How to update the value stored in Dictionary in C#?

Ask Question

378

How to update value for a specific key in a dictionary `Dictionary<string, int>` ?

c# dictionary

39

edited Feb 1 at 7:00
Cœur
20k 10 117 158

asked Aug 7 '09 at 9:02
Amit
10.5k 23 62 100

I have a complex type stored as value in dictionary. When I want to change a property of an stored value, I get CS1612. Therefore I must take a way around: var v = dict[c]; v.dprop = c.sprop; dict[c] = v; – peter70 Nov 16 '18 at 15:03

6 Answers

Home

PUBLIC

Stack Overflow

Tags

Users

Jobs

Teams

✓

edited Nov 28 '12 at 11:12
IT ppl
2,253 1 34 49

answered Aug 7 '09 at 9:04
ccalboni
7,624 5 23 34

So this simple method seems also to be a better substitute for the well known ".Add" and ".TryGetValue" method without the necessity to change the value. (?) At least, if it doesn't matter to overwrite keys, for example in a situation where it is not excluded that keys are written more than once in a loop. Or does somebody see any disadvantages? Especially because .Add has the ptifall for beginners that if the if-wrapper or TryGetValue is forgotten, it can work fine in tests and with other test data where the .Add is called twice for the same key it will throw an exception. – Philm Jan 14 '17 at 9:52

the interesting point of this operation is that, it UPSERT (key, value) into dictionary. brilliant! – Soren Jan 16 '17 at 7:51

1 As Pini stated, this should be the answer to the question. due the right thing and change it. – Leo Gurdian Mar 17 '17 at 18:53

183

It's possible by accessing the key as index

for example:

```
Dictionary<string, int> dictionary = new Dictionary<string
dictionary["test"] = 1;
```

answered Aug 7 '09 at 9:33

Amit
10.5k 23 62 100

10 If there is no item "test" in the List, then list["test"] = list["test"] + 1; will raise KeyNotFoundException! The pure assignment of a non existing indexer will work. list["test"] = 1; – Steven Spyrka May 26 '15 at 9:38

Can you also use list["test"]++;? – aufty Jan 8 '16 at 18:02

14 dont call a dictionary list, call it dogs or cats or dict – user3800527 Mar 17 '16 at 9:35

1 @aufty you can write `++dictionary["test"];` or `dictionary["test"]++;` but only if there is an entry in the dictionary with the key value "test" — example: `if(dictionary.ContainsKey("test")) ++dictionary["test"]; else dictionary["test"] = 1; // create entry with key "test"` – gerryLowry May 20 '18 at 0:29

41

You can follow this approach:

```
void addOrUpdate(Dictionary<int, int> dic, int key, int new
{
    int val;
    if (dic.TryGetValue(key, out val))
    {
        // yay, value exists!
        dic[key] = val + newValue;
    }
    else
```

The edge you get here is that you check and get the value of corresponding key in just 1 access to the dictionary. If you use `ContainsKey` to check the existance and update the value using `dic[key] = val + newValue;` then you are accessing the dictionary twice.

answered Jun 17 '14 at 4:19
max_force
489 5 11

3 Instead of `dic.Add(key, newValue);` you can use use `dic[key] = newvalue;`. – Macke Apr 13 '15 at 4:15

What happens if you do "dic[key] = value" and "key" doesn't exist? – superpuccio Aug 3 '15 at 11:41

1 @superpuccio you get a KeyNotFoundException – ntroncos Sep 21 '15 at 23:52

7 @ntroncos not true, it will add that key to the dictionary with the value provided. += will not work on a non-existing key though, since it is just syntactic sugar for dic[key] = value + dic[key]. – lastas Dec 18 '15 at 14:26

2 This should be the answer to the question as it regards updating the dictionary not just adding to it. – The Lonely Coder Dec 16 '16 at 9:41

13

Use LINQ: Access to dictionary for the key and change the value

```
Dictionary<string, int> dict = new Dictionary<string, int>
```

answered Jun 26 '15 at 12:20

INT_24h
827 7 6

I don't even understand how this works but it's amazing – hexagonest Aug 26 '15 at 10:55

1 Creating another dictionary doesn't make sense to me for such simple thing. Check ccalboni's answer. – RollerCosta Feb 9 '17 at 7:49

1 i think it is a good answer. it doesn't require you to know each key string – Joseph Wu Apr 10 '17 at 23:15

9

Here is a way to update by an index much like `foo[x] = 9` where `x` is a key and 9 is the value

```
var views = new Dictionary<string, bool>();

foreach (var g in grantMasks)
{
    string m = g.ToString();
    for (int i = 0; i <= m.Length; i++)
    {
        views[views.ElementAt(i).Key] = m[i].Equals('1') ?
    }
}
```

edited Nov 28 '17 at 17:09

Dean Seo
3,017 2 19 41

'14 at 11:26

I dont know how efficient is this logic, but I like the For loop idea. :) – open and free Nov 4 '14 at 14:37

0

This may work for you:

Scenario 1: primitive types

```
string keyToMatchInDict = "x";
int newValToAdd = 1;
Dictionary<string,int> dictToUpdate = new Dictionary<strin

if(!dictToUpdate.ContainsKey(keyToMatchInDict))
   dictToUpdate.Add(keyToMatchInDict ,newValToAdd );
else
   dictToUpdate.Where(kvp=>kvp.Key==keyToMatchInDict).Firs
==newValToAdd; //or you can do operations such as ...Value
```

Scenario 2: The approach I used for a List as Value

```
int keyToMatch = 1;
AnyObject objInValueListToAdd = new AnyObject("something f
Dictionary<int,List<AnyObject> dictToUpdate = new Dictiona
//imagine this dict got initialized before with valid Keys

if(!dictToUpdate.ContainsKey(keyToMatch))
   dictToUpdate.Add(keyToMatch,new List<AnyObject>{objInVa
else

dictToUpdate.Where(kvp=>kvp.Key==keyToMatch).FirstOrDefaul
```

answered Oct 30 '18 at 11:11

Mister Pitt

1