How does Stack Overflow generate its SEO-friendly URLs?

Asked 16 years, 4 months ago Modified 2 years, 3 months ago Viewed 46k times

269

What is a good complete regular expression or some other process that would take the title:

> How do you change a title to be part of the URL like Stack Overflow?

and turn it into

```
how-do-you-change-a-title-to-be-part-of-the-url-like-stack-overflow
```

that is used in the SEO-friendly URLs on Stack Overflow?

The development environment I am using is Ruby on Rails, but if there are some other platform-specific solutions (.NET, PHP, Django), I would love to see those too.

I am sure I (or another reader) will come across the same problem on a different platform down the line.

I am using custom routes, and I mainly want to know how to alter the string to all special characters are removed, it's all lowercase, and all whitespace is replaced.

regex language-agnostic seo friendly-url slug

Share
Improve this question
Follow

edited Jan 9, 2020 at 4:15
bb216b3acfd8f72cbc8f8 99d4d6963
763 ● 11 ● 21

asked Aug 24, 2008 at 18:21
wusher
12.4k ● 22 ● 73 ● 96

What about funny characters? What are you going to do about those? Umlauts? Punctuation? These need to be considered. Basically, I would use a white-list approach, as opposed to the black-list approaches above: Describe which characters you will allow, which characters you will convert (to what?) and then change the rest to something meaningfull (""). I doubt you can do this in one regex... Why not just loop through the characters?
– Daren Thomas Aug 24, 2008 at 18:58

1 Should be migrated to meta; as the question and answer both specifically deal with SO implementation, and the accepted answer is from @JeffAtwood. – casperOne Nov 18, 2011

at 20:21

19 @casperOne Do you think Jeff is not allowed some non-meta reputation? The question is about "how can one do something like this", not specifically "how is this done here". – Paŭlo Ebermann Nov 19, 2011 at 13:05

@PaŭloEbermann: It's not about Jeff getting some non-meta reputation (how much reputation he has is really not my concern); the question body *specifically referenced StackOverflow's implementation* hence the rationale for it being on meta. – casperOne Nov 22, 2011 at 14:04

21 Answers

Sorted by: Highest score (default)

323

Here's how we do it. Note that there are probably more edge conditions than you realize at first glance.

This is the second version, unrolled for 5x more performance (and yes, I benchmarked it). I figured I'd optimize it because this function can be called hundreds of times per page.

```
/// <summary>
/// Produces optional, URL-friendly version of a title, "like-this-one".
/// hand-tuned for speed, reflects performance refactoring contributed
/// by John Gietzen (user otac0n)
/// </summary>
public static string URLFriendly(string title)
{
    if (title == null) return "";

    const int maxlen = 80;
    int len = title.Length;
    bool prevdash = false;
    var sb = new StringBuilder(len);
    char c;

    for (int i = 0; i < len; i++)
    {
        c = title[i];
        if ((c >= 'a' && c <= 'z') || (c >= '0' && c <= '9'))
        {
            sb.Append(c);
            prevdash = false;
        }
        else if (c >= 'A' && c <= 'Z')
        {
            // tricky way to convert to lowercase
            sb.Append((char)(c | 32));
            prevdash = false;
        }
        else if (c == ' ' || c == ',' || c == '.' || c == '/' ||
            c == '\\' || c == '-' || c == '_' || c == '=')
        {
            if (!prevdash && sb.Length > 0)
            {
                sb.Append('-');
                prevdash = true;
            }
```

To see the previous version of the code this replaced (but is functionally equivalent to, and 5x faster), view revision history of this post (click the date link).

Also, the `RemapInternationalCharToAscii` method source code can be found here.

