

# *Net* **Gender** 5.1

## Name Parsing and Gender Identification API User's Guide

The Software Company, Inc.  
<https://SoftwareCompany.com>

**The Software Company**<sup>TM</sup>  
Software is our middle name

**NetGender for .NET** allows you to quickly and easily build name verification, parsing and gender determination into your custom applications. Accurately verify whether or not a particular field contains a valid individual or company name.

Accept names free-form and let NetGender automatically split each name into its standard components: Prefix, First, Middle, Last and Suffix no matter what the original format. Using a 200,000+ first and last name dictionary in combination with a 10,000+ term company dictionary, the gender is determined with unmatched precision. Easily updated tables control the entire process.

Also included in the dictionary are extensive name variants to help identify obscure duplicate names such as “Bill”, “Billy”, “Will”, “Willy”, “William”, “Wm”, etc.

**NetGender for .NET** can process all styles of names including inverse, natural order, hyphenated and multi-part last names. Multiple names in the same field and companies can be easily separated providing you with powerful formatting control.

### Benefits

- **200,000+ First and Last Name Dictionary with Variants** – for pinpoint accuracy
- **Quickly Identify Incomplete or Incorrect Names** – before they enter your database
- **Automatic Name Style Identification** – standardize lists of various formats
- **Name Variants Property** - invaluable for finding common duplicates
- **Unlimited Processing Volume** – no recurring update charges

### Features

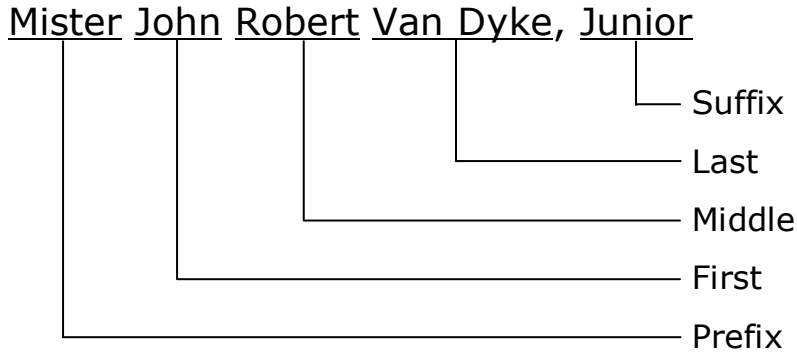
- Accurately separate name elements: Prefix, First, Middle, Last, Suffix
- Proper-case conversion for more attractive data presentation
- User-updatable tables so your applications never go out-of-date
- Royalty-free runtime
- Designed for use with all .NET-compatible programming languages

**NetGender for .NET** starts by meticulously identifying each individual name element based on the "Name\_Style" selected. Intuitive algorithms examine the results then a selection is made of the most complete and correct data. Next, the name elements are split according to "Name\_Style" and the user-specified prefix/suffix abbreviations are applied. If not already coded by the gender-override table, the gender is now determined according to the 200,000+ name system gender table. The "Name\_Quality" flag is then set to indicate how complete and correct the name appears. Finally, the standardized name components are returned to your application along with a complete and cleansed composite name.

### Examples using "AutoDetect" name style:

<b>INPUT NAME:</b>		<b>DE LA ROSA, JUAN R OR MARIA</b>
NAME 1:	Male	Juan R De La Rosa
NAME 2:	Female	Maria De La Rosa
<b>INPUT NAME:</b>		<b>LA BELLA, JOHN A &amp; MARY B, PHD</b>
NAME 1:	Male	John A La Bella
NAME 2:	Female	Mary B La Bella PhD
<b>INPUT NAME:</b>		<b>MARY MYTH C/O THE SOFTWARE COMPANY</b>
NAME 1:	Female	Mary Myth
NAME 2:	Company	The Software Company
<b>INPUT NAME:</b>		<b>VAN ZANT, JUDY &amp; RONNIE</b>
NAME 1:	Neutral	Judy Van Zant
NAME 2:	Male	Ronnie Van Zant
<b>INPUT NAME:</b>		<b>AFTON JONES &amp; WIFE JOYA</b>
NAME 1:	Male	Afton Jones
NAME 2:	Female	Joya Jones

**Name Elements**



---

**Gender Coding**

NetGender uses an extensive dictionary of more than 200,000 first and last names that were specially selected for their rich ethnic-diversity.

