Simple way to parse a person's name into its component parts? [closed]

Asked 16 years, 3 months ago Modified 4 years, 6 months ago

Viewed 35k times

32

Closed. This question needs to be more focused. It is not currently accepting answers.

Want to improve this question? Update the question so it focuses on one problem only by editing this post.

Closed 4 years ago.

Improve this question

A lot of contact management programs do this - you type in a name (*e.g.*, "John W. Smith") and it automatically breaks it up internally into:

First name: John
Middle name: W.
Last name: Smith

Likewise, it figures out things like "Mrs. Jane W. Smith" and "Dr. John Doe, Jr." correctly as well (assuming you allow for fields like "prefix" and "suffix" in names).

I assume this is a fairly common things that people would want to do... so the question is... how would you do it? Is there a *simple* algorithm for this? Maybe a regular expression?

I'm after a .NET solution, but I'm not picky.

Update: I appreciate that there is no simple solution for this that covers ALL edge cases and cultures... but let's say for the sake of argument that you need the name in pieces (filling out forms - as in, say, tax or other government forms - is one case where you are bound to enter the name into fixed fields, whether you like it or not), but you don't necessarily want to force the user to enter their name into discrete fields (less typing = easier for novice users).

You'd want to have the program "guess" (as best it can) on what's first, middle, last, etc. If you can, look at how Microsoft Outlook does this for contacts - it lets you type in the name, but if you need to clarify, there's an extra little window you can open. I'd do the same thing - give the user the window in case they want to enter the name in discrete pieces - but allow for entering the name in one box and doing a "best guess" that covers *most* common names.

string **parsing** **database-design**

Share

edited Oct 1, 2019 at 9:58

Improve this question

Follow

Raedwald

48.5k ● 47 ● 159 ● 245

asked Sep 19, 2008 at 16:23

Keithius

1,124 ● 2 ● 12 ● 20

1 This is one of those problems that seems simple, but becomes horribly complex with edge cases (people with: two middle names, hyphenated last names, prefixes like 'von', Asian names with family name first, etc.) – Chris Upchurch Sep 19, 2008 at 16:25

1 Don't forget the case where you have Madonna or Prince as contacts :) – Matt Howells Sep 19, 2008 at 16:33

Actually, single names are a bigger problem than certain self-obsessed pop stars. In some countries (Indonesia, for instance) a substantial portion of the population has only one name. – Chris Upchurch Sep 19, 2008 at 16:34

kalzumeus.com/2010/06/17/… – Ňɏssa Pøngjærdenlarp Sep 3, 2014 at 16:16

Here is a article by Microsoft that explains the logic of how to accomplish this: support.microsoft.com/kb/168799. It is probably what they use in Outlook. – Ron M Sep 25, 2014 at 13:41

23 Answers

Sorted by: Highest score (default)

35

If you *must* do this parsing, I'm sure you'll get lots of good suggestions here.

My suggestion is - **don't do this parsing**.

Instead, create your input fields so that the information is already separated out. Have separate fields for title, first name, middle initial, last name, suffix, etc.

Share Improve this answer

Follow

answered Sep 19, 2008 at 16:29

shadit

2,566 1 16 21

"...create your input fields so that the information is already separated out." That's true. I guess this is the Best Practice on this kind of stuff. This is also true with the "Address" – MarlonRibunal Sep 19, 2008 at 16:34

2 What about people with only one name? People whose first name is the name of formal address (Japan, China, Korea)? If nothing else, you'd better not address "Mausam" as "Dear Mausam [NULL]..." – Michael Ratanapintha Sep 19, 2008 at 16:35

Agreed. I took over a legacy app and I'm forced to parse the names because it's all stored together. It's a pain and I'm always having to add some extra logic to the routine to account for an outlier. User input varies, of course, so it's largely a crapshoot. – Mike L Sep 19, 2008 at 16:36

1 Good points. I guess depending upon the expected user community, it might be necessary to provide a single input field that still does not get parsed - just used as-is. – shadit Sep 19, 2008 at 16:37

1 Agreed. In additional to other comments you run the risk of **seriously** pissing people off because you are ignorant of the cultural issues on how their name is composed. – Simon Munro Sep 19, 2008 at 17:10