Share

Improve this answer

Follow

edited Mar 20, 2017 at 10:29
Community Bot
1 ● 1

answered Aug 25, 2008 at 0:11
Jeff Atwood
63.9k ● 48 ● 151 ● 153

38

Here is my version of Jeff's code. I've made the following changes:

- The hyphens were appended in such a way that one could be added, and then need removing as it was the last character in the string. That is, we never want "my-slug-". This means an extra string allocation to remove it on this edge case. I've worked around this by delay-hyphening. If you compare my code to Jeff's the logic for this is easy to follow.
- His approach is purely lookup based and missed a lot of characters I found in examples while researching on Stack Overflow. To counter this, I first peform a normalisation pass (AKA collation mentioned in Meta Stack Overflow question *Non US-ASCII characters dropped from full (profile) URL*), and then ignore any characters outside the acceptable ranges. This works most of the time...
- ... For when it doesn't I've also had to add a lookup table. As mentioned above, some characters don't map to a low ASCII value when normalised. Rather than drop these I've got a manual list of exceptions that is doubtless full of holes, but it is better than nothing. The normalisation code was inspired by Jon Hanna's great post in Stack Overflow question *How can I remove accents on a string?*.
- The case conversion is now also optional.

```
public static class Slug
{
    public static string Create(bool toLower, params string[] values)
    {
        return Create(toLower, String.Join("-", values));
    }

    /// <summary>
    /// Creates a slug.
    /// References:
    /// http://www.unicode.org/reports/tr15/tr15-34.html
    /// https://meta.stackexchange.com/questions/7435/non-us-ascii-
characters-dropped-from-full-profile-url/7696#7696
    /// https://stackoverflow.com/questions/25259/how-do-you-include-a-
webpage-title-as-part-of-a-webpage-url/25486#25486
    /// https://stackoverflow.com/questions/3769457/how-can-i-remove-
accents-on-a-string
```

```
    /// </summary>
    /// <param name="toLower"></param>
    /// <param name="normalised"></param>
    /// <returns></returns>
    public static string Create(bool toLower, string value)
    {
        if (value == null)
            return "";

        var normalised = value.Normalize(NormalizationForm.FormKD);

        const int maxlen = 80;
        int len = normalised.Length;
        bool prevDash = false;
        var sb = new StringBuilder(len);
        char c;

        for (int i = 0; i < len; i++)
        {
            c = normalised[i];
```

For more details, the unit tests, and an explanation of why Facebook's URL scheme is a little smarter than Stack Overflows, I've got an expanded version of this on my blog.

Share
Improve this answer
Follow

edited May 23, 2017 at 12:26
Community Bot
1 ● 1

answered Jul 18, 2011 at 23:11
DanH
3,802 ● 2 ● 28 ● 31

16

You will want to setup a custom route to point the URL to the controller that will handle it. Since you are using Ruby on Rails, here is an introduction in using their routing engine.

In Ruby, you will need a regular expression like you already know and here is the regular expression to use:

```
def permalink_for(str)
    str.gsub(/[^\w\/]|[!\(\)\.]+/, ' ').strip.downcase.gsub(/\ +/, '-')
end
```

Share
Improve this answer
Follow

edited Jul 10, 2013 at 11:59
Peter Mortensen
31.6k ● 22 ● 109 ● 133

answered Aug 24, 2008 at 18:24
Dale Ragan
18.3k ● 3 ● 55 ● 71

11

You can also use this JavaScript function for in-form generation of the slug's (this one is based on/copied from Django):

```
function makeSlug(urlString, filter) {
    // Changes, e.g., "Petty theft" to "petty_theft".
    // Remove all these words from the string before URLifying
    if(filter) {
        removelist = ["a", "an", "as", "at", "before", "but", "by", "for",
"from",
        "is", "in", "into", "like", "of", "off", "on", "onto", "per",
        "since", "than", "the", "this", "that", "to", "up", "via", "het",
"de", "een", "en",
        "with"];
    }
    else {
        removelist = [];
    }
    s = urlString;
    r = new RegExp('\\b(' + removelist.join('|') + ')\\b', 'gi');
    s = s.replace(r, '');
    s = s.replace(/[^-\w\s]/g, ''); // Remove unneeded characters
    s = s.replace(/^\s+|\s+$/g, ''); // Trim leading/trailing spaces
    s = s.replace(/[-\s]+/g, '-'); // Convert spaces to hyphens
    s = s.toLowerCase(); // Convert to lowercase
    return s; // Trim to first num_chars characters
}
```