NetGender also uses a unique gender percentage factor. This factor is based on the proportion of males to females for a particular name allowing you to set the point at which certain names will be returned with a neutral gender. See *“Gender\_Confidence”* and *“Gender\_Percentage”* properties.

NetGender then applies the rules from a 10,000+ term company dictionary resulting in a high level of gender accuracy.

### Name\_In

**Syntax:** Name\_In = String

**Description:**

Set this property to the name string to be processed. When the “Parse” method is invoked, the “Name\_In” string is standardized and formatted then placed into the “Name\_Out” property. In addition, each element of the “Name\_In” string is placed into the corresponding name component property.

### Name\_Style

**Syntax:** Name\_Style = StringLiteral

**Description:**

Set this property to the style that most closely matches the format of the “Name\_In” string. In the examples below, F = First, M = Middle, L = Last, & = generic delimiter, ( ) = optional element.

AutoDetect	Mixed Format or unknown	Names are mixed natural & inverse
FML	John A Doe	Names are in Natural Order
FML&FM	John A Doe & Jane A	Never a Last Name in Name 2
FML&FM(L)	John A Doe & Jane A (Jones)	Sometimes a Last Name in Name 2
FML&FML	John A Doe & Jane A Jones	Always a Last Name in Name 1 & 2
FM(L)&FML	John A (Doe) & Jane A Jones	Sometimes a Last Name in Name 1
FM(L)&FM(L)	John A (Doe) & Jane A (Jones)	Sometimes a Last Name in Name 1 or 2
FM&FML	John A & Jane A Doe	Never a Last Name in Name 1
LFM	Doe, John A	Names are in Inverse Order
LFM&FM	Doe, John A & Jane A	Never a Last Name in Name 2
LFM&FM(L)	Doe, John A & Jane A (Jones)	Sometimes a Last Name in Name 2
LFM&(L)FM	Doe, John A & (Jones)Jane A	Sometimes a Last Name in Name 2
LFM&LFM	Doe, John A & Jones Jane A	Always a Last Name in Name 1 & 2
Split	Anything & everything	Split at Name Connector only
Company	Sanford & Son	Forces a company name check without splitting at the name connector

The input names can be separated by various delimiters such as: “c/o”, “&”, “dba”, “and”, “attn”, etc. The “&” above, generically represents these delimiters. The delimiters are user-defined. See “Updating User Control Tables” later in this guide. Parenthesis “( )” indicate that this element may or may not be present in all of the input name strings. If two individuals or a company and an individual co-exist in the input name string, you should choose one of the multi-name styles such as: “FML&FML”, “FML&FM”, etc. for best results.

*Best results are obtained using the closest matching style. Use “AutoDetect” when the name format is inconsistent or unknown.*

**Name Style Examples - Natural Order**

- FML**      **Use when there is only one name in the input string.**  
Name is split into First, Middle, Last:  
    **Name\_In:**     JOHN A DOE  
    **Name\_Out:**    John A Doe
- FML&FM**   **Use when last name is *never* present in name two.**  
Name two ALWAYS inherits its last name from Name one:  
    **Name\_In:**     JANE A DOE & JOHN A  
    **Name\_Out:**    Jane A Doe  
    **Name2\_Out:**  John A Doe
- FML&FM(L)** **Use when last name is *sometimes* present in name two.**  
Name two keeps its own last name if present:  
    **Name\_In:**     JOHN A DOE & JANE A JONES  
    **Name\_Out:**    John A Doe  
    **Name2\_Out:**  Jane A Jones  
Name two inherits its last name from Name one:  
    **Name\_In:**     JOHN A DOE & JANE A  
    **Name\_Out:**    John A Doe  
    **Name2\_Out:**  Jane A Doe
- FML&FML**   **Use when a full name is *always* present in name one & two.**  
Name one & two ALWAYS keep their own last names:  
    **Name\_In:**     JOHN DOE DBA THE SOFTWARE COMPANY  
    **Name\_Out:**    John Doe  
    **Name2\_Out:**  The Software Company
- FM(L)&FML** **Use when last name is *sometimes* present in name one.**  
Name one keeps its own last name if present:  
    **Name\_In:**     JOHN A DOE & JANE A JONES  
    **Name\_Out:**    John A Doe  
    **Name2\_Out:**  Jane A Jones  
Name one inherits its last name from Name two:  
    **Name\_In:**     JOHN A DOE & JANE A  
    **Name\_Out:**    John A Doe  
    **Name2\_Out:**  Jane A Doe

