadastro.  
php?nome=Mauricio&dataNascimento=2017-12-27&ende  
reco=Av.+Jer%C3%B4nimo+Ornelas&estadoCivil=C&se  
xo=M&locomocao%5B%5D=M&locomocao%5B%5D=C  
&senha=12345&confirmaSenha=12345&enviar=Enviar
```



### Fique atento

Tenha cuidado ao utilizar o método *get* em formulários web, pois os dados ficam abertos na barra de endereço do navegador e podem ser alterados diretamente pelo usuário, sem passar novamente pelo formulário. Como boa prática, utilize o método *get* apenas quando os dados não forem sigilosos.

Na tela de apresentação do resultado (Figura 5), o PHP indicou um alerta sobre a conversão de um array para string no elemento locomocao, sem apresentar os seus valores. Isso ocorreu, porque esse campo é do tipo checkbox, e foi declarado como array no formulário para permitir a seleção de várias opções sem ter que criar variáveis com nomes diferentes.

Para capturar os valores selecionados nesse checkbox, você pode criar outro laço de repetição, conforme segue:

```
1 foreach ($_GET['locomocao'] as $key => $value) {  
2     echo "$key: $value <br>";  
3 }
```

Apesar de mostrar os valores dos elementos por meio de estruturas de repetição, conforme apresentado anteriormente, é mais comum capturar os atributos um por um, a partir de seu nome de identificação. Para capturar um elemento de forma isolada no PHP e atribuir seu valor a uma variável, utilize a seguinte instrução (MILANI, 2016):

```
1 $nome = $_GET['nome'];  
2 echo $nome;
```

Para receber os dados pelo método \$\_POST do PHP, é preciso alterar a propriedade method na instrução <form> para post no formulário cadastro.php.

```
<form name="cadastro" method="post" action="cadastro.php">
```

No PHP, a forma de tratamento via post é semelhante, devendo apenas se utilizar o método \$\_POST em vez do método \$\_GET. Além disso, você pode perceber que as variáveis não são enviadas na barra de endereço do navegador, ocultando, assim, todos os dados enviados pelo usuário (SOARES, 2013).

Não é necessário realizar a submissão do formulário para a mesma página, conforme o exemplo apresentado. É possível submeter para qualquer página, inclusive de outros servidores ou domínios da internet, desde que a página de destino esteja preparada para receber os dados que serão enviados.

Neste mesmo exemplo, o seguinte script PHP faz a verificação e o recebimento dos dados do formulário pelo método \$\_POST.