Share

Improve this answer

Follow

edited Jul 10, 2013 at 16:29

Peter Mortensen
31.6k ● 22 ● 109 ● 133

answered Sep 1, 2008 at 13:16

fijter
18k ● 2 ● 26 ● 28

8

For good measure, here's the PHP function in WordPress that does it... I'd think that WordPress is one of the more popular platforms that uses fancy links.

```
    function sanitize_title_with_dashes($title) {
            $title = strip_tags($title);
            // Preserve escaped octets.
            $title = preg_replace('|%([a-fA-F0-9][a-fA-F0-9])|', '---$1---
', $title);
            // Remove percent signs that are not part of an octet.
            $title = str_replace('%', '', $title);
            // Restore octets.
            $title = preg_replace('|---([a-fA-F0-9][a-fA-F0-9])---|',
'%$1', $title);
            $title = remove_accents($title);
            if (seems_utf8($title)) {
                    if (function_exists('mb_strtolower')) {
                            $title = mb_strtolower($title, 'UTF-8');
                    }
                    $title = utf8_uri_encode($title, 200);
            }
            $title = strtolower($title);
            $title = preg_replace('/&.+?;/', '', $title); // kill entities
            $title = preg_replace('/[^%a-z0-9 _-]/', '', $title);
            $title = preg_replace('/\s+/', '-', $title);
            $title = preg_replace('|-+|', '-', $title);
            $title = trim($title, '-');
            return $title;
    }
```

This function as well as some of the supporting functions can be found in wp-includes/formatting.php.

Share Improve this answer Follow

answered Aug 25, 2008 at 1:20

The How-To Geek
1,105 ● 9 ● 13

7 This is not full answer. You are missing functions like : `remove_accents` , `seems_utf8` ...
– Nikola Loncar Jun 11, 2014 at 9:47

to complete @The How-To Geek answer you still can `git clone git://core.git.wordpress.org/` and find the `wp-includes/formatting.php` file into
– mickro Jun 9, 2017 at 16:54

5

If you are using Rails edge, you can rely on Inflector.parametrize - here's the example from the documentation:

```
  class Person
    def to_param
      "#{id}-#{name.parameterize}"
    end
  end

  @person = Person.find(1)
  # => #<Person id: 1, name: "Donald E. Knuth">

  <%= link_to(@person.name, person_path(@person)) %>
  # => <a href="/person/1-donald-e-knuth">Donald E. Knuth</a>
```

Also if you need to handle more exotic characters such as accents (éphémère) in previous version of Rails, you can use a mixture of PermalinkFu and DiacriticsFu:

```
DiacriticsFu::escape("éphémère")
=> "ephemere"

DiacriticsFu::escape("räksmörgås")
=> "raksmorgas"
```

Share
Improve this answer
Follow

edited Dec 27, 2013 at 17:18
Amal Murali
76.6k ● 18 ● 132 ● 153

answered Dec 30, 2008 at 9:59
Thibaut Barrère
8,873 ● 2 ● 24 ● 27

5

I am not familiar with Ruby on Rails, but the following is (untested) PHP code. You can probably translate this very quickly to Ruby on Rails if you find it useful.

```
$sURL = "This is a title to convert to URL-format. It has 1 number in it!";
// To lower-case
$sURL = strtolower($sURL);

// Replace all non-word characters with spaces
$sURL = preg_replace("/\W+/", " ", $sURL);

// Remove trailing spaces (so we won't end with a separator)
$sURL = trim($sURL);

// Replace spaces with separators (hyphens)
$sURL = str_replace(" ", "-", $sURL);

echo $sURL;
// outputs: this-is-a-title-to-convert-to-url-format-it-has-1-number-in-it
```

I hope this helps.

