

## Faça o que eu fiz na aula

Nosso programa está quase terminando, porém, olhe na tela de conclusão do cadastro, ela está muito genérica, não está? iria ser bacana se pudéssemos deixar ela mais amigável para o usuário.

Poderíamos exibir o sobrenome da pessoa, precedido por Sr ou Sra dependendo do gênero.

O gênero vem de um elemento `select`, com as opções escritas dentro da tag `option`, seu conteúdo é uma string M ou F.

Podemos pegar este caractere a partir da variável global `$_POST`, com a chave “genero” que é o nome do elemento `select`.

Agora com essa variável, vamos adicioná-la no objeto `Usuario` que estamos instanciando no arquivo `cadastro.php`:

```
$usuario = new Usuario($_POST['nome'], $_POST['senha'], $_POST['genero']);
```

E vamos criar este argumento na classe para conseguir receber o dado que enviamos.

```
public function __construct(string $nome, string $senha, string $genero)
```

Para exibirmos o tratamento, vamos precisar do nome e do gênero, então é isso que vamos utilizar na assinatura do método, e vamos chamar o método com o nome que descreve o que ele faz:

```
private function adicionaTratamentoAoNome(string $nome, string $genero): void
{

}
```

Legal, agora dentro desse método, temos que fazer uma ação, pegar o sobrenome e chamar de Sr se o gênero for M ou Sra se for F. Isso chama por uma condicional, vamos lá:

```
if ($genero === 'M') {

}
```

Agora para fazermos a alteração, vamos utilizar uma expressão regular para encontrar a primeira palavra, e substituir tudo isso que encontrarmos por Sr.

O caractere `\w` vai pegar todas as letras maiúsculas e minúsculas da nossa string, e usamos o `+` para pegar uma ou mais vezes.

O `\b` é o limite de palavra, assim como o `$` é o limite da linha.

```
$this->tratamento = preg_replace('/^(\\w+)\\b/', 'Sr.', $nome, 1);
```

Beleza! agora podemos fazer a mesma coisa para o gênero feminino, vamos adicionar mais uma condicional:

```
if ($genero === 'F') {  
    $this->tratamento = preg_replace('/^(\\w+)\\b', 'Sra.', $nome, 1);  
}
```

Vamos extrair essa atribuição para um método:

```
public function setTratamento($tratamento): void  
{  
    $this->tratamento = $tratamento;  
}
```

E vamos fazer o getter desse método, para podermos exibir ele na página:

```
public function getTratamento(): string  
{  
    $this->tratamento = $tratamento;  
}
```

E para conseguirmos visualizar, vamos chamar este método no HTML

```
<p><?php echo $usuario->getTratamento(); ?>, seguem os dados de sua conta:</p>
```

Nosso formulário está funcionando! mas ainda tem um ponto muito importante no quesito de segurança a ser observado.

Se digitarmos no campo input do HTML, ao invés do nosso nome, um código como este:

```
<script>alert("Código executado");</script>
```

E clicarmos em cadastrar, será exibido um pop-up na próxima tela, o que significa que o código foi executado, e isso é uma falha de segurança no nosso sistema, pois o usuário pode executar scripts na nossa página e alterar qualquer dado dentro dela como um texto, imagens ou até redirecionar o nosso site para um site mal-intencionado, isso é muito perigoso.

Se isso não aconteceu no seu navegador, o que acontece é que algumas versões recentes do Chrome, já possuem uma proteção melhor contra ataques simples assim, mas se você fizer algo mais sofisticado ou usar o Firefox ou qualquer outro navegador, você conseguirá ver o erro.

O que acontece é que alguns caracteres como o sinal de menor < e o de maior > tem sentido especial no HTML. Para corrigirmos isso, temos que utilizar a função htmlspecialchars do PHP para transformar os caracteres em entidades HTML.

Adicionamos eles depois de toda execução da função echo, ficando desta maneira:

```
<h1>Cadastro feito com sucesso.</h1>  
<?php echo htmlspecialchars($usuario->getTratamento()); ?>, seguem os dados de sua conta:  
<ul class="list-group">  
    <li class="list-group-item">Primeiro nome: <?php echo htmlspecialchars($usuario->getNome()); ?>
```

```
<li class="list-group-item">Sobrenome: <?php echo htmlspecialchars($usuario->getSobrenome()); ?>
<li class="list-group-item">Usuário: <?php echo htmlspecialchars($contato->getNomeUsuario()); ?>
<li class="list-group-item">Senha: <?php echo htmlspecialchars($usuario->getSenha()); ?></li class="list-group-item">
<li class="list-group-item">Telefone: <?php echo htmlspecialchars($contato->getTelefone()); ?></li class="list-group-item">
<li class="list-group-item">Email: <?php echo htmlspecialchars($contato->getEmail()); ?></li class="list-group-item">
<li class="list-group-item">Endereço: <?php echo htmlspecialchars($contato->getEnderecoCep()); ?></li class="list-group-item">
</ul>
```

Com isso o nosso programa está protegido contra este tipo de ataque e finalizamos o programa!