

Melhorando o filtro

Transcrição

Para o usuário conseguir realizar a busca com filtro, implementaremos um formulário em `index.php`, entre tags `<div>` de classe `container`. Neste formulário, precisaremos de algo que nos forneça diversas opções, o que geralmente chamamos de "combo box" ou "select".

```
<form>
  <select name="taxonomy">
    <option value="">São Paulo</option>
  </select>
</form>
```

Porém, já sabemos que definir "São Paulo" manualmente no código não é a forma mais interessante de trazer as localizações existentes no banco de dados. O Wordpress possui uma função que serve para esse caso, `get_terms()`, que ficará em uma variável denominada `taxonomias` e para a qual passaremos `localizacao`.

Então precisaremos que o array de taxonomias gerado seja iterado para nos retornar opções (`<option>`) para cada uma das taxonomias iteradas. Imprimiremos `<option>`s cujo valor será de `$taxonomia` passando o `slug`, que é como acessamos um atributo do objeto no PHP, algo que tem a ver com conceitos de Orientação a Objetos.

E para a tela do usuário, em vez de "São Paulo" *hardcoded*, exibiremos `<?php $taxonomia->name; ?>`. Ainda, dentro do formulário, incluiremos um botão para que a informação seja submetida, e aplicaremos um código CSS já pronto nele.

```
<?php $taxonomias = get_terms('localizacao'); ?>
<form class="busca-localizacao-form">
  <div class="taxonomy-select-wrapper">
    <select name="taxonomy">
      <?php foreach($taxonomias as $taxonomia) { ?>
        <option value="<?= $taxonomia->slug; ?>"><?= $taxonomia->name; ?></option>
      <?php } ?>
    </select>
  </div>
  <button type="submit">Filtrar</button>
</form>
```

Atualizaremos o site e teremos a nova funcionalidade com pré-seleção em "Rio de Janeiro". Acessaremos o dashboard na parte de localizações, e perceberemos que a contagem referente a "Vila Mariana" é 0, pois não registramos nenhum imóvel como pertencente a este bairro, e é por isso que não existe esta opção na tela principal, junto às demais.

No entanto, ainda não obteremos o resultado desejado ao utilizarmos a filtragem. Está faltando passarmos estas informações para algum outro lugar, certo? Da maneira em que estão, não passamos nenhum atributo de tag principal, isto é, nem `method`, nem `action`. No primeiro caso, poderemos utilizar `get`, e não `post`, porque não enviamos nada para o banco de dados efetivamente. Para `action`, poderemos passar o endereço atual, ou seja, o `index`.

O Wordpress possui uma forma inteligente de fazer isso por meio do `bloginfo()`:

```

<form class="busca-localizacao-form" action="<?= bloginfo('url'); ?>">
    <div class="taxonomy-select-wrapper">
        <select name="taxonomy">
            <?php foreach($taxonomias as $taxonomia) { ?>
                <option value="<?= $taxonomia->slug; ?>"><?= $taxonomia->name; ?></option>
            <?php } ?>
        </select>
    </div>
    <button type="submit">Filtrar</button>
</form>

```

Agora que a informação está sendo recebida adequadamente, precisaremos enviá-la para `terms` :

```

$taxQuery = array(
    array(
        'taxonomy' => 'localizacao',
        'field' => 'slug',
        'terms' => $_GET['taxonomy']
    )
);

```

Com isso, nosso filtro estará funcionando nos conformes. No entanto, existe uma questão: se acessarmos simplesmente `localhost/wordpress`, nossa página ficará vazia, quando o correto seria aparecerem todos os imóveis cadastrados. A busca por termos é realizada após o usuário submeter o formulário, passando as informações adiante. De que forma detectaremos esta ação?

Teremos que verificar se o array do GET e a sua chave existem, e para isto utilizaremos a função `array_key_exists()`, passando `taxonomy`, que é a nossa chave, o *name* da função, e o array de `GET`. Se existir, daí sim a busca será realizada. E então, vamos isolar isto em uma variável, `queryTaxonomy`.

```

<?php
$queryTaxonomy = array_key_exists('taxonomy', $_GET);
$css_especifico = 'index';
require_once('header.php');
?>

```

Então, para que a busca seja feita somente quando existir esta variável, teremos que incluir um `if` :

```

if($queryTaxonomy) {
    $taxQuery = array(
        array(
            'taxonomy' => 'localizacao',
            'field' => 'slug',
            'terms' => $_GET['taxonomy']
        )
    );
}

```

Atualizaremos a página no navegador, e tudo estará sendo exibido como gostaríamos. Todavia, há outra incongruência: dentre as opções disponíveis para a filtragem, deveria haver algo como "Todos", pois, da maneira em que está o usuário

terá que acessar a página inicial manualmente, ou então clicar no logotipo, localizado no cabeçalho.

Primeiramente, ele precisará aparecer no `<select>`, em que incluiremos uma opção a mais. Só precisamos nos atentar para não fazermos isso dentro do *foreach*, caso contrário ela será reproduzida todas as vezes em que houver alguma taxonomia. Como queremos buscar por tudo o que for possível, o valor enviado para a URL será vazio.

```
<select name="taxonomy">
  <option value="">Todos os imóveis</option>
  <?php foreach($taxonomias as $taxonomia) { ?>
    <option value="<?= $taxonomia->slug; ?>"><?= $ taxonomia->name; ?></option>
  <?php } ?>
</select>
```

No entanto, se deixarmos simplesmente `<option value="">Todos os imóveis</option>`, ao clicarmos em "Filtrar" com a opção "Todos os imóveis", não obteremos nenhum resultado. Isso porque a chave `taxonomy` é encontrada, porém o valor é inexistente. E então, quando o `if` é acessado, `terms` passa a ser "vazio". Qual dos nossos imóveis possui localização vazia? Nenhum, certo?

Para contornar este problema, incluiremos outro `if` para que, se o nosso GET for vazio, devemos redirecionar o usuário à página inicial, removendo todos os argumentos passados, com a função `wp_redirect()`. Além disso, será necessário garantir que `$_GET['taxonomy']` existe, e para isto utilizaremos `$queryTaxonomy && $_GET['taxonomy'] === ''` antes de `$_GET['taxonomy'] === ''`.

```
<?php
$queryTaxonomy = array_key_exists('taxonomy', $_GET);
if( $queryTaxonomy && $_GET['taxonomy'] === '' ) {
    wp_redirect( home_url() );
}
$css_especifico = 'index';
require_once('header.php');
?>
```

Testaremos a implementação na nossa página, e tudo estará funcionando adequadamente.