Share
Improve this answer
Follow

edited Jan 2, 2014 at 15:13

answered Aug 24, 2008 at 18:41
Vegard Larsen
13k ● 14 ● 60 ● 102

4

I don't much about Ruby or Rails, but in Perl, this is what I would do:

```
my $title = "How do you change a title to be part of the url like
Stackoverflow?";

my $url = lc $title;   # Change to lower case and copy to URL.
$url =~ s/^\s+//g;     # Remove leading spaces.
$url =~ s/\s+$//g;     # Remove trailing spaces.
$url =~ s/\s+/\-/g;    # Change one or more spaces to single hyphen.
$url =~ s/[^\w\-]//g;  # Remove any non-word characters.

print "$title\n$url\n";
```

I just did a quick test and it seems to work. Hopefully this is relatively easy to translate to Ruby.

Share Improve this answer Follow

answered Aug 24, 2008 at 18:48
Brian
717 ● 1 ● 8 ● 12

4

T-SQL implementation, adapted from dbo.UrlEncode:

```
CREATE FUNCTION dbo.Slug(@string varchar(1024))
RETURNS varchar(3072)
AS
BEGIN
    DECLARE @count int, @c char(1), @i int, @slug varchar(3072)
    SET @string = replace(lower(ltrim(rtrim(@string))),' ','-')

    SET @count = Len(@string)
    SET @i = 1
    SET @slug = ''

    WHILE (@i <= @count)
    BEGIN
        SET @c = substring(@string, @i, 1)

        IF @c LIKE '[a-z0-9--]'
            SET @slug = @slug + @c

        SET @i = @i +1
    END

    RETURN @slug
END
```

Share Improve this answer Follow

answered Sep 6, 2008 at 16:29

Sören Kuklau
19.9k ● 8 ● 55 ● 90

4

I know it's very old question but since most of the browsers now **support unicode urls** I found a great solution in **XRegex** that converts everything except letters (in all languages to '-').

That can be done in several programming languages.

The pattern is `\\p{^L}+` and then you just need to use it to replace all non letters to '-'.

Working example in node.js with xregex module.

```
var text = 'This ! can @ have # several $ letters % from different
languages such as עברית or Español';

var slugRegEx = XRegExp('((?!\\d)\\p{^L})+', 'g');

var slug = XRegExp.replace(text, slugRegEx, '-').toLowerCase();

console.log(slug) ==> "this-can-have-several-letters-from-different-
languages-such-as-עברית-or-español"
```

Share

Improve this answer

Follow

edited Sep 30, 2015 at 14:47

answered Sep 26, 2015 at 7:15

Rotem
2,356 ● 4 ● 26 ● 45

3

Assuming that your model class has a title attribute, you can simply override the to_param method within the model, like this:

```
def to_param
  title.downcase.gsub(/ /, '-')
end
```

This Railscast episode has all the details. You can also ensure that the title only contains valid characters using this:

```
validates_format_of :title, :with => /^[a-z0-9-]+$/,
                    :message => 'can only contain letters, numbers and
hyphens'
```

Share Improve this answer Follow

answered Aug 24, 2008 at 18:49

John Topley
115k ● 47 ● 199 ● 240

2

Brian's code, in Ruby:

```
title.downcase.strip.gsub(/\ /, '-').gsub(/[^\w\-]/, '')
```

`downcase` turns the string to lowercase, `strip` removes leading and trailing whitespace, the first `gsub` call *g*lobally *sub*stitutes spaces with dashes, and the second removes everything that isn't a letter or a dash.

Share Improve this answer Follow

answered Aug 24, 2008 at 19:03

Sören Kuklau
19.9k ● 8 ● 55 ● 90

2

There is a small Ruby on Rails plugin called PermalinkFu, that does this. The escape method does the transformation into a string that is suitable for a URL. Have a look at the code; that method is quite simple.

To remove non-ASCII characters it uses the iconv lib to translate to 'ascii//ignore//translit' from 'utf-8'. Spaces are then turned into dashes, everything is downcased, etc.