```
1 <?php
2     if ($_POST['enviar'] == 'Enviar') {
3         foreach ($_POST as $key => $value) {
4             if ($key <> 'locomocao')
5                 echo "$key: $value <br>";
6         }
7
8         foreach ($_POST['locomocao'] as $key => $value) {
9             echo "Locomoção: $value <br>";
10        }
11    }
12 ?>
```

O resultado é apresentado na Figura 6.

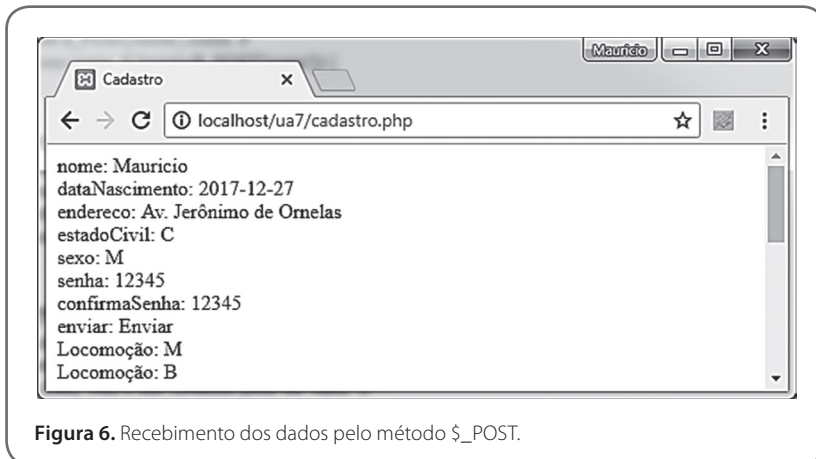


Figura 6. Recebimento dos dados pelo método `$_POST`.

## Aplicar tratamento de dados em formulários com PHP

Uma das atividades mais importantes no desenvolvimento de sistemas web fala do tratamento de dados em formulários. Esse tratamento serve para garantir a confiabilidade da informação que será processada pelo sistema, uma vez que não se deve permitir o armazenamento de dados incorretos, por exemplo, um campo vazio que requer preenchimento obrigatório, letras em um campo que exige apenas números e um e-mail inválido digitado por engano pelo usuário, entre outros (MILETTO; BERTAGNOLLI, 2014).

Existem diversas formas de tratar os dados de um formulário web, entre as quais destacam-se as validações no navegador do cliente por meio das

linguagens HTML5 e Javascript, e por scripts PHP no servidor de aplicação web. Tanto a validação no navegador do usuário como a validação no servidor web são necessárias, e uma completa a outra.

Neste capítulo trataremos especificamente da validação de dados por scripts PHP no servidor de aplicação web.



### Fique atento

Para realizar o estudo de tratamento de dados via script PHP, o formulário utilizado não apresenta nenhuma validação via HTML5, fazendo toda e qualquer informação ser enviada ao servidor da forma como o usuário submeteu. No entanto, as validações via HTML5 devem ser implementadas posteriormente, para aumentar ainda mais a segurança e a confiabilidade dos dados.

Considerando, nesse momento, que você está rodando uma página que recebe dados de um formulário web, inicialmente será necessário verificar se houve a submissão de dados quando ela for carregada. Essa verificação é importante porque uma página pode ser chamada diretamente pelo navegador do usuário, sem ter sido submetida por um formulário.

```
1  if (!isset($_POST) || empty($_POST)) {  
2      echo "Não houve postagem via formulário.";   
3  }
```

O comando `isset` é utilizado para verificar se uma variável foi iniciada e a função `empty` verifica se o array `$_POST` está vazio. Você pode, ainda, ser mais restritivo e verificar a submissão do formulário especificamente pelo botão enviar, evitando, assim, o acesso direto (PHP, c2001-2018c).

```
1  if (isset($_GET['enviar']) <> 'Enviar') {  
2      echo "Não houve postagem pelo botão enviar do formulário.";   
3  }
```

Utilize a instrução `isset` em conjunto com o operador ternário do PHP para definir um valor padrão para determinado campo recebido pelo formulário. Isso é importante para os casos em que o formulário possui campos HTML do tipo `checkbox` e `radio`, que não são enviados quando nenhuma opção é selecionada.

```
1 $locomocao = isset($_GET['locomocao']) ? $_GET['locomocao'] : null;
```

É possível, também, remover tags HTML e tags PHP do conteúdo dos dados enviados pelo formulário pela função `strip_tags` do PHP. Para testar esse comando, preencha no campo nome do formulário a expressão “Maurício <?php”, e perceba que essa instrução irá retirar a parte que se refere a um script PHP.

```
1 $nome = strip_tags($_POST['nome']);  
2 echo $nome; //resultado: Maurício
```

Para verificar todos os campos do formulário automaticamente com a função `strip_tags`, utilize uma estrutura de repetição `foreach` no array de elementos do formulário, conforme exemplo a seguir. Lembre-se que o campo `locomocao` está definido como um array, então dele deve ser tratado separadamente em outra estrutura de repetição.

```
1 foreach ($_POST as $chave => $valor) {  
2     $$chave = strip_tags($valor);  
3 }
```

Utilize a função `is_numeric` para verificar se determinado campo contém apenas um valor numérico. Nessa função, a separação de casas decimais é realizada pelo ponto e não por vírgula (PHP, c2001-2018b).

