

Listando livros usando como filtro um documento BSON

Transcrição

Aprenderemos a realizar buscas utilizando critérios específicos.

Primeiro iremos adicionar a nova classe chamada `listandoDocumentosFiltroBson`.

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.

Utilizaremos como base o programa estudado anteriormente:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using MongoDB.Bson;
using MongoDB.Driver;

namespace exemploMongoDB
{
    class listandoDocumentosFiltroBson
    {
        static void Main(string[] args)
        {
            Task T = MainAsync(args);
            Console.WriteLine();
            Console.WriteLine("Pressione Enter");
            Console.ReadLine();
        }
        static async Task MainAsync(string[] args)
        {
            var conexaoBiblioteca = new conectandoMongoDB();
            Console.WriteLine("Listando Documentos");

            var listaLivros = await conexaoBiblioteca.Livros.find(new BsonDocument()).ToListAsync();
            foreach (var doc in listaLivros)
            {
                Console.WriteLine(doc.toJson<Livro>());
            }

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

Nele, não utilizamos nenhum critério de busca, tendo definido como parâmetro simplesmente `new BsonDocument()`.

Como, neste caso, queremos um critério específico, criaremos uma nova variável `filtro` que receberá `new BsonDocument()`, removendo a referência citada acima.

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

Se salvarmos e executarmos o programa neste ponto, ele exibirá todos os documentos do banco.

Se olharmos o exemplo anterior `manipulandoDocumentos.cs`, vemos que manipulamos documentos BSON. Por exemplo, na declaração:

```
var doc = new BsonDocument
{
    {"Título", "Guerra dos Tronos"}
}
```

A variável `doc` recebe as informações `Título` e `Guerra dos Tronos` e, se visualizarmos este documento, leremos somente estas duas informações em formato JSON.

Retornando ao exemplo atual, iremos copiar o seguinte trecho:

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

E colar ao final do nosso código.

Neste trecho que colamos, iremos listar apenas documentos em que o autor seja `Machado de Assis`.

Removeremos a referência à variável, apagando a palavra reservada `var`, pois já foi feita acima, e em vez de declararmos um documento vazio (`new BsonDocument`), incluiremos os parâmetros de busca.

Para separarmos as listas, inseriremos duas linhas em branco antes da segunda busca.

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

Filtro = new BsonDocument
{
    {"Autor", "Machado de Assis"}
};
```

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

Feito isso, executaremos o programa.

Surgirá uma caixa de diálogo onde, inicialmente, veremos a lista completa de livros e, logo abaixo, teremos a lista contendo apenas as obras escritas por Machado de Assis.

Trata-se de um critério de busca utilizando um `BsonDocument`.