Casting string to enum [duplicate]

Asked 6 years, 8 months ago Active 2 years, 1 month ago Viewed 114k times

72

9

This question already has an answer here:
[Convert a string to an enum in C#](#) *22 answers*

I'm reading file content and take string at exact location like this

```
 string fileContentMessage = File.ReadAllText(filename).Substring(411, 3);
```

Output will always be either `Ok` or `Err`

On the other side I have `MyObject` which have `ContentEnum` like this

```
public class MyObject

    {
      public enum ContentEnum { Ok = 1, Err = 2 };
      public ContentEnum Content { get; set; }
    }
```

Now, on the client side I want to use code like this (to cast my string `fileContentMessage` to `Content` property)

```
string fileContentMessage = File.ReadAllText(filename).Substring(411, 3);

    MyObject myObj = new MyObject ()
    {
       Content = /// ///,
    };
```

c# string casting enums

Developer
18.5k 61 241 409

user1765862
4,977 18 69 132

marked as duplicate by Patrick Hofman c# Oct 3 '14 at 13:36

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.

1 What is your question exactly? – TimVK Dec 20 '12 at 10:40

Enum.TryParse(string value, out variable); msdn.microsoft.com/en-us/library/dd783499(v=vs.100).aspx – ryadavilli Dec 20 '12 at 10:42

4 Answers

168 ✓

Use `Enum.Parse()` .

```
var content = (ContentEnum)Enum.Parse(typeof(ContentEnum), fileContentMessage);
```

edited Jul 20 '17 at 12:12

answered Dec 20 '12 at 10:40
CodeCaster
114k 18 162 207

4 `Enum.Parse` returns an `object` which needs to be casted. I felt free to edit :) – Matthias Meid Dec 20 '12 at 10:42

Perfect, a one-line solution. Thanks! This saves me from writing another method to handle it. – John Suit Nov 11 '14 at 15:20

1 I tried with Parity Enum and when I give the string value of 123 it returns it without issue, even though Parity enum does not include 123 value of enum – John Demetriou Aug 7 '17 at 8:31

@JohnDemetriou that's documented: *"If value is the string representation of an integer that does not represent an underlying value of the enumType enumeration, the method returns an enumeration member whose underlying value is value converted to an integral type. If this behavior is undesirable, call the IsDefined method to ensure that a particular string representation of an integer is actually a member of enumType"*. – CodeCaster Aug 7 '17 at 8:52

26

As an extra, you can take the `Enum.Parse` answers already provided and put them in an easy-to-use static method in a helper class.

```
public static T ParseEnum<T>(string value)
{
    return (T)Enum.Parse(typeof(T), value, ignoreCase: true);
}
```

And use it like so:

```
{
   Content = ParseEnum<ContentEnum>(fileContentMessage);
};
```

Especially helpful if you have lots of (different) Enums to parse.

answered Dec 20 '12 at 10:47
pleinolijf
648 10 28

Thanks for the ignoreCase: true tidbit! – Colin Jan 24 '17 at 0:31

21

.NET 4.0+ has a generic Enum.TryParse

```
ContentEnum content;
Enum.TryParse(fileContentMessage, out content);
```

answered Dec 20 '12 at 11:00
Chris Fulstow
32.2k 7 77 104

Have a look at using something like

> Converts the string representation of the name or numeric value of one or more enumerated constants to an equivalent enumerated object. A parameter specifies whether the operation is case-sensitive. The return value indicates whether the conversion succeeded.

or

Enum.Parse

> Converts the string representation of the name or numeric value of one or more enumerated constants to an equivalent enumerated object.

answered Dec 20 '12 at 10:41

Adriaan Stander
137k 25 249 264