I know this is old and might be answers somewhere I couldn't find already, but since I couldn't find anything that works for me, this is what I came up with which I think works a lot like Google Contacts and Microsoft Outlook. It doesn't handle edge cases well, but for a good CRM type app, the user can always be asked to resolve those (in my app I actually have separate fields all the time, but I need this for data import from another app that only has one field):

```
    public static void ParseName(this string s,
out string prefix, out string first, out string
middle, out string last, out string suffix)
    {
        prefix = "";
        first = "";
        middle = "";
        last = "";
        suffix = "";

        // Split on period, commas or spaces, but
don't remove from results.
        List<string> parts = Regex.Split(s, @"(?<=
[., ])").ToList();

        // Remove any empty parts
        for (int x = parts.Count - 1; x >= 0; x--)
            if (parts[x].Trim() == "")
                parts.RemoveAt(x);

        if (parts.Count > 0)
        {
            // Might want to add more to this list
            string[] prefixes = { "mr", "mrs",
"ms", "dr", "miss", "sir", "madam", "mayor",
"president" };

            // If first part is a prefix, set
```

```
prefix and remove part
            string normalizedPart =
parts.First().Replace(".", "").Replace(",",
"").Trim().ToLower();
            if (prefixes.Contains(normalizedPart))
            {
                prefix = parts[0].Trim();
                parts.RemoveAt(0);
            }
```

Sorry that the code is long and ugly, I haven't gotten around to cleaning it up. It is a C# extension, so you would use it like:

```
string name = "Miss Jessica Dark-Angel Alba";
string prefix, first, middle, last, suffix;
name.ParseName(out prefix, out first, out middle,
out last, out suffix);
```

Share Improve this answer

Follow

answered Apr 26, 2013 at 17:20

eselk

6,884 ● 7 ● 62 ● 97

That's a little nicer than what I just trying to do. I found this blog post useful when determining which prefixes I need to support: notes.ericwillis.com/2009/11/… – Tim S Jul 14, 2014 at 20:42

2 Using a bunch of out parameters is NOT good. You should instead create a return type with the parts in that. – Kirk Sep 26, 2014 at 16:10

I like what you did here, and I took it and made a lot of changes to deal with things like multi-word last names (e.g. "van Winkle"), or multiple middle names (e.g. "John Nathan Edward dos Santos"). I'm posting it below. – PKD Dec 11, 2018 at 21:10

13

There is no simple solution for this. Name construction varies from culture to culture, and even in the English-speaking world there's prefixes and suffixes that aren't necessarily part of the name.

A basic approach is to look for honorifics at the beginning of the string (e.g., "Hon. John Doe") and numbers or some other strings at the end (e.g., "John Doe IV", "John Doe Jr."), but really all you can do is apply a set of heuristics and hope for the best.

It might be useful to find a list of unprocessed names and test your algorithm against it. I don't know that there's anything prepackaged out there, though.

Share Improve this answer

Follow

edited Oct 1, 2019 at 9:52

Raedwald

48.5k ● 47 ● 159 ● 245

answered Sep 19, 2008 at 16:31

Stephen Deken

3,695 ● 28 ● 31

Shad's answer is more correct when it comes to making actual systems. – Bob Cross Oct 29, 2008 at 21:30

Yeah, I hope you have a high tolerance for mistakes that it'll inevitably make. – OsamaBinLogin Jul 14, 2016 at 22:43

5

You probably don't need to do anything fancy really. Something like this should work.

```
    Name = Name.Trim();

    arrNames = Name.Split(' ');

    if (arrNames.Length > 0) {
        GivenName = arrNames[0];
    }
    if (arrNames.Length > 1) {
        FamilyName = arrNames[arrNames.Length -
1];
    }
    if (arrNames.Length > 2) {
        MiddleName = string.Join(" ", arrNames, 1,
arrNames.Length - 2);
    }
```

You may also want to check for titles first.

Share Improve this answer

Follow

edited Dec 9, 2011 at 4:44

StarCub

4,321 ● 7 ● 44 ● 58

answered Sep 19, 2008 at 16:34

Vincent McNabb

34.5k ● 7 ● 33 ● 54

If the design dictates that this parsing must take place, I agree this would be a good type of approach. – shadit Sep 19, 2008 at 16:42

There is a little problem with the code above. The check for length should be done using the arrNames array and not the Name variable. Also you could Trim and Split in one call. I

agree that for most scenarios this simple approach will work. – JCallico Nov 30, 2010 at 18:47

2 This won't work for names in the format of "McNabb, Vincent". – Chris Kerekes Jul 26, 2013 at 15:16

4

I had to do this. Actually, something much harder than this, because sometimes the "name" would be "Smith, John" or "Smith John" instead of "John Smith", or not a person's name at all but instead a name of a company. And it had to do it automatically with no opportunity for the user to correct it.

