Newtonsoft JsonSerializer - Lower case properties and dictionary [duplicate]

Asked 3 years, 9 months ago Active 1 year, 5 months ago Viewed 46k times

47

1

This question already has an answer here:
Keep casing when serializing dictionaries *4 answers*

I'm using json.net (Newtonsoft's JsonSerializer). I need to customize serialization in order to meet following requirements:

1. property names must start with lower case letter.
2. Dictionary must be serialized into jsonp where keys will be used for property names. LowerCase rule does not apply for dictionary keys.

for example:

```
var product = new Product();
procuct.Name = "Product1";
product.Items = new Dictionary<string, Item>();
product.Items.Add("Item1", new Item { Description="Lorem Ipsum" });
```

must serialize into:

```
{
  name: "Product1",
  items : {
    "Item1": {
       description : "Lorem Ipsum"
    }
  }
}
```

notice that property Name serializes into "name", but key Item1 serializes into "Item1";

I have tried to create CustomJsonWriter to serialize property names, but it changes also dicionary keys.

```
        public CustomJsonWriter(TextWriter writer) : base(writer)
        {

        }
        public override void WritePropertyName(string name, bool escape)
        {
            if (name != "$type")
            {
                name = name.ToCamelCase();
            }
            base.WritePropertyName(name, escape);
        }
    }
```

c# json.net

edited Mar 27 '18 at 23:33
abatishchev
72.1k 70 269 404

asked Dec 3 '15 at 15:55
Liero
10.8k 8 57 131

marked as duplicate by Brian Rogers json.net Dec 3 '15 at 17:47

This question has been asked before and already has an answer. If those answers do not fully address your question, please ask a new question.

2 Answers

103

You could try using the `CamelCasePropertyNamesContractResolver` .

```
var serializerSettings = new JsonSerializerSettings();
serializerSettings.ContractResolver = new CamelCasePropertyNamesContractResolver();
var json = JsonConvert.SerializeObject(product, serializerSettings);
```

I'm just not sure how it'll handle the dictionary keys and I don't have time right this second to try it. If it doesn't handle the keys correctly answered Dec 3 '15 at 16:33

Craig W.
13k 4 39 75

9 Be careful with this option. This option cause lowercasing keys of the serialized dictionaries as well. This is usually is not what is desired. – user1936595 Mar 6 '18 at 2:16

2 This question's answer tells how you can implement your own ContractResolver for example for using lower case. stackoverflow.com/questions/6288660/… – Mark Baijens Sep 24 '18 at 9:42

5 If you only want the casing to apply to one object you can use: `var json = JsonConvert.SerializeObject(product, new JsonSerializerSettings { ContractResolver = new CamelCasePropertyNamesContractResolver(), Formatting = Formatting.Indented });` – PutoTropical Jan 25 at 17:16

40

You can use a JsonProperty to change how something is serialized/deserialized. When you define your object add the property items to the fields you would like represented differently in the JSON. This only works with NewtonsoftJSON. Other libraries may do it differently.

```
public class Product
{
    [JsonProperty("name")]
    public string Name { get; set; }

    [JsonProperty("items")]
    public Dictionary<string, Item> Items { get; set; }
}

public class Item
{
    [JsonProperty("description")]
    public string Description { get; set; }
}
```

answered Dec 3 '15 at 16:27

Brian from state farm
2,278 8 15

10 This is not a violation. It provides a mapping which may be necessary when dealing with data serialization. This is due to mismatch between C# variable names and the underlying serialization format - not limited to JSON. – Metro Feb 12 '17 at 16:28

of any single element of a system does not require a change in other logically unrelated elements. Additionally, elements that are logically related all change predictably and uniformly, and are thus kept in sync." – Liero Jul 8 '17 at 17:00

2 In this case the knowledge is property name and the convention used for serialization. But since you specified both property name and related serialized property name, they are not kept in sync. In your example, changing one element requires (property name) requires change to the other element (serialized property name). Clearly a violation of DRY – Liero Jul 8 '17 at 17:01