Share

Improve this answer

Follow

edited Jan 1, 2014 at 11:47

Peter Mortensen
31.6k ● 22 ● 109 ● 133

answered Sep 1, 2008 at 13:13

Lau
153 ● 7

While this works perfectly, I somehow feel it isn't very efficient. – WhyNotHugo Jul 20, 2012 at 1:29

2

You can use the following helper method. It can convert the Unicode characters.

```
public static string ConvertTextToSlug(string s)
{
    StringBuilder sb = new StringBuilder();

    bool wasHyphen = true;

    foreach (char c in s)
    {
        if (char.IsLetterOrDigit(c))
        {
            sb.Append(char.ToLower(c));
            wasHyphen = false;
        }
        else
            if (char.IsWhiteSpace(c) && !wasHyphen)
            {
                sb.Append('-');
                wasHyphen = true;
            }
    }

    // Avoid trailing hyphens
    if (wasHyphen && sb.Length > 0)
        sb.Length--;

    return sb.ToString().Replace("--","-");
}
```

Share

Improve this answer

Follow

edited Jan 1, 2014 at 12:14

Peter Mortensen
31.6k ● 22 ● 109 ● 133

answered Mar 27, 2012 at 22:28

Peyman Mehrabani
729 ● 7 ● 18

2

Here's my (slower, but fun to write) version of Jeff's code:

```
public static string URLFriendly(string title)
{
    char? prevRead = null,
        prevWritten = null;

    var seq =
        from c in title
        let norm =
RemapInternationalCharToAscii(char.ToLowerInvariant(c).ToString())[0]
        let keep = char.IsLetterOrDigit(norm)
        where prevRead.HasValue || keep
        let replaced = keep ? norm
```

```
              :  prevWritten != '-' ? '-'
              :  (char?)null
          where replaced != null
          let s = replaced + (prevRead == null ? ""
              : norm == '#' && "cf".Contains(prevRead.Value) ? "sharp"
              : norm == '+' ? "plus"
              : "")
          let _ = prevRead = norm
          from written in s
          let __ = prevWritten = written
          select written;

    const int maxlen = 80;
    return string.Concat(seq.Take(maxlen)).TrimEnd('-');
}

public static string RemapInternationalCharToAscii(string text)
{
    var seq = text.Normalize(NormalizationForm.FormD)
        .Where(c => CharUnicodeInfo.GetUnicodeCategory(c) !=
UnicodeCategory.NonSpacingMark);

    return string.Concat(seq).Normalize(NormalizationForm.FormC);
}
```

My test string:

`"  I love C#, F#, C++, and... Crème brûlée!!! They see me codin'... they hatin'... tryin' to catch me codin' dirty... "`