What I ended up doing was coming up with a finite list of patterns that the name could be in, like:
Last, First Middle-Initial
First Last
First Middle-Initial Last
Last, First Middle
First Middle Last
First Last

Throw in your Mr's, Jr's, there too. Let's say you end up with a dozen or so patterns.

My application had a dictionary of common first name, common last names (you can find these on the web), common titles, common suffixes (jr, sr, md) and using that would be able to make real good guesses about the patterns. I'm not that smart, my logic wasn't that fancy,

and yet still, it wasn't that hard to create some logic that guessed right more than 99% of the time.

Share Improve this answer

Follow

answered Sep 20, 2008 at 2:58

Corey Trager

23.1k ● 20 ● 87 ● 121

4 Nice to see some actual answers here, not "don't do it". Google (Contacts) and Microsoft (Outlook) both do it, for the exact same reasons the OP wants to. They have "set the standard", so doesn't matter if good design or not, the rest of us sheep must follow if we want to compete. Users want sleek/cool more than things that work 100% of the time. Blame the big guys, or the users, but don't just say "don't do it". – eselk Apr 26, 2013 at 15:14

4

Having come to this conversation 10 years late, but still looking for an elegant solution, I read through this thread, and decided to take the path @eselk took, but expand on it:

```
public class FullNameDTO
{
    public string Prefix     { get; set; }
    public string FirstName  { get; set; }
    public string MiddleName { get; set; }
    public string LastName   { get; set; }
    public string Suffix     { get; set; }
}

public static class FullName
{
    public static FullNameDTO
GetFullNameDto(string fullName)
    {
```

```
        string[] knownPrefixes    = { "mr", "mrs",
"ms", "miss", "dr", "sir", "madam", "master",
"fr", "rev", "atty", "hon", "prof", "pres", "vp",
"gov", "ofc" };
        string[] knownSuffixes    = { "jr", "sr",
"ii", "iii", "iv", "v", "esq", "cpa", "dc", "dds",
"vm", "jd", "md", "phd" };
        string[] lastNamePrefixes = { "da", "de",
"del", "dos", "el", "la", "st", "van", "von" };

        var prefix     = string.Empty;
        var firstName  = string.Empty;
        var middleName = string.Empty;
        var lastName   = string.Empty;
        var suffix     = string.Empty;

        var fullNameDto = new FullNameDTO
        {
            Prefix     = prefix,
            FirstName  = firstName,
            MiddleName = middleName,
            LastName   = lastName,
            Suffix     = suffix
```

This will handle quite a few different scenarios, and I've written out (thus far) over 50 different unit tests against it to make sure.

Props again to @eselk for his ideas that helped in my writing an expanded version of his excellent solution. And, as a bonus, this also handles the strange instance of a person named "JR".

Share Improve this answer edited Dec 28, 2018 at 19:53

Follow

answered Dec 11, 2018 at 21:16

PKD

707 ● 1 ● 13 ● 39

1 Even though I haven't actually gone through the whole method yet, so far this is looking really good. But I did have one quick question: Does this assume any particular order for the fullName variable (i.e., "first name first" or "last name first")? I'm guessing based on my initial observations that it *does* assume a "first name first" order. Also, this obviously does not take into account business names, but I'm thinking I might have a way to handle that... – G_Hosa_Phat Feb 7, 2019 at 20:40

1 @G_Hosa_Phat - It actually handles quite a few varieties of name. I have a metric ton of Unit Tests I wrote against it to test various patterns, but the TL:DR is "It handles Prefix First Middle Last Suffix, as well as Last Suffix, Prefix First Middle, and a whole lot more.". And for fun, the other day I ported it to SQL as a UDF, so I could have it for DB work. – PKD Feb 8, 2019 at 0:05

1 Some of the names I tested it with: "Dr. Indiana Johnathan Jones Jr.", "Dr. Jones Jr.", "Dr. Jones", "Indy", "Indiana Jones", "Jones Jr., Dr. Indiana Johnathan". – PKD Feb 8, 2019 at 0:06

Thanks for following up and clarifying. I've been going through the code and doing some conversion for VB and adding some logic for business names, but it seems to do a really good job so far. – G_Hosa_Phat Feb 8, 2019 at 18:09

"And for fun, the other day I ported it to SQL as a UDF, so I could have it for DB work."... Only on a site like this does such a statement NOT sound strange. :P – G_Hosa_Phat Feb 8, 2019 at 18:09

3

I appreciate that this is hard to do *right* - but if you provide the user a way to edit the results (say, a pop-up window to edit the name if it didn't guess right) and still guess "right" for most cases... of course it's the guessing that's tough.

