

Re: ToUpper() Better solution

Source:

<http://www.tech-archive.net/Archive/DotNet/microsoft.public.dotnet.general/2007-03/msg00060.html>

- *From:* "Chris Mullins [MVP]" <cmullins@xxxxxxxxxx>
 - *Date:* Thu, 1 Mar 2007 17:22:35 -0800
-

The closest thing that comes to mind is an RFC called stringprep. There are a wide variety of stringprep profiles, and while they don't quite do what you're looking for, they're close. Included in stringprep is a set of mapping tables for Uppder->Lower case conversions. These are (in that context) called case-foldings, are found in table B.2. Unfortunately, they're Upper->Lower, not the other way around.

Stringprep:

<http://www.faqs.org/rfcs/rfc3454.html>

There are a number of profiles:

[Profile for Internaional Domain Names]

<http://www.rfc-editor.org/rfc/rfc3491.txt>

[Profile for iSCSI names]

<http://tools.ietf.org/html/draft-ietf-ips-iscsi-string-prep-01>

[Profile for SASL UserNames & Passwords]

<http://www.ietf.org/rfc/rfc4013.txt>

[Profile for XMPP Resources]

<http://www.xmpp.org/internet-drafts/attic/draft-ietf-xmpp-resourceprep-02.html>

There's a C# implementation of this RFC that's part of the libidn library.

There's also a C++ & Java version.

<http://www.gnu.org/software/libidn/>

We've actually got a full implementation of stringprep as well – it's much more .Net 2.0 ish than the libidn one, which is just a native C++ app that was then ported to Java & .Net. It's found in our open-source SoapBox Framework.

—

Chris Mullins, MCSD.NET, MCPD:Enterprise, Microsoft C# MVP

<http://www.coversant.com/blogs/cmullins>

"Jon Skeet [C# MVP]" <skeet@xxxxxxxxxx> wrote in message
news:MPG.2050e52a399075e398d906@xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

Re: ToUpper() Better solution

Ornette <abstrait...nospam...@xxxxxxx> wrote:

So how would you do ?

The mapping table idea you had before looked best to me, although I wouldn't quite implement it the same way. I'd have a look up table for every possible character, where it defaults to the Unicode character, but for all the accented characters you care about, you specify the non-accented version.

You'd then call ToCharArray() on the string in question, go through each character replacing the original with the mapped character, and then create a new string with the char array.

It does require you to manually map all the accented characters you care about though.

My guess is that there are libraries around to do this somewhere, but I don't know of any myself.

--

Jon Skeet - <skeet@xxxxxxxx>

<http://www.pobox.com/~skeet> Blog: <http://www.msmvps.com/jon.skeet>

If replying to the group, please do not mail me too