13

Usando o modelo OO com MongoDB

Transcrição

No exemplo do vídeo anterior criamos uma variável do tipo "JSON", construímos o documento manualmente e inserimos no MongoDB.

Na prática, utilizaremos outra abordagem. Criaremos um novo objeto dentro do Visual Studio, uma classe, que receberá valores e será inserida diretamente no MongoDB.

Essa classe deverá ter uma correspondência ao documento "JSON".

Há um site que pode nos ajudar, principalmente nos primeiros passos.

Abriremos novamente o arquivo "JSON" contendo a informação sobre o livro.

```
{

 "Titulo":"Guerra dos Tronos",
 "Autor":"George R R Martin",
 "Ano":1999,
 "Paginas":856
 "Assunto": [
 "Fantasia",
 "Ação"

 ]
}
```

Copiaremos o texto, e entraremos no site Json2CSharp (http://json2csharp.com/).

Nele, colaremos o texto copiado na caixa de diálogo e clicaremos em "Generate". Automaticamente ele criará a classe correspondente.

```
public class RootObject
{
    public string Titulo {get; set}
    public string Autor {get; set}
    public int Ano {get; set}
    public int Paginas {get; set}
    public List<string> Assunto {get; set}
}
```

Criaremos a referida classe e incluiremos dados no MongoDB por meio dela.

Abrindo nosso projeto, clicaremos com o botão direito do mouse sobre o nome do arquivo, selecionaremos "Adicionar > Novo Item".

A classe se chamará "Livro" e deverá ser pública, permitindo que seja acessada por outras classes. Faremos isso digitando `public` antes de `class Livro`.

O primeiro passo é adicionar a conexão com a biblioteca do MongoDB, inserindo `using MongoDB.Bson`.

Há ainda uma segunda bilbioteca `using MongoDB.Bson.Serialization.Attributes` que devemos acrescentar, principalmente às classes que representam documentos JSON a serem manipulados no MongoDB.

Copiaremos a classe gerada no site Json2CSharp e colaremos no código.

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

namespace exemplosMongoDB
{
    public class Livro
    {
        public string Titulo {get; set}
        public string Autor {get; set}
        public int Ano {get; set}
        public int Paginas {get; set}
        public List<string> Assunto {get; set}
    }
}
```

Ainda falta uma informação importante.

Todo documento "JSON" cria um `ObjectId` automaticamente sempre que não for feita menção específica, sendo possível ainda criá-lo manualmente.

Precisamos representar este objeto dentro da classe. Faremos isso utilizando o comando `public string Id {get; set;}`.

Fazendo a declaração desta forma, teremos problemas na sua manipulação, porque, dentro do MongoDB, ele é do tipo object string, e não `string`. É um tipo especial que somente o MongoDB manipula.

Precisaremos incluir isso no programa, declarando `[BsonRepresentation(BsonType.ObjectId)]`. Isto indica ao programa que o `Id` será um tipo `ObjectId`, apesar de ser interpretado como um tipo `string` para o .NET.

Dessa maneira é possível fazer as manipulações diretamente no MongoDB.

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

```
namespace exemplosMongoDB
{
    public class Livro
    {
        [BsonRepresentation(BsonType.ObjectId)]
        public string Id {get; set;}

        public string Titulo {get; set}
        public string Autor {get; set}
        public int Ano {get; set}
        public int Paginas {get; set}
        public List<string> Assunto {get; set}
    }
}
```

Agora retornaremos ao exemplo da aula anterior, onde substituiremos a variável do tipo `BsonDocument` pela classe que acabamos de criar.

Criaremos uma nova classe para nosso projeto, clicando com o botão direito do mouse em "exemplosMongoDB" e selecionando "Adicionar > Novo Item". Ela se chamará "manipulandoClasses".

O resultado será:

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

namespace exemploMongoDB
{
    class manipulandoClasses
```

Copiaremos o código do "acessandoMongoDB.cs", visto anteriormente, e colaremos na nova classe.

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

namespace exemploMongoDB
{
    class manipulandoClasses
    {
        static void Main(string[] args)
        {
            Task T = MainSync(args);
            Console.WriteLine("Pressione ENTER")
```

```
            Console.ReadLine();
        }

        static async Task MainSync(string[] args);
        {
            {
//    "Título":"Guerra dos Tronos",
//    "Autor":"George R R Martin",
//    "Ano":1999,
//    "Páginas":856
//    "Assunto": [
//        "Fantasia",
//        "Ação"
            }
]
}
        var doc = new BsonDocument
            {
                {"Título", "Guerra dos Tronos"}
                {"Autor", "George R R Martin"}
                {"Ano", "1999"}
                {"Páginas", "856"}
            };

        var assuntoArray = new BsonArray ();
        assuntoArray.Add ("Fantasia");
        assuntoArray.Add ("Ação");
        doc.Add("Assunto", assuntoArray);

        Console.WriteLine(doc);
    }
}

        // acesso ao servidor do MongoDB

        string stringConexao = "mongodb://localhost":27017;
        IMongoClient cliente = new MongoClient(stringConexao);

        // acesso ao banco de dados

        IMongoDatabase bancoDados = cliente.GetDatabase("Biblioteca")

        // acesso a coleção

        IMongoCollection<BsonDocument> colecao = bancoDados.GetCollection<Bsondocument>("Livros'

        //incluindo documento

        await colecao.InsertOneAsync(doc);

        Console.WriteLine("Documento Incluido");
```

A primeira modificação a ser feita é na variável `Bson.Document` . Não vamos mais usá-la na forma como está, mas sim como uma variável do tipo objeto livro `Livro livro = new Livro ()` .

Iremos inicializá-la manualmente, declarando o título, autor, ano, número de páginas e assuntos.

Lembrado que `Assuntos` é do tipo lista de `strings`, criaremos uma variável `List<string> listaAssuntos = new List<string()`.

Em seguida listaremos os assuntos.

Com a variável preenchida, indicaremos que `livro.Assunto` receberá `listaAssuntos`.