It's easy to say "don't do it" when looking at the problem theoretically, but sometimes circumstances dictate otherwise. Having fields for all the parts of a name (title, first, middle, last, suffix, just to name a few) can take up a lot of screen real estate - and combined with the problem of the address (a topic for another day) can really clutter up what *should* be a clean, simple UI.

I guess the answer should be "don't do it unless you absolutely have to, and if you do, keep it simple (some methods for this have been posted here) and provide the user the means to edit the results if needed."

Share Improve this answer

Follow

answered Sep 19, 2008 at 18:42

Keithius

1,124 ● 2 ● 12 ● 20

If your are going to be displaying the results of your guess on the screen why not forgo the guessing and let the user enter their own text into the fields? Displaying the guesses introduces as much clutter as skipping the guesswork.
– Frosty Sep 19, 2008 at 19:23

Not if it's a separate window :-) – Keithius Sep 19, 2008 at 19:39

I agree with you.. "Don't do it" isn't a very helpful answer. This is definitely not easy and is never going to be perfect, but I think with a little effort you can provide a better user experience doing something like this. Also, as long as you give a user a chance to review/edit the fields on a different screen if needed than this is fine. – delux247 Sep 6, 2010 at 19:55

3

Understanding this is a bad idea, I wrote this regex in perl - here's what worked the best for me. I had already filtered out company names.
Output in vcard format: (hon_prefix, given_name, additional_name, family_name, hon. suffix)

```
/^ \s*
    (?:((?:Dr.)|(?:Mr.)|(?:Mr?s.)|(?:Miss)|
(?:2nd\sLt.)|(?:Sen\.?))\s+)? # prefix
    ((?:\w+)|(?:\w\.)) # first name
(?: \s+ ((?:\w\.?)|(?:\w\w+)) )?  # middle initial
(?: \s+ ((?:[OD]['’]\s?)?[-\w]+))    # last name
(?: ,? \s+ ( (?:[JS]r\.?) | (?:Esq\.?) | (?:
(?:M)|(?:Ph)|(?:Ed) \.?\s*D\.?) |
         (?: R\.?N\.?) | (?: I+) )  )? # suffix
\s* $/x
```

notes:

- doesn't handle IV, V, VI
- Hard-coded lists of prefixes, suffixes. evolved from dataset of ~2K names
- Doesn't handle multiple suffixes (eg. MD, PhD)

- Designed for American names - will not work properly on romanized Japanese names or other naming systems

Share Improve this answer

Follow

answered Dec 26, 2008 at 21:43

Thelema

14.7k ● 6 ● 29 ● 35

Could you please tell me how to add suffix support. American name support is all I am looking for. An algorithm for actual implementation will be awesome. Thank you.. – ThinkCode Mar 29, 2010 at 15:21

3

The real solution here does not answer the question. The portent of the information must be observed. A name is not just a name; it is how we are known.

The problem here is not knowing exactly what parts are labled what, and what they are used for. Honorable prefixes should be granted only in personal corrospondences; Doctor is an honorific that is derived from a title. All information about a person is relavent to their identity, it is just determining what is relavent information. You need a first and last name for reasons of administration; phone number, email addresses, land descriptions and mailing addresses; all to the portent of identity, knowing who you are dealing with.

The real problem here is that the person gets lost in the administration. All of a sudden, after only entering their personal information into a form and submitting it to an

arbitrary program for processing, they become afforded all sorts of honorifics and pleasentries spewed out by a prefabricated template. This is wrong; honorable Sir or Madam, if personal interest is shown toward the reason of corrospondence, then a letter should never be written from a template. Personal corrospondance requires a little knowledge about the recipient. Male or female, went to school to be a doctor or judge, what culture in which they were raised.

