06

Manipulando documentos JSON no .NET

Transcrição

Falaremos agora sobre documentos "JSON".

Aqueles que já estão familiarizados com o MongoDB sabem que a sua biblioteca armazena documentos no formato "JSON" na forma de JavaScript, e as informações são delimitadas por meio de chaves e colchetes.

Para ilustrar isso, trabalharemos com um arquivo neste formato onde lê-se:

```
{
    "Título":"Guerra dos Tronos",
    "Autor":"George R R Martin",
    "Ano":1999,
    "Páginas":856
    "Assunto": [
        "Fantasia",
        "Ação"
]
}
```

Temos um exemplo de arquivo "JSON", contendo informações sobre livros. Vemos o título e autor, em formato string, o ano de publicação, número de páginas e um array de assuntos, no caso fantasia e ação.

Retornando ao Visual Studio, demonstraremos como são criados documentos do tipo "JSON" por meio do .NET utilizando o Mongo Driver.

Utilizaremos um modelo de programa assíncrono ao qual vamos adicionar um novo item. Para isso vamos clicar com o botão direito do mouse sobre o diretório "exemplosMongoDB" e selecionar "Adicionar > Novo Item", definiremos como uma Classe em C#. Daremos a ela o nome de "manipulandoDocumentos.cs". Agora vamos clicar em "Adicionar" para finalizar o processo.

Nosso programa assíncrono foi construído da seguinte forma:

```
using System;
using System.Collections.Generic;
using System.Ling;
using System.Text;
using System.Threading.Tasks;

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

```
        static async Task MainSync(string[] args);
        {
            Console.WriteLine("Esperando 10 segundos ....");
            await Task.Delay(10000);
            Console.WriteLine("Esperei 10 segundos ....");
        }
    }
}
```

Para nosso exemplo da biblioteca de livros vamos fazer algumas alterações a este código.

A primeira coisa que faremos é sinalizar que vamos utilizar uma parte da bilbioteca do Mongo Driver, por meio do comando `using MongoDB.Bson` .

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

Vamos utilizar "BSON", que fornece algumas funções para manipulação de documentos "JSON".

Em seguida iremos colar o nosso modelo de programa, ainda em comentários, apenas para termos um guia para nos ajudar.

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

namespace exemploMongoDB
{
    class programAssincrono
    {
        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"
]
}
            Console.WriteLine("Esperando 10 segundos ....");
            await Task.Delay(10000);
            Console.WriteLine("Esperei 10 segundos ....");
        }
    }
}
```

Agora criaremos uma variável, do tipo vetor, onde teremos em primeiro lugar a informação sobre o título do livro.

Portanto, ainda dentro da função `static async Task MainSync(string[] args)` adicionaremos um documento "BSON". Utilizando portanto uma função da `MongoDB.Bson` , que é o nosso tipo "JSON".

Embaixo do "BSON" vamos indicar o comando `Console.WriteLine(doc)` .

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

namespace exemploMongoDB
{
    class programAssincrono
    {
        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"}
                };

            Console.WriteLine(doc);
        }
```

```
        }
    }
```

Salvaremos o arquivo desta forma para, em seguida, executar-lo.

Para isso clicaremos com o botão direito do mouse no diretório com o nome do arquivo "exemplosMongoDB", em seguida selecionaremos "Propriedades".

Alteramos o item "Objeto de Inicialização" para que seja executado o "exemplosMongoDB.manipulandoDocumentos". A seguir clicaremos em "Iniciar", na barra de menu superior.

O resultado que temos é o título "Guerra dos Tronos" sendo exibido.

Ainda não é esse o resultado final - o documento "JSON" é mais complexto.

Num documento do tipo `BsonDocumento` utilizaremos, por exemplo, o comando `doc.Add("Autor", "George R R Martin")` , adicionando mais uma propriedade.

Se salvarmos e repetirmos o procedimento para execução, veremos que o documento "JSON" já possui as duas primeiras propriedades, sendo elas o "título" e o "autor".

Ainda precisamos adicionar mais informações a este documento. Serão elas o ano e páginas.

Dados que são números inteiros e, portanto, não são strings.

```
                var doc = new BsonDocument
                {
                    {"Título", "Guerra dos Tronos"}
                    {"Autor", "George R R Martin"}
                    {"Ano", "1999"}
                    {"Páginas", "856"}
                };

            Console.WriteLine(doc);
        }
    }
}
```

Salvaremos e executaremos novamente. Serão exibidos o título, autor, ano e número de páginas. Falta adicionarmos o tópico "Assunto", que é um array com dois elementos, uma variável do tipo `new BsonArray` .

Com a nova variável criada iremos adicionar um novo documento:

```
                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);
        }
    }
}
```

Em seguida iremos salvar e executar. Serão exibidos na tela o título, autor, ano, número de páginas do livro e ainda o assunto.

Temos basicamente o mesmo formato do documento "JSON" no exemplo inicial, mas utilizando o .NET com funções do tipo "BSON", que está dentro da biblioteca MongoDB.

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

namespace exemploMongoDB
{
    class programAssincrono
    {
        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);
        }
    }
```