03

Listando livros usando como filtro a classe

Transcrição

Além de critérios de busca por `BsonDocument`, é possível selecionarmos por classes.

Adicionaremos um novo item, ao qual daremos o nome de "listandoDocumentosFiltroClasse".

> Para criar uma nova classe, clique com o botão direito do mouse sobre o nome do projeto e selecione "Adicionar > Novo Item". Na caixa de diálogo é possível definir o nome do arquivo.

Aproveitaremos o código do exemplo anterior, apagando a primeira busca, onde não havia critérios, e manteremos a busca com o critério do autor Machado de Assis.

O próximo passo será filtrarmos a busca por uma classe.

Copiaremos o seguinte trecho do código:

```
Console.WriteLine("Listando Documentos Autor = Machado de Assis")
```

Em seguida, adicionaremos a indicação de que filtraremos por classe:

```
Console.WriteLine("Fim da Lista");
                Console.WriteLine("");
                Console.WriteLine("");

                Console.WriteLine("Listando Documentos Autor = Machado de Assis - Classe");
```

O próximo passo será criar a variável `construtor`:

```
var construtor = Builders<Livro>.Filter
```

Observe que ela é do tipo `Builders`, especial dentro do MongoDB.

Como parâmetro, teremos o nome da nossa classe e a propriedade `Filter`. Desta forma, construímos um filtro para esta classe.

Depois, vamos adicionar uma segunda variável: `condicao`.

```
var condicao = construtor
```

Após ela, teremos um ".", que ao digitarmos mostrará diversos critérios disponíveis. Dentre eles, selecionaremos `Eq`.

Criaremos então uma condição. Ela deverá possuir uma variável, que podemos denominar de qualquer forma, aqui escolheremos "x" e a associaremos à propriedade "Autor" da classe `Livro`, além da condição "Machado de Assis", da

seguinte forma: `(X => x.Autor, "Machado de Assis")` .

```
Console.WriteLine("Fim da Lista");
                Console.WriteLine("");
                Console.WriteLine("");

                Console.WriteLine("Listando Documentos Autor = Machado de Assis - Classe");
                var construtor = Builders<Livro>.Filter
                var condicao = construtor.Eq(X => x.Autor, "Machado de Assis")
```

Assim temos um comando que utilizará a classe `Livro` para buscar somente os elementos cujo autor é Machado de Assis.

Agora, copiando o comando para efetuar a busca dos documentos dentro da coleção do MongoDB, em vez de usarmos a variável "Filtro" (BsonDocument), utilizaremos a variável `condicao` .

```
Console.WriteLine("Fim da Lista");
                Console.WriteLine("");
                Console.WriteLine("");

                Console.WriteLine("Listando Documentos Autor = Machado de Assis - Classe");
                var construtor = Builders<Livro>.Filter
                var condicao = construtor.Eq(X => x.Autor, "Machado de Assis")

                listaLivros = await conexaoBiblioteca.Livros.Find(condicao).ToListAsync();
foreach (var doc in listaLivros)
{
    Console.WriteLine(doc.ToJson<Livros>());
}
Console.WriteLine("Fim da Lista");
```

No exemplo anterior, foi utilizada uma variável `BSON` dentro de `Find` , e, neste exemplo temos uma estrutura do tipo `Builders` , utilizando uma condição `Eq` , e fazendo referência ao valor dentro da classe, onde a propriedade "Autor" deve ser igual a "Machado de Assis".

Iremos executar o programa, salvando e clicando em "Iniciar".

Surgirá uma caixa de diálogo onde serão exibidas as listas com os livros filtrados, conforme nosso critério.

Partiremos agora para critérios mais complexos.

Copiaremos o seguinte trecho do código anterior:

```
Console.WriteLine("Listando Documentos Autor = Machado de Assis - Classe");
                var construtor = Builders<Livro>.Filter
                var condicao = construtor.Eq(X => x.Autor, "Machado de Assis")

                listaLivros = await conexaoBiblioteca.Livros.Find(condicao).ToListAsync();
foreach (var doc in listaLivros)
{
    Console.WriteLine(doc.ToJson<Livros>());
}
```

```
Console.WriteLine("Fim da Lista")
Console.WriteLine("");
Console.WriteLine("");
```

Como exemplo, listaremos documentos em que a data de publicação seja maior ou igual a 1999.

Removeremos a menção à variável, uma vez que já foi feita acima.

```
Console.WriteLine("Listando Documentos Autor = Machado de Assis - Classe");
                construtor = Builders<Livro>.Filter
                condicao = construtor.Eq(X => x.Autor, "Machado de Assis")

                listaLivros = await conexaoBiblioteca.Livros.Find(condicao).ToListAsync();
foreach (var doc in listaLivros)
{
    Console.WriteLine(doc.ToJson<Livros>());
}
Console.WriteLine("Fim da Lista")
Console.WriteLine("");
Console.WriteLine("");
```

Atualmente, a variável `condicao` está assim:

```
condição = constructor.Eq(x => x.Autor,"Machado de Assis")
```

Trocaremos a condição `Eq` (igual), pela função `Gte` (maior ou igual a).