Share

Improve this answer

Follow

edited Apr 10, 2015 at 0:47

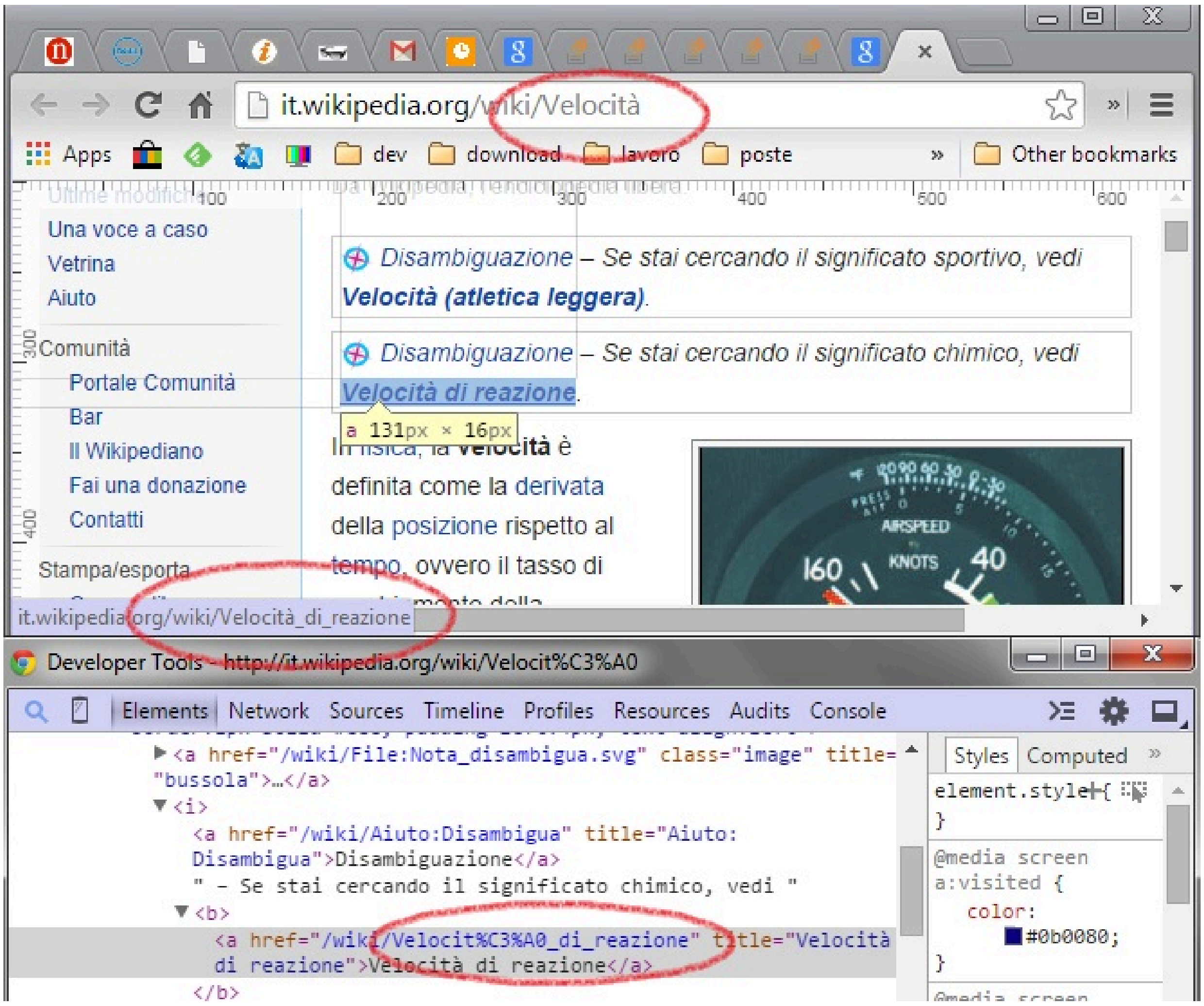
answered Apr 10, 2015 at 0:31

Ronnie Overby

46.5k ● 73 ● 271 ● 350

2

The stackoverflow solution is great, but modern browser (excluding IE, as usual) now handle nicely utf8 encoding:


it.wikipedia.org/wiki/Velocità

Apps dev download lavoro poste Other bookmarks

Una voce a caso
Vetrina
Aiuto
Comunità
Portale Comunità
Bar
Il Wikipediano
Fai una donazione
Contatti
Stampa/esporta

Disambiguazione – Se stai cercando il significato sportivo, vedi **Velocità (atletica leggera)**.

Disambiguazione – Se stai cercando il significato chimico, vedi **Velocità di reazione**.

a 131px × 16px

In fisica, la **velocità** è definita come la derivata della posizione rispetto al tempo, ovvero il tasso di

it.wikipedia.org/wiki/Velocità_di_reazione

Developer Tools - http://it.wikipedia.org/wiki/Velocit%C3%A0

Elements Network Sources Timeline Profiles Resources Audits Console

```
▶<a href="/wiki/File:Nota_disambigua.svg" class="image" title=
"bussola">…</a>
▼<i>
  <a href="/wiki/Aiuto:Disambigua" title="Aiuto:
  Disambigua">Disambiguazione</a>
  " – Se stai cercando il significato chimico, vedi "
 ▼<b>
   <a href="/wiki/Velocit%C3%A0_di_reazione" title="Velocità
   di reazione">Velocità di reazione</a>
  </b>
```

