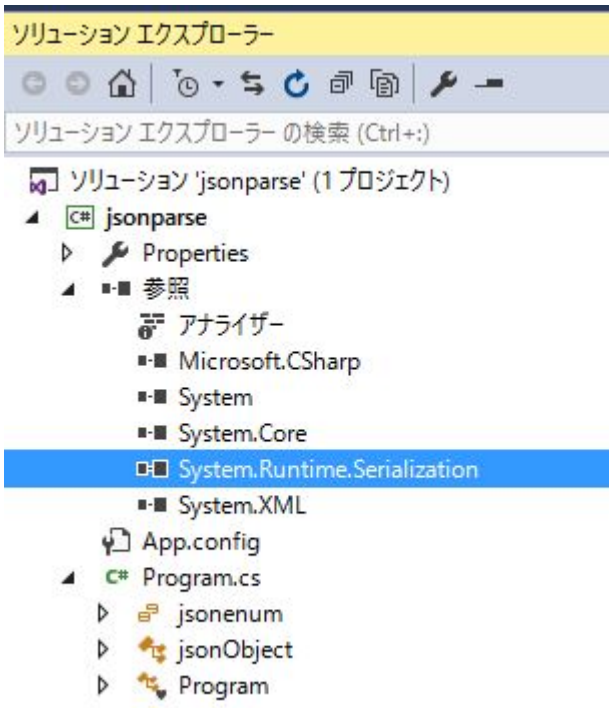
C#でJSONを読み書きしてみる

2016/12/11
自分用メモ

下準備

参照にアセンブリ System.Runtime.Serialization を追加しておく。



クラスをJSONでシリアライズする

C#の属性でシリアライズするクラスやメンバーを明示する“[DataContract]“や ” [EnumMember]””[DataMember]”がそれ。

```
using System;
using System.IO;
using System.Text;
using System.Collections;
using System.Runtime.Serialization.Json;
using System.Runtime.Serialization;

namespace jsonparse
{
    [DataContract]
    public enum jsonenum
    {
```

```
        [EnumMember] key1,
        [EnumMember] key2,
        [EnumMember] key3
    };
    [DataContract]
    public class jsonObject
    {
        [DataMember] public bool keyBool = true;
        [DataMember] public int keyInt = 360;
        [DataMember] public double keyDouble = 3.141592653;
        [DataMember] public string keyString = "this is json";
        [DataMember] public jsonenum keyEnum = jsonenum.key2;
        [DataMember] public int[] keyArraryInt = new int[] { 10, 100, 100 };
        [DataMember] public ArrayList keyArrayList = new ArrayList() { 1980,
"abcd" };
        [DataMember] public Hashtable keyHashtable = new Hashtable() { {
"key100", 100 }, { "key200", 200 } };
    }
    class Program
    {
        static void Main(string[] args)
        {
            jsonObject json = new jsonObject();

            DataContractJsonSerializer js = new
DataContractJsonSerializer(typeof(jsonObject));
            MemoryStream ms = new MemoryStream();
            js.WriteObject(ms, json);

            Encoding e = Encoding.UTF8;

            Console.WriteLine(e.GetString(ms.GetBuffer()));
        }
    }
}
```

コンパイルして jsonparse.exe **を作ったので実行してみる。**

```
E:\work>jsonparse.exe
{"keyArraryInt":[10,100,100],"keyArrayList":[1980,"abcd"],"keyBool":true,"ke
yDouble":3.141592653,"keyEnum":1,"keyHashtable":[{"Key":"key100","Value":100
},{"Key":"key200","Value":200}],"keyInt":360,"keyString":"this is json"}

E:\work>
```

出力**されたJSONを** JSON Pretty Linter - JSONの整形と構文チェック **で整形するとこうなる。**

出力されたJSON	対応するjsonObjectのメンバ

{
"keyArraryInt": [
10,
100,
100
],
"keyArrayList": [
1980,
"abcd"
],
"keyBool": true,
"keyDouble": 3.141592653,
"keyEnum": 1,
"keyHashtable": [
{
"Key": "key100",
"Value": 100
},
{
"Key": "key200",
"Value": 200
}
],
"keyInt": 360,
"keyString": "this is json"
}
	[DataMember] public int[] keyArraryInt = new int[] { 10, 100, 100 };
	[DataMember] public ArrayList keyArrayList = new ArrayList() { 1980, "abcd" };
	[DataMember] public bool keyBool = true;
	[DataMember] public double keyDouble = 3.141592653;
	[DataMember] public jsonenum keyEnum = jsonenum.key2;
	[DataMember] public Hashtable keyHashtable = new Hashtable() { { "key100", 100 }, { "key200", 200 } };
	[DataMember] public int keyInt = 360;
	[DataMember] public string keyString = "this is json";

enum は JavaScript に無いので名称ではなく値で表現される。 値をしてしていない jsonenum はkey1□key3のメンバーに0～2が割り当てされている。 配列やリストは、要素の並びが正しく反映されている。

