

Re: Name parsing routine

Source:

<http://www.tech-archive.net/Archive/VB/microsoft.public.vb.general.discussion/2006-05/msg00499.html>

- *From:* John Hatpin <noone@xxxxxxxxxxxx>
 - *Date:* Thu, 04 May 2006 22:56:41 +0100
-

On Thu, 4 May 2006 15:27:37 -0400, "Matt Williamson" <ih8spam@xxxxxxxxxxxx> wrote:

I'm looking for some input on the best way to accomplish this. I have a data extract from Outlook that I'm parsing to go into a SQL table. I need to take a Full name and put it into 6 fields. PrimaryFirst, PrimaryMiddle, PrimaryLast, SecondaryFirst, SecondaryMiddle, SecondaryLast.

90% of the names fall into the following patterns:

FirstName Lastname
'Joe Anderson
FirstName MiddleName Lastname 'Joe
J. Anderson
Prefix FirstName MiddleName LastName 'Mr. Joe
J. Anderson
Prefix FirstName MiddleName LastName Suffix 'Mr. Joe
J. Anderson Jr.
FirstName and/& SecFirstName LastName 'Joe &
Jane Anderson

[etc]

This is one of those problems that can't really be solved with today's technology, especially when data entry has been at all slapdash in the past.

I've written stuff (not in VB) to attempt solutions for both names and addresses – in both cases, just like you say, you've got a large unformatted string (several for addresses) to deal with, and you need to split the data into more precise elements.

Addresses I'll leave aside – here in the UK, they can be cross-referenced against postal code data for validation, which helps a lot.

Re: Name parsing routine

Oh, and we did telephone numbers too, back when UK area codes all changed. That was easy in comparison, since you've got rules to work by, and you can pretty much guess that the letter O is an operator error for the number 0, I for 1, etc.

If it's any help, what the bosses wanted was a way to format free-entry name fields into title, first name, middle name, surname, suffix – just like you describe – and it proved to be impossible, especially trying to fit a single algorithm across the 250+ users' databases we had to convert.

The end solution was to make a best guess, to print out the before and after values in a structured report, and get the clients to distribute copies of the relevant parts of the report among their users for confirmation or written correction. Since the clients were firms of lawyers, and the users themselves lawyers (for whom time is money), this was an unpopular but necessary part of the solution.

Another consideration was case. Ideally, "Smith", "SMITH" and "smith" should all come out the other end as "Smith", but what about names like "DeGroot" (a Dutch surname), where the internal capital is very important?

The best I could come up with was about 40% valid names overall, very much depending on which clients had input their own clients' names under which system, fine-tuning for each client and **always** making sure the users could check the resultant data against the original, even if the self-checking algorithm reported a valid conversion.

You've got a tough job on – but that depends a lot on how consistently well the data's been entered in the first place.

I'll not even mention non-European naming conventions, where the distinction between forename and surname can get extremely confusing depending on the individual person's cultural heritage.

—

John Hatpin

.