In other cultures, a name is made up from a variable number of characters. The person's name to us can only be interpretted as a string of numbers where the spaces are actually determined by character width instead of the space character. Honorifics in these cases are instead one or many characters prefixing and suffixing the actual name. The polite thing to do is use the string you are given, if you know the honorific then by all means use it, but this again implies some sort of personal knowledge of the recipient. Calling Sensei anything other than Sensei is wrong. Not in the sense of a logic error, but in that you have just insulted your caller, and now you should find a template that helps you apologize.

For the purposes of automated, impersonal corrospondence, a template may be devised for such things as daily articles, weekly issues or whatever, but the problem becomes important when corrospondence is instigated by the recipient to an automated service.

What happens is an error. Missing information. Unknown or missing information will always generate an Exception. The real problem is not how do you seperate a person's name into its seperate components with an expression, but what do you call them.

The solution is to create an extra field, make it optional if there is already a first and last name, and call it "What may we call you" or "How should we refer to you as". A doctor and a judge will ensure you address them properly. These are not programming issues, they are issues of communication.

Ok, bad way to put it, but in my opinion, Username, Tagname, and ID are worse. So my solution; is the missing question, "What should we call you?"

This is only a solution where you can afford to make a new question. Tact prevails. Create a new field upon your user forms, call it Alias, label for the user "What should we call you?", then you have a means to communicate with. Use the first and last name unless the recipient has given an Alias, or is personally familiar with the sender then first and middle is acceptable.

```
To Me, _____________________ (standard
subscribed corrospondence)
To Me ( Myself | I ), _________ (standard recipient
instigated corrospondence)
To Me Myself I, _______________ (look out, its your
mother, and you're in big trouble;
                                 nobody addresses a
person by their actual full name)
```

```
Dear *(Mr./Mrs./Ms./Dr./Hon./Sen.) Me M. I *(I),
To Whom it may Concern;
```

Otherwise you are looking for something standard: hello, greetings, you may be a winner.

Where you have data that is a person's name all in one string, you don't have a problem because you already have their alias. If what you need is the first and last name, then just Left(name,instr(name," ")) & " " & Right(name,instrrev(name," ")), my math is probably wrong, i'm a bit out of practice. compare left and right with known prefixes and suffixes and eliminate them from your matches. Generally the middle name is rarely used except for instances of confirming an identity; which an address or phone number tells you a lot more. Watching for hyphanation, one can determine that if the last name is not used, then one of the middle ones would be instead.

For searching lists of first and last names, one must consider the possibility that one of the middle ones was instead used; this would require four searches: one to filter for first & last, then another to filter first & middle, then another to filter middle & last, and then another to filter middle & middle. Ultimately, the first name is always first, and the last is always last, and there can be any number of middle names; less is more, and where zero is likely, but improbable.

Sometimes people prefer to be called Bill, Harry, Jim, Bob, Doug, Beth, Sue, or Madonna; than their actual

names; similar, but unrealistically expected of anyone to fathom all the different possibilities.

The most polite thing you could do, is ask; What can we call you?

Share Improve this answer Follow

edited Mar 2, 2012 at 15:41

answered Mar 1, 2012 at 0:11

Aeryes
31 ● 2

I would like to point out that this thread is a little old and may not get reviewed; however, as there are many viable and good solutions on this page, it will still help others later. To me, the consensus seems to be, less is more; don't do this parsing. – Aeryes Mar 2, 2012 at 15:15

2

There are a few add-ins we have used in our company to accomplish this. I ended up creating a way to actually specify the formats for the name on our different imports for different clients. There is a company that has a tool that in my experience is well worth the price and is really incredible when tackling this subject. It's at: http://www.softwarecompany.com/ and works great. The most efficient way to do this w/out using any statistical approach is to split the string by commas or spaces then: 1. strip titles and prefixes out 2. strip suffixes out 3, parse

name in the order of (2 names = F & L, 3 names = F M L or L M F) depending on order of string().

Share Improve this answer

Follow

answered Jan 25, 2010 at 20:20

Sean Fair

21 ● 1

1

You can do the obvious things: look for Jr., II, III, etc. as suffixes, and Mr., Mrs., Dr., etc. as prefixes and remove them, then first word is first name, last word is last name, everything in between are middle names. Other than that, there's no foolproof solution for this.

A perfect example is David Lee Roth (last name: Roth) and Eddie Van Halen (last name: Van Halen). If Ann Marie Smith's first name is "Ann Marie", there's no way to distinguish that from Ann having a middle name of Marie.

