

Re: Fast UTF-8 strlen function

Source: <http://coding.derkeiler.com/Archive/Assembler/alt.lang.asm/2005-05/msg00566.html>

- *From:* "Beth" <BethStone21@xxxxxxxxxxxxxxxxxxxxxxxxxxxx>
 - *Date:* Tue, 17 May 2005 02:47:35 GMT
-

NoDot wrote:

> Sevag Krikorian wrote:

>> Isn't that the case with all languages? The vowel modifies the
>> consonant. It seems you already have the representations of Japanese
>> syllables using Latin characters.

>

> Um, it's a standard. They call it Romanji, IIRC.

Yeah, that's right...

>> I'm not talking about a new language (though that would be nice for
>> the linguists to figure out). I'm talking about people writing
>> their native language using the Latin-characters.

>

> Most languages already have that standardised, probably.

Yeah, that is, in fact, mostly the case...I knew a Vietnamese guy and there was a "standard" way to write Vietnamese stuff using ASCII characters...remember, ASCII has dominated for a while and, as such, "workarounds" already exist...

The whole point with UNICODE is that they DON'T have to do that retarded crap anymore...and can use their own native language directly in their own script...and also without nonsense of "code pages" or "special character sets" to do it either...

Point 1: The UNICODE character set itself is exactly like ASCII in the sense that there is a unique number for every character...it's just a bigger sized "ASCII chart", if you will...

The stuff about "encodings" is a SEPARATE ISSUE...these are just "standards" for storage and transmission...

Basically, as the world still has ASCII, then UTF-8 exists to provide UNICODE in a "ASCII compatible" way...

UTF-7 is designed to work with 7-bit transmissions, where that 8th bit might be "cropped off"...this is often called "Mail-safe" UNICODE...note that if all software and transmissions could be trusted to pass on the 8th

Re: Fast UTF-8 strlen function

bit of the bytes, then there'd be zero point for UTF-7 and it could be phased out...

Both of these are, in fact, "backwards compatibility" stuff...if it weren't for your precious ASCII having to still be supported, then UTF-8 wouldn't be there...if there wasn't software that only accepts 7-bit data (and people insisting that everything should be sent "7-bit" all the time...happily wasting 1 bit in every byte – because they are all bloody 8-bits now (the 7 bit and 9 bit byte are basically dead) – because they are still living in 1952 in their own minds) then there'd be no point to UTF-7...

This stuff exists for "backwards compatibility"...

As for UTF-16, that exists due to a major screw up by the UNICODE people...they started off saying "ah, we'll use 16-bit words"...then, half-way through, they realised that was not enough and "expanded" it...hence, really, UTF-16 is "screw up compatibility"...compatibility with their original 16-bit screw up...if they had only had a brain between them to, like, actually `_COUNT_` how many likely characters they needed then the fact that the "Han ideographs" (Chinese / Japanese / Korean) stuff `_BY ITSELF_` exceeds 64K easily would have tipped them off that 16-bits was just not big enough...this encoding is "backward idiots compatibility", so to speak...

Unfortunately, Microsoft jumped on board with UNICODE while they were still going along with this 16-bit screwed up idea...hence, UTF-16 (though, Windows doesn't actually implement that completely...more like "UNICODE cut off at 16-bits") is unfortunately an encoding that we're likely going to live with for a long time because Microsoft went and put that into their NT kernels directly...and "backwards compatibility" will ensure that it will refuse to die for at least the next 50 years or so, if not longer...perhaps even well beyond the demise of Microsoft...look at ASCII, it has "control characters" for controlling "dumb terminals" and that kind of nonsense...it's lived on way beyond that...

And UTF-32 is just the "formal" way of saying "using 32-bits per character"...which, in practice, means just using the UNICODE numbers directly, one dword per character...a kind of "non-encoding", if you will...this actually being like your favourite ASCII, just with much "wider" characters...