Em vez de "Autor" teremos o "Ano" e, no lugar de "Machado de Assis", escreveremos "1999".

Portanto:

```
Console.WriteLine("Listando Documentos ano publicacao seja maior ou igual a 1999");
construtor = Builders<Livro>.Filter;
condicao = constructor.Gte(x => x.Ano, 1999);

listaLivros = await conexaoBiblioteca.Livros.Find(condicao).ToListAsync();
foreach (var doc in listaLivros)
{
    Console.WriteLine(doc.ToJson<Livro>());
}
Console.WriteLine("Fim da Lista");
Console.WriteLine("");
Console.WriteLine("");
```

Clicaremos em "Iniciar" para executarmos o programa.

Surgirá uma caixa de diálogo, onde estarão listados todos os documentos com data de lançamento igual ou posterior a "1999".

Criaremos um próximo critério, mais complexo.

Listaremos todos os livros publicados a partir de 1999, e que possuam mais de 300 páginas.

Já temos o critério `condicao = construtor.Gte(x => x.Ano, 1999)` que nos fornece as publicações lançadas de 1999 em diante.

Criaremos ao lado o novo critério.

Para isso utilizaremos o caractere `&` que indica uma adição, e incluiremos outro construtor:

```
constructor.Gte(x => x.Paginas, 300)
```

Assim serão listados os livros com publicação igual ou maior ao ano de 1999, e aqueles que possuem 300 páginas ou mais.

Resultando em:

```
Console.WriteLine("Listando Documentos ano publicacao seja maior ou igual a 1999 e que tenham ma
construtor = Builders<Livro>.Filter;
condicao = constructor.Gte(x => x.Ano, 1999) & constructor.Gte(x => x.Paginas, 300);

listaLivros = await conexaoBiblioteca.Livros.Find(condicao).ToListAsync();
foreach (var doc in listaLivros)
{
    Console.WriteLine(doc.ToJson<Livro>());
}
Console.WriteLine("Fim da Lista");
Console.WriteLine("");
Console.WriteLine("");
```

Iremos salvar e executar.

Surgirá uma caixa de diálogo onde serão exibidos, na parte superior, todos os livros com lançamento igual ou posterior a 1999, e, na parte inferior, os livros com 300 páginas ou mais.

Por fim, criaremos mais um filtro.

Nas características que demos aos livros está o critério "Assunto", que pode ter um ou mais elementos.

Suponhamos que desejamos listar todos os livros de "Ficção Científica".

Copiando o código com o qual acabamos de trabalhar e colando logo abaixo, realizaremos as modificações necessárias.

Em primeiro lugar, indicaremos o que desejamos filtrar `Console.WriteLine("Listando Documentos somente de ficção científica");` .

A condição específica neste caso não será `Gte` , mas sim `AnyEq` . Buscaremos especificamente o assunto e, por fim, informaremos a condição, resultando na seguinte linha:

```
condicao = construtor.AnyEq(x => x.Assunto, "ficcao cientifica")
```

Desta forma, serão listados todos os livros em que, pelo menos, um dos assuntos seja ficção científica.

```
Console.WriteLine("Listando Documentos ano publicacao seja maior ou igual a 1999 e que tenham ma
construtor = Builders<Livro>.Filter;
condicao = constructor.Gte(x => x.Ano, 1999) & constructor.Gte(x => x.Paginas, 300);

listaLivros = await conexaoBiblioteca.Livros.Find(condicao).ToListAsync();
foreach (var doc in listaLivros)
{
    Console.WriteLine(doc.ToJson<Livro>());
}
Console.WriteLine("Fim da Lista");
Console.WriteLine("");
Console.WriteLine("");

Console.WriteLine("Listando Documentos somente de ficção científica");
construtor = Builders<Livro>.Filter;
condicao = construtor.AnyEq(x = > x.Assunto, "Ficção Científica");

listaLivros = await conexaoBiblioteca.Livros.Find(condicao).ToListAsync();
foreach (var doc in listaLivros)
{
    Console.WriteLine(doc.ToJson<Livro>());
}
Console.WriteLine("Fim da Lista");
Console.WriteLine("");
Console.WriteLine("");
```

Em seguida, salvaremos o projeto e o executaremos, clicando em "Iniciar".

Surgirá uma janela onde serão exibidos, na parte inferior, todos os documentos em que pelo menos um dos assuntos seja ficção científica.