Share Improve this answer

Follow

answered Sep 19, 2008 at 16:31

Graeme Perrow

57.2k ● 24 ● 86 ● 125

1

If you simply have to do this, add the guesses to the UI as an optional selection. This way, you could tell the user how you parsed the name and let them pick a different parsing from a list you provide.

Share Improve this answer

Follow

answered Sep 20, 2008 at 22:49

Omer van Kloeten

12k ● 9 ● 44 ● 54

There is a 3rd party tool for this kind of thing called NetGender that works surprisingly well. I used it to parse a massive amount of really mal-formed names in unpredictable formats. Take a look at the examples on their page, and you can download and try it as well.

http://www.softwarecompany.com/dotnet/netgender.htm

I came up with these statistics based on a sampling of 4.2 million names. Name Parts means the number of distinct parts separated by spaces. A very high percentage were correct for most names in the database. The correctness went down as the parts went up, but there were very few names with >3 parts and fewer with >4. This was good enough for our case. Where the software fell down was recognizing not-well-known multi-part last names, even when separated by a comma. If it was able to decipher this, then the number of mistakes in total would have been less than 1% for all data.

```
Name Parts | Correct | Percent of Names in DB
 2             100%       48%
 3              98%       42%
 4              70%        9%
 5              45%        0.25%
```

Share Improve this answer edited Sep 8, 2011 at 20:06

Follow

answered Sep 1, 2011 at 20:31

Micah B.
1,156 ● 4 ● 13 ● 28

1

I already do this server-side on page load. Wrote a Coldfusion CFC that gets two params passed to it - actual user data(first name, middle, last name) and data type(first,middle,last). Then routine checks for hyphens, apostrophes, spaces and formats accordingly. ex. MacDonald, McMurray, O'Neill, Rodham-Clinton, Eric von Dutch, G. W. Bush, Jack Burton Jr., Dr. Paul Okin, Chris di Santos. For case where users only have one name, only the first name field is required, middle and last names are optional.

All info is stored lowercase - except Prefix, Suffix and Custom. This formatting is done on page render, not during store to db. Though there is validation filtering when user inputs data. Sorry, cannot post code. Started out using Regex but became too confusing and unreliable for all scenarios. Used standard logic blocks(if/else, switch/case), easier to read and debug. MAKE EVERY INPUT/DB FIELD SEPARATE! Yes, this will take some coding, but after you are finished it should account for 99% of combinations. Only based on English names so far, no internationalization, that's another ball of wax.

Here's some things to consider:

- Hypens (ex. Rodham-Clinton, could be in first, middle or last)

- Apostrophes (ex. O'Neill, could be in first, middle or last)
- Spaces
- Mc and Mac (ex. McDonald, MacMurray, could be in first, middle or
  last)
- First names: multiple first names (ex. Joe Bob Briggs)
- Last names: de,di,et,der,den,van,von,af should be lowercase (ex Eric von Dander, Mary di Carlo)
- Prefix: Dr., Prof., etc
- Suffix: Jr., Sr., Esq., II, III, etc

When user enters info, field schema in db is like so:

- Prefix/Title (Dr., etc using a dropdown)
- Prefix/Title Custom (user can enter custom, ex. Capt. using a text field)
- First Name
- Middle
- Last Name
- Suffix (Jr., III, Prof., Ret., etc using a dropdown)
- Suffix Custom (user can enter custom, ex. CPA)

Here's the one Regex I do use to make first letter of each name uppercase. I run this first, then following routines

format according to rules(it's in Coldfusion format but you get the idea):

```
<cfset var nameString =
REReplace(LCase(nameString), "(^[[:alpha:]]|
[[:blank:]][[:alpha:]])", "\U\1\E", "ALL")>
```

You could also do this client-side using JavaScript and CSS - might even be easier - but I prefer to do server-side since I need the variables set before page loads client-side.

Share Improve this answer

Follow

edited Nov 16, 2011 at 7:44

Nightfirecat

11.6k ● 6 ● 37 ● 53

answered Nov 16, 2011 at 7:11

user1049064

11 ● 2

0

I would say Strip out salutations from a list then split by space, placing list.first() as first name, list.last() as last name then join the remainder by a space and have that as a middle name. And ABOVE ALL display your results and let the user modify them!