BUT note that all the "encodings" are encoding the `_SAME THING_`...the numbers for the characters are `_THE SAME_`...the encodings are just various "backwards compatible" ways of storing it that are suitable for different situations...UTF-8, as noted, is "ASCII compatible" (uses the unused 8th bit in ASCII to "escape" the other UNICODE characters beyond ASCII :)...if people weren't "obsessed" with ASCII and there wasn't all that ASCII software out there, then we could say "screw UTF-8" because that is what it's there for: "ASCII compatibility"...UTF-7 is because some software won't accept anything but 7-bit data (if there wasn't such software and

Re: Fast UTF-8 strlen function

they all took 8-bit data with no fuss, then this "encoding" would not be needed...generally speaking, it ISN'T needed most of the time, which is why it isn't mentioned much...but if some software refuses 8-bits and only transfers 7-bits, UTF-7 is the "work around" to get the "compatibility" there...though, UTF-7, unlike UTF-8, CAN get confused...basically, it uses "+" followed by a number...which, amusingly – when I switched it on to read one of Randy's posts – made the "UTC +0600" time zone thing (or whatever number it was) show up as a Kanji ideograph! UTF-7 is, thus, not really recommended as it can "confuse itself" but, well, if you can only send 7-bits then you can only send 7-bits and this is the "work around", however crap it can sometimes be)...

UTF-16 is "screw up compatibility"...so, if they'd never screwed up in the first place, it wouldn't exist...unfortunately, Microsoft were eager to rush into placing support for this "screw up" directly in their NT kernels...and, thus, it's the "encoding" which really shouldn't even exist in the first place...but, well, now that it's ended up in Windows, it'll probably end up proliferated everywhere and living 50 years beyond its "sell by" date...and, as is typical "form" for the computer industry, the "standard that never should have existed in the first place", which is "compatibility" with a total screw up will inevitably end up as the most common "encoding" ...it's that "deep irony" thing, you see...you cannot escape it! ;)

UTF-32 is a "non-encoding" in the sense that it's just the name for doing UNICODE "raw" and ASCII-like, with a simple "32-bits per character" thing...same basic way ASCII always worked, just the characters are 32-bits "wide", not just 8-bits...so, you know, it's not really an "encoding" in the same sense...it's just using the UNICODE numbers directly...but, well, as we have all these other "encodings", it's been given a "name" in order to distinguish it from the other "compatibility" crap...UTF-32 just says "NOT the other compatibility encodings"...

Hence, there aren't really "all these encodings"...there is UNICODE: It is a SINGLE character set with one unique number for every possible character...because of ASCII software and 7-bit software, UTF-7 and UTF-8 are added for "compatibility" with that (BUT the numbers they encode are exactly the same UNICODE numbers...these are just "tricks" with the storage to make them "ASCII compatible" / "7-bit compatible"...they are NOT different character sets...just different ways to store the same character set that tries to get "compatibility" with older stuff)...UTF-16 is an embarrassment that should never have happened but, well, that is the nature of "backwards compatibility", you've got to maintain even your most horrid, terrible screw ups forever with "compatibility"...as I've always said: "backwards compatibility" is aptly named because it is "compatibility" that's clearly "backwards" (in the "slightly retarded" sense of the word ;)...)

Point 2: Regards your "phonetic alphabet"...it exists...it's called the IPA ("International Phonetic Alphabet")...it uses a letter for every sound and is a "standard" commonly used by dictionaries in order to show

Re: Fast UTF-8 strlen function

pronunciation...

As for your idea of doing it in 27 letters to fit a keyboard? Nope, sorry, the IPA has more characters than that...you are underestimating the amount of phonetic sounds out there...

For example, the Welsh town of "Llanelli" (the double "l" is a diphthong in Welsh that, as far as I know, has no comparison in any other language...you place your tongue like an "l" sound but then exhale hard, pushing the air over your tongue...this creates, well, a kind of "Darth Vader breathing" sound...and, very often, a lot of spit flying out of your mouth...the Welsh master being able to say it without spitting on people but it takes practice ;)...