```
1 if (!is_numeric($_POST['senha']))  
2     echo $_POST['senha'] . " não é numérico.";
```

No caso em que se deseja extrair os números de um campo que deveria conter apenas valores inteiros, pode-se usar a função `filter_var` com o parâmetro `FILTER_SANITIZE_NUMBER_INT`. Essa função extrai apenas os números e os sinais positivo e negativo (PHP, c2001-2018a).

```
1 $num = filter_var($_POST['senha'], FILTER_SANITIZE_NUMBER_INT);  
2 echo "$num";
```

Para realizar a validação de um campo e-mail, utilize a função `filter_var` com a parâmetro `FILTER_VALIDATE_EMAIL`.

```
1 if (!filter_var($_POST['email'], FILTER_VALIDATE_EMAIL))  
2     echo "não é um e-mail válido.";
```



### Link

Confira os filtros de validate e sanitize do PHP no seguinte link (JAQUES, 2015).

<https://goo.gl/d8azKm>





## Referências

BANCO DO BRASIL. *Autoatendimento pessoa física*. Brasília, DF, c2018. Disponível em: <<https://www2.bancobrasil.com.br/aapf/login.jsp>>. Acesso em: 21 jan. 2018.

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

JAQUES, R. *Filtrando e validando dados no PHP com filter\_var()*. [S.l.]: PHPit, 2015. Disponível em: <[http://www.phpit.com.br/artigos/filtrando-e-validando-dados-no-php-com-filter\\_var.phpit](http://www.phpit.com.br/artigos/filtrando-e-validando-dados-no-php-com-filter_var.phpit)>. Acesso em: 21 jan. 2018.

MILANI, A. *Construindo aplicações web com PHP e MySQL*. 2. ed. São Paulo: Novatec, 2016.

MILETTO, E. M.; BERTAGNOLLI, S. C. *Desenvolvimento de software II: introdução ao desenvolvimento web com HTML, CSS, JavaScript e PHP*. Porto Alegre: Bookman, 2014. (Tekne).

PHP. *filter\_var*. [S.l.]: The PHP Group, c2001-2018a. Disponível em: <[http://php.net/manual/pt\\_BR/function.filter-var.php](http://php.net/manual/pt_BR/function.filter-var.php)>. Acesso em: 26 dez. 2017.

PHP. *is\_numeric*. [S.l.]: The PHP Group, c2001-2018b. Disponível em: <[http://php.net/manual/pt\\_BR/function.is-numeric.php](http://php.net/manual/pt_BR/function.is-numeric.php)>. Acesso em: 27 dez. 2017.

PHP. *isset*. [S.l.]: The PHP Group, c2001-2018c. Disponível em: <[http://php.net/manual/pt\\_BR/function.isset.php](http://php.net/manual/pt_BR/function.isset.php)>. Acesso em: 27 dez. 2017.

RIBEIRO, D. C. *Introdução ao HTML: elementos de formulários HTML*. [S.l.]: Linha de Código, c2018. Disponível em: <<http://www.linhadecodigo.com.br/artigo/3443/introducao-ao-html-elementos-de-formularios-html.aspx>>. Acesso em: 21 jan. 2018.

SOARES, W. *PHP 5: conceitos, programação e integração com banco de dados*. 7. ed. São Paulo: Érica, 2013.

VINÍCIUS, T. *Como funcionam as aplicações web*. Rio de Janeiro: DevMedia, c2018. Disponível em: <<https://www.devmedia.com.br/como-funcionam-as-aplicacoes-web/25888>>. Acesso em: 21 jan. 2018.

XAVIER, D. W. *Obter dados de formulário*. [S.l.]: Expert.net, c2007-2010. Disponível em: <<http://www.tiexpert.net/programacao/web/php/get-post-request.php>>. Acesso em: 21 jan. 2018.



Encerra aqui o trecho do livro disponibilizado para esta Unidade de Aprendizagem. Na Biblioteca Virtual da Instituição, você encontra a obra na íntegra.

Conteúdo:



SOLUÇÕES  
EDUCACIONAIS  
INTEGRADAS