**Name Style Examples - Natural Order (cont.)**

**FM(L)&FM(L)** Use when last name is *sometimes* present in name one or two.

Name one and two keep their own last names if present:

**Name\_In:** JOHN A DOE & JANE A JONES

**Name\_Out:** John A Doe

**Name2\_Out:** Jane A Jones

Name one inherits its last name from Name two:

**Name\_In:** JOHN A & JANE A JONES

**Name\_Out:** John A Jones

**Name2\_Out:** Jane A Jones

**FM&FML** Use when last name is *never* present in name one.

Name one ALWAYS inherits its last name from Name two:

**Name\_In:** JANE A & JOHN DOE

**Name\_Out:** Jane A Doe

**Name2\_Out:** John Doe

**Name Style Examples - Inverse Order**

- LFM**            **Use when there is only one name in the input string.**  
Name is split into First, Middle, Last:  
    **Name\_In:**     DOE, JOHN A  
    **Name\_Out:**    John A Doe
- LFM&FM**       **Use when last name is *never* present in name two.**  
Name two ALWAYS inherits its last name from Name one:  
    **Name\_In:**     DOE, JANE A & JOHN A  
    **Name\_Out:**    Jane A Doe  
    **Name2\_Out:** John A Doe
- LFM&FM(L)**   **Use when last name is *sometimes* present in name two.**  
Name two keeps its own last name if present:  
    **Name\_In:**     DOE, JOHN A & JANE A JONES  
    **Name\_Out:**    John A Doe  
    **Name2\_Out:** Jane A Jones  
Name two inherits its last name from Name one:  
    **Name\_In:**     DOE, JOHN A & JANE A  
    **Name\_Out:**    John A Doe  
    **Name2\_Out:** Jane A Doe
- LFM&(L)FM**   **Use when last name is *sometimes* present in name two.**  
Name two keeps its own last name if present:  
    **Name\_In:**     DOE, JOHN A & JONES, JANE A  
    **Name\_Out:**    John A Doe  
    **Name2\_Out:** Jane A Jones  
Name two inherits its last name from Name one:  
    **Name\_In:**     DOE, JOHN A & JANE A  
    **Name\_Out:**    John A Doe  
    **Name2\_Out:** Jane A Doe
- LFM&LFM**     **Use when a full name is *always* present in name one & two.**  
Name one & two ALWAYS keep their own last names:  
    **Name\_In:**     DOE, JOHN DBA THE SOFTWARE COMPANY  
    **Name\_Out:**    John Doe  
    **Name2\_Out:** The Software Company

**Name Style Examples - Other****AutoDetect**

Use when the format of the name(s) is inconsistent or unknown.

Style is automatically determined based on the format, punctuation and gender lookup results. Name one is FML, Name two is LFM:

**Name\_In:** JOHN A DOE & JONES, JANE A

**Name\_Out:** John A Doe

**Name2\_Out:** Jane A Jones

**Split**

Use when you want to split name one & two at the first [NameConnector]

Filters are removed, no other formatting takes place.

[NameConnector] of “c/o” was specified in the “NetGender.ref” file:

**Name\_In:** DOE JOHN A c/o THE SOFTWARE COMPANY

**Name\_Out:** Doe John A

**Name2\_Out:** The Software Company

**Company**

Use when company names also contain name connectors.