Or compare "th" in "the" with the "th" in "theme" ...ah, despite the same letters "th", there are actual TWO potential sounds...

Or consider an African surname like "Mbowe"...the "mb" is one sound, that's somewhat "m" and "b" at the same time...it takes practice to master...

And when it comes to vowels, this is NOT easy at all...English, in fact, is a terrible starting point for vowels because it actually has "double vowels" (or, at least, these seem like "two vowels back-to-back" in any other languages)...think of the "names" of the vowels when reciting the alphabet...in other languages, "e" is often a sound something like "air" (but without the ending and kind of "higher pitched", if that makes any sense)...I have to mention this because the first one: "a" is actually this "air-ee" (but, generally, English speakers aren't even aware or able to pronounce the "air" sound alone...it's not "natural" in English, you see, so "no practice" doing so...but, well, say the name of "a" but "crop off" the "ee" sound at the end...you can say it...as you noted, English speakers do often have trouble with some pronunciations, even though, ironically, they CAN say the sounds because they do exist in "combinations" with other sounds...your example of not being able to say "ts" directly but not hesitating to say "boots" is a good example of this kind of thing...ah, they can say it...they just aren't used to saying it alone or at the start of a word...but it can be mastered with enough practice, of course :)...then there's "i", which is "ah-ee"... "o" which is "oh-oo"...and "u" which is "ee-ew"...many English speakers will find it difficult, in fact, to even comprehend what it is I'm even talking about there...as these "double vowels" are ingrained in their minds...and, you know, they've NEVER once considered these as actually "two sound combinations"...but they are)...

Then there's the "obscure vowel"...it's not given its own letter in English but it is spoken all the time (indeed, more so in "flat" British English than American English but both do say it)...it sounds like "uh"...and is the "-er" sound in something like "worker"...which is a word which also demonstrates another sound, which is like the "urgh!" sound when you find something horrible or nasty...so, there's "oo-urgh-ck-uh" for you...you think you've "got all the vowels in English"? No way, jose...

Re: Fast UTF-8 strlen function

Re: Fast UTF-8 strlen function

What about the "rolled tongue" Swedish-chef-from-the-Muppets sounds...umlauted "o" and "u" in German...those ain't anywhere in English...

So, I've already brought in the "obscure vowel"...the "urgh!" vowel...and at least two "rolled tongue" variants, characteristic of the vowels are Nordic friends like to "birdie-birdie" with...you think we're done? Nope, sorry...

Because, for some weird reason, English has categorised "w" as being a "consonant"...sometimes it's called a "semi-vowel"...well, it may be used in a "consonant"-like way for spellings and such...but the actual phonetic sound it makes is 100% `_VOWEL_`: "ooh"...

Then, for all of these, don't forget there's "short" and "long" versions...oh, and, in some languages (such as in the Far East with Chinese), there's "really long" versions too...which, to everyone else, seem to go on for an eternity...you know, where you hear someone say a word and it ends with a "ah" vowel sound but – by the standards of other languages – isn't just "long" but is "extra long"..."yoshiiiiikaaaaaah" or something...

The word "worker" presented some challenges...what about "xylophone"? "X" said as "z", "y" as "long i", "ph" as "f"...and, of course, that great English device of the "e" at the end to "hint" that the vowel earlier in the word was a `_long_` vowel, not short...but which you don't actually ever pronounce...

What about "j" and "g"? "Gist" like "judge" but "got" is completely different..."c" like "s", even in the name of the letter itself (i.e. "the letter sea")...or "ch" as in "chuck" or "chocolate" (this seems one "ch" you left out and note that it is not a simple "k" sound, even if vaguely similar), while "chef" is a "sh" sound (because that's how the French say "ch" and that's where most of the words that start like that were "borrowed" from)...