Styles Computed

```
element.style {
}
@media screen
a:visited {
   color: ■#0b0080;
}
```

So I upgraded the proposed solution:

```
public static string ToFriendlyUrl(string title, bool useUTF8Encoding =
false)
{
    ...

        else if (c >= 128)
        {
            int prevlen = sb.Length;
            if (useUTF8Encoding )
            {

sb.Append(HttpUtility.UrlEncode(c.ToString(CultureInfo.InvariantCulture),Encoc
            }
            else
            {
                sb.Append(RemapInternationalCharToAscii(c));
            }
    ...
}
```

[Full Code on Pastebin](#)

Edit: [Here's the code](#) for `RemapInternationalCharToAscii` method (that's missing in the pastebin).

Share

Improve this answer

Follow

edited May 23, 2017 at 12:10

Community Bot

1 ● 1

answered Jul 31, 2014 at 15:43

giammin

19k ● 9 ● 73 ● 90

According to Wikipedia, Mozilla 1.4, Netscape 7.1, Opera 7.11 were among the first applications to support IDNA. A browser plugin is available for Internet Explorer 6 to provide IDN support. Internet Explorer 7.0 and Windows Vista's URL APIs provide native support for IDN. Sounds like removing UTF-8 characters is a waste of time. Long live UTF-8!!!
– Muhammad Rehan Saeed Apr 24, 2015 at 14:10

1

I liked the way this is done without using regular expressions, so I ported it to PHP. I just added a function called `is_between` to check characters:

```
function is_between($val, $min, $max)
{
    $val = (int) $val; $min = (int) $min; $max = (int) $max;

    return ($val >= $min && $val <= $max);
}

function international_char_to_ascii($char)
{
    if (mb_strpos('àåáâäãåa', $char) !== false)
    {
        return 'a';
    }

    if (mb_strpos('èéêëe', $char) !== false)
    {
        return 'e';
    }

    if (mb_strpos('ìíîïi', $char) !== false)
    {
        return 'i';
    }

    if (mb_strpos('òóôõö', $char) !== false)
    {
        return 'o';
    }

    if (mb_strpos('ùúûüuu', $char) !== false)
    {
        return 'u';
    }

    if (mb_strpos('çccc', $char) !== false)
    {
        return 'c';
    }
```

Share

Improve this answer

edited Jan 1, 2014 at 13:10

answered Dec 6, 2013 at 12:17

Follow

Peter Mortensen
31.6k ● 22 ● 109 ● 133

user1765062

1

Now all Browser handle nicely utf8 encoding, so you can use [WebUtility.UrlEncode](#) Method , its like [HttpUtility.UrlEncode](#) used by @giamin but its work outside of a web application.

Share Improve this answer Follow

answered May 14, 2015 at 16:51

ikourfaln
91 ● 1 ● 3

1

I ported the code to TypeScript. It can easily be adapted to JavaScript.

I am adding a `.contains` method to the `String` prototype, if you're targeting the latest browsers or ES6 you can use `.includes` instead.

```
if (!String.prototype.contains) {
    String.prototype.contains = function (check) {
        return this.indexOf(check, 0) !== -1;
    };
}

declare interface String {
    contains(check: string): boolean;
}

export function MakeUrlFriendly(title: string) {
            if (title == null || title == '')
                return '';

            const maxlen = 80;
            let len = title.length;
            let prevdash = false;
            let result = '';
            let c: string;
            let cc: number;
            let remapInternationalCharToAscii = function (c: string) {
                let s = c.toLowerCase();
                if ("àåáâäãåą".contains(s)) {
                    return "a";
                }
                else if ("èéêëę".contains(s)) {
                    return "e";
                }
                else if ("ìíîïı".contains(s)) {
                    return "i";
                }
                else if ("òóôõöøőð".contains(s)) {
                    return "o";
                }
                else if ("ùúûüŭů".contains(s)) {
                    return "u";
                }
```

Share Improve this answer Follow

answered Apr 18, 2018 at 21:39

Sam
976 ● 7 ● 22

0

No, no, no. You are all so very wrong. Except for the diacritics-fu stuff, you're getting there, but what about Asian characters (shame on Ruby developers for not considering their nihonjin brethren).