デシリアライズでJSONの内容をクラスに反映させる

入力サンプルのJSONコードを用意する。

sample.json

```
{
    "keyArraryInt": [
        5000,
        200,
        300
    ],
    "keyArrayList": [
        "ABCD",
        2016
    ],
```

```
    "keyBool": false,
    "keyDouble": 1.41421356,
    "keyEnum": 0,
    "keyHashtable": [
        {
            "Key": "key300",
            "Value": 300
        },
        {
            "Key": "key500",
            "Value": 500
        },
        {
            "Key": "key800",
            "Value": 800
        }
    ],
    "keyInt": 180,
    "keyString": "これはJSONですね"
}
```

先のプログラムを書き換えてコンパイル、実行してみる。

```
using System;
using System.IO;
using System.Collections;
using System.Runtime.Serialization.Json;
using System.Runtime.Serialization;

namespace jsonparse
{
    [DataContract]
    public enum jsonenum
    {
        [EnumMember] key1,
        [EnumMember] key2,
        [EnumMember] key3
    };
    [DataContract]
    public class jsonObject
    {
        [DataMember] public bool keyBool = true;
        [DataMember] public int keyInt = 360;
        [DataMember] public double keyDouble = 3.141592653;
        [DataMember] public string keyString = "this is json";
        [DataMember] public jsonenum keyEnum = jsonenum.key2;
        [DataMember] public int[] keyArraryInt = new int[] { 10, 100, 100 };
        [DataMember] public ArrayList keyArrayList = new ArrayList() { 1980,
"abcd" };
        [DataMember] public Hashtable keyHashtable = new Hashtable() { {
```

```
"key100", 100 }, { "key200", 200 } };
    }
    class Program
    {
        static void Main(string[] args)
        {

            DataContractJsonSerializer js = new
DataContractJsonSerializer(typeof(jsonObject));
            FileStream fs = new FileStream("sample.json", FileMode.Open);
            jsonObject json = (jsonObject)js.ReadObject(fs);
            fs.Close();

            Console.WriteLine("keyBool      = {0}", json.keyBool);
            Console.WriteLine("keyInt       = {0}", json.keyInt);
            Console.WriteLine("keyDouble    = {0}", json.keyDouble);
            Console.WriteLine("keyString    = {0}", json.keyString);
            Console.WriteLine("keyEnum      = {0}", json.keyEnum);
            Console.WriteLine("keyArrayInt  = {0}", String.Join(", ",
json.keyArraryInt));
            Console.WriteLine("keyArrayList = {0}", String.Join(", ",
json.keyArrayList.ToArray()));

            ArrayList sa = new ArrayList();
            foreach(DictionaryEntry x in json.keyHashtable)
            {
                sa.Add(x.Key.ToString() + "->"  + x.Value.ToString());
            }
            Console.WriteLine("keyHashtable = {0}", String.Join(", ",
sa.ToArray()));

        }
    }
}
```

実行結果から、クラス jsonObject の初期値がJSONドキュメントで更新されていることが分かる。

```
C:\work>jsonparse.exe
keyBool      = False
keyInt       = 180
keyDouble    = 1.41421356
keyString    = これはJSONですね
keyEnum      = key1
keyArrayInt  = 5000, 200, 300
keyArrayList = ABCD, 2016
keyHashtable = key800->800, key500->500, key300->300

C:\work>
```

JSONからNULLを入力してみる

もし入力にNULLが含まれる場合、デシリアライズするクラスはNULL許容するメンバで構成されている必要がある。

```
using System;
using System.IO;
using System.Runtime.Serialization.Json;
using System.Runtime.Serialization;

namespace jsonparse
{
    [DataContract]
    public enum jsonenum
    {
        [EnumMember] key1,
        [EnumMember] key2,
        [EnumMember] key3
    };
    [DataContract]
    public class jsonObject
    {
        [DataMember] public bool? keyBool = true;
        [DataMember] public int? keyInt = 360;
        [DataMember] public double? keyDouble = 3.141592653;
        [DataMember] public string keyString = "this is json";
        [DataMember] public jsonenum? keyEnum = jsonenum.key2;
    }
    class Program
    {
        static void Main(string[] args)
        {
            DataContractJsonSerializer js = new
DataContractJsonSerializer(typeof(jsonObject));
            FileStream fs = new FileStream("sample.json", FileMode.Open);
            jsonObject json = (jsonObject)js.ReadObject(fs);
            fs.Close();

            Console.WriteLine("keyBool      = {0}", json.keyBool );
            Console.WriteLine("keyInt       = {0}", json.keyInt );
            Console.WriteLine("keyDouble    = {0}", json.keyDouble );
            Console.WriteLine("keyString    = {0}", json.keyString );
            Console.WriteLine("keyEnum      = {0}", json.keyEnum );
        }
    }
}
```

bool,int,double,enum で ”？”を付与しているのはNULL許容型の宣言□stringは元々NULLを許容している。 結果は以下の通り。

JSON入力	結果

{
"keyBool": null,
"keyInt": null,
"keyDouble": null,
"keyString": null,
"keyEnum": null
}
	C:\work>jsonparse.exe
	keyBool =
	keyInt =
	keyDouble =
	keyString =
	keyEnum =
	
	C:\work>

{
"keyBool": false,
"keyInt": 270,
"keyDouble": 2.77,
"keyString": "This is a pen.",
"keyEnum": 2
}
	C:\work>jsonparse.exe
	keyBool = False
	keyInt = 270
	keyDouble = 2.77
	keyString = This is a pen.
	keyEnum = key3
	
	C:\work>

C#, 技術資料, json