And, yeah, speak of the devil, what about those "borrow words"? You say "pee-ts-ah", even though you write "pee-zah" for "pizza"...but you say "pee-zah" for "Pisa", when you've just written "pizah" (short "i")...of course, you're temporarily obey some of the pronunciation rules of the language from whence the "borrow word" came, which is why the pronunciation and spelling don't match up with what's "normal" in English (actually, when it comes to English, there is no real "normal"...it must be the `_LEAST_` phonetic language there is on the entire planet ;)...)

Do we even dare approach the "clicking throat" sounds of Klingon?

27 keys, you say? No chance!!!

Re: Fast UTF-8 strlen function

Someone's already done the work, though, as I say, about coming up with an "international standard" for phonetics that does cover all known sounds from different languages...it's called the IPA ("International Phonetic Alphabet") and is often used in dictionaries for pronunciation and by phoneticists:

<http://www2.arts.gla.ac.uk/IPA/ipa.html>

Here's a funny example (actually, it's supposed to be a UTF-8 "test page") of the lyrics of the Beastie Boys' "Intergalactic" rap in the IPA characters:

<http://www.cl.cam.ac.uk/~mgk25/ucs/examples/lyrics-ipa.txt>

The funny thing about the IPA is that, as the file notes, your "accent" can effect your phonetics...it's an alphabet that `_CAN_` actually notate "accents" directly...

See if you can work out how to read the file...it `_IS_` all in English...of course, the "hint" is to remember that it's a `_phonetic_` alphabet...hence, every character represents a particular sound and always is that sound, never changing...and if you know the lyrics (or Google for them), then you can compare (knowing what the English words sound like, anyway...and if you weren't sure, you could always play the actual song...NOT that any illegal file sharing of copyrighted material is endorsed or advocated by that suggestion...you, of course, already own the CD, as you're a big Beastie Boys fan, yes? ;)...work out what each character "sounds like" and then, work it out all in reverse...then, go back over it and you can read it all directly...

This example also proves just how "NOT phonetic" English is as a language...as all the ordinary Latin characters mostly have the sounds you'd expect, so it shouldn't really be so hard to read this file...but, give it a go and it's not as easy as it should...that's "non-phonetic" English for you (ah, here in Wales, it's easy to pronounce the Welsh words and names on the signs, once you master the "ll" and "ch" sounds because the language is entirely phonetic (actually, there is a single exception with the letter "y"...but it's easy to learn: It's "uh" – the obscure vowel – in any position but when it's the final "y" in the word, it sounds like "ee")...no idea what the words actually `_MEAN_`, mind you...but you can easily learn to pronounce them properly, anyway ;)...

Beth :)

.

• *Follow-Ups:*

Re: Fast UTF-8 strlen function

- ◆ **Re: Fast UTF-8 strlen function**
 - ◇ From: Annie
- ◆ **Re: Fast UTF-8 strlen function**
 - ◇ From: Sevag Krikorian

• **References:**

- ◆ **Fast UTF-8 strlen function**
 - ◇ From: randyhyde
- ◆ **Re: Fast UTF-8 strlen function**
 - ◇ From: Sevag Krikorian
- ◆ **Re: Fast UTF-8 strlen function**
 - ◇ From: randyhyde
- ◆ **Re: Fast UTF-8 strlen function**
 - ◇ From: Beth
- ◆ **Re: Fast UTF-8 strlen function**
 - ◇ From: Sevag Krikorian
- ◆ **Re: Fast UTF-8 strlen function**
 - ◇ From: NoDot
- ◆ **Re: Fast UTF-8 strlen function**
 - ◇ From: Sevag Krikorian
- ◆ **Re: Fast UTF-8 strlen function**
 - ◇ From: NoDot

- Prev by Date: **Re: create a file**
- Next by Date: **Re: create a file**
- Previous by thread: **Re: Fast UTF-8 strlen function**
- Next by thread: **Re: Fast UTF-8 strlen function**
- Index(es):
 - ◆ **Date**
 - ◆ **Thread**