Use “COMPANY” style to avoid confusion with company names that also contain Name Connectors. When [NameConnectors] such as “&” are specified in the user-defined “NetGender.ref” file, certain company names such as “Dewey Cheatem & Howe” can sometimes be confused with a compound individual name such as “John & Mary Smith”. If this is the case, use the “COMPANY” style first to identify the field as a company.

If an exact match on a company name is found in the “NetGender.ref” file or certain company keywords such as “ABC” are found in the system gender table, the Gender property is returned as “C” (company) and the “Name\_Quality” property is returned as “High”. A “Name\_Quality” of “Medium” is returned when certain keywords such as “CLUB” are found in the system gender table indicating that this could possibly be either a company or individual.

### Company\_Check

**Syntax:** Company\_Check = Boolean (True/False)

**Description:**

Set this property to Boolean (True/False) to indicate if an attempt should be made to identify the "Name\_In" string as a company. Set "Company\_Check" to "False" if your data has no company names. **Default is "False"**.

---

### Name\_Variant\_Check

**Syntax:** Name\_Variant\_Check = Boolean (True/False)

**Description:**

Set this property to Boolean (True/False) to indicate if an attempt should be made to identify common name variants associated with "Name" and "Name2". Set "Name\_Variant\_Check" property to "True" if you want name variants returned in "Name\_Variants" and "Name2\_Variants" properties. **Default is "False"**.

---

### Parse\_Nicknames

**Syntax:** Parse\_Nicknames = Boolean (True/False)

**Description:**

Set this property to Boolean (True/False) to indicate whether or not to parse nicknames from the "Name\_In" property. A nickname is identified as being enclosed in single or double quotes or parenthesis. Set "Parse\_Nicknames" to "True" if you want nicknames returned in "Nickname" and "Nickname2" properties. **Default is "False"**.

### Gender\_Confidence

**Syntax:** Gender\_Confidence = Integer

**Description:**

Set this property to a value between 51% and 100% representing the cutoff point below which a gender is considered neutral. Each name in the System Gender Table is encoded with a percentage from 51% to 100% based on the proportion of males to females for that particular name. After the name lookup, if a percentage is found that is below the “Gender\_Confidence” level, NetGender will return a gender of Neutral for that name. A setting of 51% will force a gender of either Male or Female. **Default is 70%.**

**Neutral Gender Breakdown:**

<u>Gender_Confidence</u>	<u># of Neutral</u>	<u>% of Gender Table</u>	
100%	7,800	7.8%	
95%	6,400	6.4%	
90%	5,100	5.1%	
80%	3,900	3.9%	
70%	2,400	2.4%	Default Setting
60%	900	.9%	
51%	0	.0%	

---

### Capitalization

**Syntax:** Capitalization = StringLiteral

**Description:**

Set this property to “Upper”, “Lower”, “Mixed” or “None” to indicate your capitalization preference for the output name and its components. Use “None” when you want to preserve the existing capitalization. **Default is “None”.**

### Reference\_File\_Path, Gender\_File\_Path

**Syntax:**      Reference\_File\_Path = String  
                 Gender\_File\_Path = String

**Description:**

Set this property to the full path and file name of the following system and user files:

“**NetGender.ref**” is a user-defined file containing the tables for Prefix, Suffix, Filter and Connector identification as well as Gender and Name Variant Overrides.

“**NetGender.gnd**” is a system file that contains the main Genderization control tables.

A standard set of these files is supplied and installed in the NetGender installation folder under the names: “NetGender.ref” and “NetGender.gnd”. You can rename and relocate these files to any other folder as long as you provide the full path and file name information in each respective path string. **Default path is first the folder of the invoking application: “AppDomain.CurrentDomain.BaseDirectory” then the NetGender installation folder.**

*See “Updating User Control Tables” later in this guide for information on customizing this file.*

---

### Static\_Key\_Name (licensed version)

**Syntax:**      Static\_Key\_Name = String

**Description:**

Set this property to the name portion of the static key assignment or blank.