```
var doc = new BsonDocument
//            {
//                {"Título", "Guerra dos Tronos"}
//                {"Autor", "George R R Martin"}
//                {"Ano", "1999"}
//                {"Páginas", "856"}
//            };
//
//        var assuntoArray = new BsonArray ();
//        assuntoArray.Add ("Fantasia");
//        assuntoArray.Add ("Ação");
//        doc.Add("Assunto", assuntoArray);

//        Console.WriteLine(doc);

// inicializar uma variável do tipo objeto livro

Livro livro = new Livro ();
livro.Titulo = "Sob a redoma";
livro.Autor = "Stephen King";
livro.Ano = "2012"
livro.Paginas = "679"

List<string> listaAssuntos = new List<string();
listaAssuntos.Add("Ficção Científica");
listaAssuntos.Add("Terror");
listaAssuntos.Add("Ação");
livro.Assunto = listaAssuntos;
```

Faremos então uma modificação na forma como acessamos o MongoDB, já que, originalmente, nossa coleção era de `BsonDocument`. Agora ela passará a ser de objetos da classe `Livros`.

Em todos os lugares onde há referência a `BsonDocument` trocaremos por `Livro`.

```
        // acesso a coleção

        IMongoCollection<Livro> colecao = bancoDados.GetCollection<Livro>("Livros");
```

A variável `livro` possui a informação do novo livro que será adicionado à biblioteca, portanto iremos substituir a variável do tipo "BSON" `doc` por essa nova.

```
        //incluindo documento

        await colecao.InsertOneAsync(livro);

        Console.WriteLine("Documento Incluido");
```

Portanto nosso programa ficou da seguinte forma:

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

namespace exemploMongoDB
{
    class manipulandoClasses
    {
        static void Main(string[] args)
        {
            Task T = MainSync(args);
            Console.WriteLine("Pressione ENTER")
            Console.ReadLine();
        }

        static async Task MainSync(string[] args);
        {
            {
//    "Título":"Guerra dos Tronos",
//    "Autor":"George R R Martin",
//    "Ano":1999,
//    "Páginas":856
//    "Assunto": [
//        "Fantasia",
//        "Ação"
            }
]
}
        var doc = new BsonDocument
//            {
//                {"Título", "Guerra dos Tronos"}
//                {"Autor", "George R R Martin"}
//                {"Ano", "1999"}
//                {"Páginas", "856"}
//            };
//
//        var assuntoArray = new BsonArray ();
//        assuntoArray.Add ("Fantasia");
//        assuntoArray.Add ("Ação");
//        doc.Add("Assunto", assuntoArray);

//        Console.WriteLine(doc);

//      inicializar uma variável do tipo objeto livro

        Livro livro = new Livro ();
        livro.Titulo = "Sob a redoma";
        livro.Autor = "Stephen King";
        livro.Ano = "2012"
```

```
livro.Paginas = "679"

List<string> listaAssuntos = new List<string();
listaAssuntos.Add("Ficção Científica");
listaAssuntos.Add("Terror");
listaAssuntos.Add("Ação");
livro.Assunto = listaAssuntos;

// acesso ao servidor do MongoDB

string stringConexao = "mongodb://localhost":27017;
IMongoClient cliente = new MongoClient(stringConexao);

// acesso ao banco de dados

IMongoDatabase bancoDados = cliente.GetDatabase("Biblioteca")

// acesso a coleção

IMongoCollection<Livro> colecao = bancoDados.GetCollection<Livro>("Livros");

//incluindo documento

await colecao.InsertOneAsync(livro);

Console.WriteLine("Documento Incluido");
```

Salvaremos o arquivo e posteriormente clicaremos com o botão direito do mouse sobre nosso projeto, selecionaremos "Propriedades" e, em "Objeto de Inicialização", definiremos como "exemplosMongoDB.manipulandoClasses".

Agora executaremos nosso projeto, clicando no botão "Iniciar", na barra de menu superior.

Uma vez executado, será exibida a mensagem "Documento Incluido".

Abriremos o Robomongo e, clicando duas vezes com o botão esquerdo do mouse sobre a coleção "Livros", veremos dois documentos.

O primeiro documento é aquele que inserimos no formato `JSON`, com os dados do livro "Guerra dos Tronos", e em seguida temos o segundo documento, com os dados do livro "Sob a Redoma".

O que vimos foram duas maneiras de se inserir documentos no MongoDB, por meio de documentos `JSON` no `BsonDocument`, ou por classes.