Firefox and Safari both display non-ASCII characters in the URL, and frankly they look great. It is nice to support links like 'http://somewhere.com/news/read/お前たちはアホじゃないかい'.

So here's some PHP code that'll do it, but I just wrote it and haven't stress tested it.

```
<?php
    function slug($str)
    {
        $args = func_get_args();
        array_filter($args);  //remove blanks
        $slug = mb_strtolower(implode('-', $args));

        $real_slug = '';
        $hyphen = '';
        foreach(SU::mb_str_split($slug) as $c)
        {
            if (strlen($c) > 1 && mb_strlen($c)===1)
            {
                $real_slug .= $hyphen . $c;
                $hyphen = '';
            }
            else
            {
                switch($c)
                {
                    case '&':
                        $hyphen = $real_slug ? '-and-' : '';
                        break;
                    case 'a':
                    case 'b':
                    case 'c':
                    case 'd':
                    case 'e':
                    case 'f':
                    case 'g':
                    case 'h':
                    case 'i':
                    case 'j':
                    case 'k':
                    case 'l':
                    case 'm':
                    case 'n':
                    case 'o':
```

Example:

```
$str = "~!@#$%^&*()_+-=[]\{}|;':\",./<>?\n\r\t\x07\x00\x04 コリン ~!@#$%^&*
()_+-=[]\{}|;':\",./<>?\n\r\t\x07\x00\x04 トーマス ~!@#$%^&*()_+-=[]\
{}|;':\",./<>?\n\r\t\x07\x00\x04 アーノルド ~!@#$%^&*()_+-=[]\{}|;':\",./<>?
\n\r\t\x07\x00\x04";
echo slug($str);
```

Outputs: コリン-and-トーマス-and-アーノルド

The '-and-' is because &'s get changed to '-and-'.

Share

Improve this answer

Follow

edited Jan 1, 2014 at 12:02

Peter Mortensen
31.6k ● 22 ● 109 ● 133

answered Mar 14, 2009 at 1:12

Colin Thomas-Arnold

4 I really don't know what to say about this piece of information. – sjas Jul 15, 2012 at 17:27

3 That's a really good example of when NOT to use a switch case statement. – NickG Dec 18, 2014 at 17:07

-1

Rewrite of Jeff's code to be more concise

```
    public static string RemapInternationalCharToAscii(char c)
    {
        var s = c.ToString().ToLowerInvariant();

        var mappings = new Dictionary<string, string>
        {
            { "a", "àåáâäãåą" },
            { "c", "çćčĉ" },
            { "d", "đ" },
            { "e", "èéêëę" },
            { "g", "ğĝ" },
            { "h", "ĥ" },
            { "i", "ìíîïı" },
            { "j", "ĵ" },
            { "l", "ł" },
            { "n", "ñń" },
            { "o", "òóôõöøőð" },
            { "r", "ř" },
            { "s", "śşšŝ" },
            { "ss", "ß" },
            { "th", "Þ" },
            { "u", "ùúûüŭů" },
            { "y", "ýÿ" },
            { "z", "żźž" }
        };

        foreach(var mapping in mappings)
        {
            if (mapping.Value.Contains(s))
```

```
                return mapping.Key;
        }

        return string.Empty;
    }
```

Share Improve this answer Follow

answered Sep 10, 2020 at 12:06

David
727 ● 1 ● 6 ● 16

I don't really see the pint of a dictionary if you have to scann it entirely.. why not just a list? if that list of structured data (i see you use dictionary like a tuple). if the list is also a constant, why not define it as follow outside of the method? IMHO in such scenario is much more efficent a constant dictionary as yours but with key and value switched : 'à'-> 'a' , ... , 'ą'->'a', ..., 'ç' -> 'c', ... ,'ž' -> 'z' – Skary Sep 22, 2021 at 8:56