---

### Static\_Key (licensed version)

**Syntax:**      Static\_Key = String

**Description:**

Set this property to the key portion of the static key assignment or blank.

### **Name\_Out, Name2\_Out** (read only)

**Syntax:**     String = Name\_Out  
              String = Name2\_Out

**Description:**

After invoking the “Parse” method, this property will contain the corrected full name string, in natural order (FML), from the “Name\_In” property.

---

### **Name\_Quality, Name2\_Quality** (read only)

**Syntax:**     String = Name\_Quality  
              String = Name2\_Quality

**Description:**

After invoking the “Parse” method, this property is set to “Low”, “Medium” or “High” to indicate how complete a name appears.

---

### **Prefix, Prefix2** (read only)

**Syntax:**     String = Prefix  
              String = Prefix2

**Description:**

After invoking the “Parse” method, this property is set to the name “Prefix” component of “Name\_In”. Values will be valid prefixes (Mr, Mrs, etc.) or blank.

---

### **First, First2** (read only)

**Syntax:**     String = First  
              String = First2

**Description:**

After invoking the “Parse” method, this property is set to the First Name component of “Name\_In”. Value will be an individual’s first name or blank.

### **Middle, Middle2** (read only)

**Syntax:**     String = Middle  
              String = Middle 2

**Description:**

After invoking the “Parse” method, this property is set to the Middle Name component of “Name\_In”. Value will be an individual’s middle name or blank.

---

### **Last, Last2** (read only)

**Syntax:**     String = Last  
              String = Last2

**Description:**

After invoking the “Parse” method, this property is set to the Last Name component of “Name\_In”. Value will be an individual’s last name or blank.

---

### **Suffix, Suffix2** (read only)

**Syntax:**     String = Suffix  
              String = Suffix2

**Description:**

After invoking the “Parse” method, this property is set to the Suffix component of “Name\_In”. Values will be valid suffixes (Jr, Sr, Esq, etc.) or blank.

---

### **Company, Company2** (read only)

**Syntax:**     String = Company  
              String = Company2

**Description:**

After invoking the “Parse” method, this property is set to the Company component of “Name\_In”. Values will be a company name or blank.

---

### **Gender, Gender2** (read only)

**Syntax:**     String = Gender  
              String = Gender2

**Description:**

After invoking the “Parse” method, this property is set to the Gender indicated by the “First” property. Values will be “M” (male), “F” (female), “N” (neutral), “C” (company) or “U” (unknown). See “Gender\_Confidence” property.

### **Gender\_Percentage, Gender2\_Percentage** (read only)

**Syntax:** Integer = Gender\_Percentage  
Integer = Gender2\_Percentage

**Description:**

After invoking the “Parse” method, this property is set to the statistical gender percentage. Each name in the System Gender Table is encoded with a percentage from 51% to 100% based on the proportion of males to females for that particular name. For example, the name “Chris” returns a gender of “M” and a “Gender\_Percentage” of 89%. This indicates that 89% of the persons with the name “Chris” are statistically male and 11% are female. *See “Gender\_Confidence” and “Gender” properties.*

**Note:** This property is only valid when the Gender property returns a value of “M” or “F”.

---

### **Name\_Variants, Name2\_Variants** (read only)

**Syntax:** StringArray[,] = Name\_Variants  
VariantName = Name\_Variants[i,0]  
VariantGender = Name\_Variants[i,1]  
StringArray = Name2\_Variants  
VariantName2 = Name2\_Variants[i,0]  
VariantGender2 = Name2\_Variants[i,1]

**Description:**

After invoking the “Parse” method, this property is set to a list of common name variants associated with the names returned in “First” and “First2”. Values will be a list of one or more common name variants.

**Note:** Name variants are only returned when “Name\_Variant\_Check” property is set to “True”.

---

### **Name\_Filtered\_Data** (read only)

**Syntax:** String = Name\_Filtered\_Data

**Description:**

After invoking the “Parse” method, this property will contain all data that was filtered out before processing according to the [NameFilter] section of “NetGender.ref” file.