Share Improve this answer

Follow

answered Sep 19, 2008 at 16:31

George Mauer

122k ● 139 ● 395 ● 626

0

Sure, there is a simple solution - split the string by spaces, count the number of tokens, if there is 2, interpret them to be FIRST and LAST name, if there is 3, interpret it to be FIRST, MIDDLE, and LAST.

The problem is that the simple solution will not be a 100% correct solution - someone could always enter a name with many more tokens, or could include titles, last names with a space in it (is this possible?), etc. You can come up with a solution that works for most names most of the time, but not an absolute solution.

I would follow Shad's recommendation to split the input fields.

Share Improve this answer

Follow

answered Sep 19, 2008 at 16:33

matt b

140k ● 66 ● 284 ● 350

As you point out, that approach would fail for Mr. T. I pity the fool who parses his name wrong! – shadit Sep 19, 2008 at 18:37

0

You don't want to do this, unless you are only going to be contacting people from one culture.

For example:

Guido van Rossum's last name is van Rossum.

MIYAZAKI Hayao's first name is Hayao.

The most success you could do is to strip off common titles and salutations, and try some heuristics.

Even so, the easiest solution is to just store the full name, or ask for given and family name seperately.

Share Improve this answer

Follow

answered Sep 19, 2008 at 16:35

1729

5,031 ● 4 ● 29 ● 17

0

This is a fools errand. Too many exceptions to be able to do this deterministically. If you were doing this to pre-process a list for further review I would contend that less would certainly be more.

1. Strip out salutations, titles and generational suffixes (big regex, or several small ones)
2. if only one name, it is 'last'.
3. If only two names split them first,last.
4. If three tokens and middle is initial split them first, middle, last
5. Sort the rest by hand.

Any further processing is almost guaranteed to create more work as you have to go through recombining what your processing split-up.

Share Improve this answer

Follow

answered Sep 19, 2008 at 16:53

Frosty
6,443 • 3 • 26 • 20

0

I agree, there's no simple solution for this. But I found an awful approach in a Microsoft KB article for VB 5.0 that is an actual implementation to much of the discussion talked about here: http://support.microsoft.com/kb/168799

Something like this could be used in a pinch.

Share Improve this answer

Follow

answered Dec 2, 2008 at 20:39

Shawn Miller
7,080 • 6 • 47 • 54

0

There is no 100% way to do this.

You can split on spaces, and try to understand the name all you want, but when it comes down to it, you will get it wrong sometimes. If that is good enough, go for any of the answers here that give you ways to split.

But some people will have a name like "John Wayne Olson", where "John Wayne" is the first name, and someone else will have a name like "John Wayne Olson" where "Wayne" is their middle name. There is nothing present in that name that will tell you which way to interpret it.

That's just the way it is. It's an analogue world.

My rules are pretty simple.

Take the last part --> Last Name
If there are multiple parts left, take the last part --> Middle name
What is left --> First name

But don't assume this will be 100% accurate, nor will any other hardcoded solution. You will need to have the ability to let the user edit this him/her-self.

Share Improve this answer

Follow

answered Dec 26, 2008 at 22:34

Lasse V. Karlsen
391k ● 106 ● 646 ● 844

0

I did something similar. The main problem I had was when people entered stuff like "Richard R. Smith, Jr." I posted my code at http://www.blackbeltcoder.com/Articles/strings/splitting-a-name-into-first-and-last-names. It's in C# but could easily be converted to VB.

Share Improve this answer

Follow

answered Jan 28, 2011 at 22:21

Jonathan Wood
67k ● 80 ● 299 ● 523

0

A simpler way:

Install `HumanNameParser` NuGet:

```
Install-Package HumanNameParser
```

And call Parse extension method.

```
string name = "Mr Ali R Von Bayat III";

var result = name.Parse();

//result = new Name()
//{
//    Salutation = "Mr",
//    FirstName = "Ali",
//    MiddleInitials = "R",
//    LastName = "Von Bayat",
//    Suffix = "III"
//};
```

Share Improve this answer

Follow

answered Jun 17, 2020 at 3:41

Ali Bayat

4,026 ● 3 ● 47 ● 43

-2

I agree with **not to do this** . The name Rick Van DenBoer would end up with a middle name of Van but it's part of the last name.

Share Improve this answer

Follow

answered Sep 19, 2008 at 16:32

osp70

1,102 ● 1 ● 12 ● 20