Detecting what language or script a run of text is written in, redux

devblogs.microsoft.com/oldnewthing/20160829-00

August 29, 2016

Raymond Chen

Some time ago, I discussed the confusion surrounding the question, "How can I detect the language a run of text is in?" because the person asking the question was from an East Asian country, and in that part of the world, scripts and languages line up pretty closely. Chinese uses Hanzi, Korean uses Hangul, Japanese has a few scripts, Thai has its own alphabet, and so on. There is overlap, sure, but overall, you can tell what language a run of text is in without understanding anything about the language. You just have to see what font it's written in.

By comparison, the languages of Western Europe nearly all use the Latin alphabet. You need to know something about the languages themselves in order to distinguish French from Italian.

And then there are languages like Serbian and Chinese which have multiple writing systems. In Chinese, you can write in either Simplified or Traditional characters. In Serbian, you can choose between Latin or Cyrillic characters.

Extended Linguistic Services tries to address all three of these issues.[1]

- Language Detection guesses what language that segment might be written in, offering its results in decreasing order of confidence.
- Script Detection breaks a string into segments, each of which shares the same script.
- Transliteration converts text from one writing system to another.

I'm not going to write a Little Program to demonstrate this because there are already plenty of existing samples.

- The linguistic services sample on GitHub has wrapper functions in a single header file, offering you a one-stop-shopping experience. (But see remarks below.)
- MSDN has sample code for both the synchronous and asynchronous versions of the services.

When you adapt these samples into production code, note that MSDN recommends that you enumerate services only once, and then reuse the result, rather than enumerating each time you need the service.

(It appears to me that the Extended Linguistic Services was over-engineered. Enumeration seems unnecessary since there are only three services. Trying to force each service to use the same `MAPPING_PROPERTY_BAG` seems unnecessarily complicated. But what do I know. Maybe there's a method to their madness.)[2]

Instead of showing yet another sample, I'll just show the output of the services on various types of input. Note that language detection generally improves the longer the input, so these short snippets can generate lots of false positives.

Language detection	
Input	**Results**
That's Greek to me.	en, hr, sl, sr-Latin, da, es, et, fr, lv, nb, nn, pl, pt, sq, tn, yo
Das kommt mir spanisch vor.	de, gl, pt, ro
Αυτά μου φαίνονται κινέζικα.	el
Это для меня китайская грамота.	ru, be, uk
看起來像天書。	zh-Hant, zh

Script detection

In Greece, they say, “	Αυτά μου φαίνονται κινέζικα.”
Latn	Grek

ラドクリフ、マラソン	五輪代表	に	1	万	m	出場	にも	含	み
Kana	Hani	Hira				Hani	Hira		
		Hani↑		↑Latn			Hani↑		↑Hira

Observe that neutral characters (like the quotation mark in the first example and the digit 1 in the second example) get attached to the preceding script run.

Transliterator	Input	Output
Bengali to Latin	বাংলা	baaṁmlaa
Cyrillic to Latin	Кириллица	Kirillica
Devanagari to Latin	देवनागरी	devnaagrii
Mayalam to Latin	മലയാളം	mlyaaḷṁ
Simplified to Traditional Chinese	正体字	正體字
Traditional to Simplified Chinese	正體字	正体字

[1] Why “Extended” linguistic services instead of just plain “linguistic services”? Probably because that gave them a TLA.

[2] The method to their madness is that they anticipated building an entire empire of linguistic services, maybe even have multiple competing implementations, so your program could say, “You know, the Contoso script detector does a much better job than the Microsoft one. I’ll use that if available.” Except, of course, in practice, nobody writes script detectors except Microsoft.

Raymond Chen

Follow