*See “Updating User Control Tables” later in this guide for information on customizing this file.*

### **Return\_Code** (read only)

**Syntax:** String = Return\_Code

#### **Description:**

After invoking the “Parse” method, this property is set to blank upon successful completion. Most exceptions occur on the first invocation. *This property should be examined on each return from NetGender.*

#### **Common Return Codes:**

<b>G00</b>	Gender file wrong version
<b>G30</b>	Gender file open/read error
<b>G35</b>	Gender file not found ( <i>see “Gender_File_Path” property</i> )
<b>R30</b>	Reference file open/read error
<b>R35</b>	Reference file not found ( <i>see “Reference_File_Path” property</i> )
<b>S00</b>	Unrecognized name style ( <i>see “Name_Style” property</i> )
<b>T00</b>	Prefix/Suffix Table Limit Reached (1,024)
<b>T01</b>	Name Variant Override Table Limit Reached (256)
<b>T02</b>	Gender Override Table Limit Reached (1,024)
<b>T03</b>	Spelling Override Table Limit Reached (1,024)
<b>L00</b>	Evaluation period expired
<b>L01</b>	Static key validation failed ( <i>see “Static_Key” property</i> )
<b>L50</b>	Evaluation license error

### Clear

**Syntax:** NetGender.Clear

**Description:**

When this method is invoked, all properties are cleared with the exception of “Static\_Key”, “Static\_Key\_Name” and “Reference\_File\_Path”.

---

### Parse

**Syntax:** NetGender.Parse

**Description:**

When this method is invoked, each element of the “Name\_In” property will be inspected. Multiple names are separated, each name is gender coded and placed into the “Name\_Out” property. Each individual element of the “Name\_In” property will be placed into the appropriate name component property. And, if the “Name\_Variant\_Check” property is set to “True”, a list of common name variants is returned in “Name\_Variants” and “Name2\_Variants”. The “Return\_Code” property is also set and should be checked after each invocation of the Parse method. See “Return\_Code” property.

### Updating User Control Tables

“NetGender.ref” is a file containing the name standardization control tables. It is located by default in the “NetGender” installation folder. Use Notepad or a similar text editor to edit the contents. Detailed information on the format of the entries is contained within the file. This file can also be relocated. See “*Reference\_File\_Path*” property.

NetGender allows you to specify which Prefixes and Suffixes are to be recognized as well as your preferred abbreviations.

---

**Extensive tables are included. Below are a few examples:**

**[NamePrefix]**

M&M	Mr & Mrs
M/M	Mr & Mrs

**[NameSuffix]**

PHD	Ph.D.
MANAGER	Mgr

**[NameConnector]**

&  
C/O  
GUARDIAN OF

**[NameFilter]**

ETAL  
TRUST

**[CompanyOverride]**

C	TED’S SHEDS
U	BAYOU

**[GenderOverride]**

M	BILLY-JOE
F	BO PEEP
L	EXOTIC

**[NameVariantOverride]**

WOODY = WOODIE, WOODROW, WOODFORD

**[SpellingOverride]**

diGenova  
The UPS Store

### Deploying Your Applications

Be sure to include the following in your deployment package:

**NetGender.dll** – usually placed in the application folder or Global Assembly Cache (GAC)

**NetGender.ref** – usually placed in the application folder\*

**NetGender.gnd** – usually placed in the application folder\*

\* “NetGender.ref” (reference file) and “NetGender.gnd” (gender file) can be placed anywhere on the target machine as long as the full path to it is specified in the “Reference\_File\_Path” and “Gender\_File\_Path” properties.

---

In addition to the above, there is a common runtime that can be placed in the application folder or the Global Assembly Cache (GAC) of the target machine.

**Fujitsu.COBOL.dll**

---

### Evaluation License

The evaluation license is valid for a period of 7 days or up to 1,000 calls.

[Sales@SoftwareCompany.com](mailto:Sales@SoftwareCompany.com)

[Support@SoftwareCompany.com](mailto:Support@SoftwareCompany.com)