# NetSpell.SpellChecker Namespace

## Classes

<table>
<thead>
<tr>
<th>Class</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>AssemblyInfo</strong></td>
<td>AssemblyInfo class</td>
</tr>
<tr>
<td><strong>ReplaceWordEventArgs</strong></td>
<td>Summary description for ReplaceWordEventArgs.</td>
</tr>
<tr>
<td><strong>Spelling</strong></td>
<td>The Spelling class encapsulates the functions necessary to check the spelling of inputted text.</td>
</tr>
<tr>
<td><strong>SpellingEventArgs</strong></td>
<td>Class sent to the event handler when the DoubleWord or MisspelledWord event occurs</td>
</tr>
</tbody>
</table>

## Delegates

<table>
<thead>
<tr>
<th>Delegate</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Spelling.DeletedWordEventHandler</strong></td>
<td>This represents the delegate method prototype that event receivers must implement</td>
</tr>
<tr>
<td><strong>Spelling.DoubledWordEventHandler</strong></td>
<td>This represents the delegate method prototype that event receivers must implement</td>
</tr>
<tr>
<td><strong>Spelling.EndOfTextEventHandler</strong></td>
<td>This represents the delegate method prototype that event receivers must implement</td>
</tr>
<tr>
<td><strong>Spelling.IgnoredWordEventHandler</strong></td>
<td>This represents the delegate method prototype that event receivers must implement</td>
</tr>
<tr>
<td><strong>Enumeration</strong></td>
<td><strong>Description</strong></td>
</tr>
<tr>
<td>------------------------------</td>
<td>----------------------------------------------</td>
</tr>
<tr>
<td><code>Spelling.MisspelledWordEventHandler</code></td>
<td>This represents the delegate method prototype that event receivers must implement</td>
</tr>
<tr>
<td><code>Spelling.ReplacedWordEventHandler</code></td>
<td>This represents the delegate method prototype that event receivers must implement</td>
</tr>
</tbody>
</table>

### Enumerations

<table>
<thead>
<tr>
<th><strong>Enumeration</strong></th>
<th><strong>Description</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><code>Spelling.SuggestionEnum</code></td>
<td>The suggestion strategy to use when generating suggestions</td>
</tr>
</tbody>
</table>

Copyright (C) 2003 Paul Welter
AssemblyInfo Class

AssemblyInfo class

For a list of all members of this type, see AssemblyInfo Members.

**System.Object**  **AssemblyInfo**

<table>
<thead>
<tr>
<th>[Visual Basic]</th>
<th>Public Class AssemblyInfo</th>
</tr>
</thead>
<tbody>
<tr>
<td>[C#]</td>
<td>public class AssemblyInfo</td>
</tr>
</tbody>
</table>

Requirements

**Namespace:** NetSpell.SpellChecker

**Assembly:** NetSpell.SpellChecker (in NetSpell.SpellChecker.dll)

See Also

[AssemblyInfo Members](#) | [NetSpell.SpellChecker Namespace](#)

Copyright (C) 2003 Paul Welter
## AssemblyInfo Members

### AssemblyInfo overview

### Public Instance Constructors

<table>
<thead>
<tr>
<th>Constructor</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>AssemblyInfo Constructor</strong></td>
<td>Initialized the AssemblyInfo class with the given type</td>
</tr>
</tbody>
</table>

### Public Instance Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>CodeBase</strong></td>
<td>CodeBase of Assembly</td>
</tr>
<tr>
<td><strong>Company</strong></td>
<td>Company of Assembly</td>
</tr>
<tr>
<td><strong>Copyright</strong></td>
<td>Copyright of Assembly</td>
</tr>
<tr>
<td><strong>Description</strong></td>
<td>Description of Assembly</td>
</tr>
<tr>
<td><strong>FullName</strong></td>
<td>FullName of Assembly</td>
</tr>
<tr>
<td><strong>Name</strong></td>
<td>Name of Assembly</td>
</tr>
<tr>
<td><strong>Product</strong></td>
<td>Product of Assembly</td>
</tr>
<tr>
<td><strong>Title</strong></td>
<td>Title of Assembly</td>
</tr>
<tr>
<td><strong>Version</strong></td>
<td>Version of Assembly</td>
</tr>
</tbody>
</table>

### Public Instance Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Equals</strong> (inherited from Object)</td>
<td>Determines whether the specified Object is equal to the current Object.</td>
</tr>
<tr>
<td><strong>GetHashCode</strong> (inherited from Object)</td>
<td>Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table.</td>
</tr>
<tr>
<td><strong>GetType</strong> (inherited from Object)</td>
<td>Gets the Type of the current instance.</td>
</tr>
<tr>
<td><strong>ToString</strong> (inherited from Object)</td>
<td>Returns a String that represents the current Object.</td>
</tr>
</tbody>
</table>
## Protected Instance Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>Finalize</code> (inherited from Object)</td>
<td>Allows an <a href="#">Object</a> to attempt to free resources and perform other cleanup operations before the <a href="#">Object</a> is reclaimed by garbage collection.</td>
</tr>
<tr>
<td><code>MemberwiseClone</code> (inherited from Object)</td>
<td>Creates a shallow copy of the current <a href="#">Object</a>.</td>
</tr>
</tbody>
</table>

### See Also

* [AssemblyInfo Class](#)| [NetSpell.SpellChecker Namespace](#)

---

Copyright (C) 2003 Paul Welter
Initialized the AssemblyInfo class with the given type

[Visual Basic]
Public Sub New( _
    ByVal type As Type _
)

[C#]
public AssemblyInfo(
    Type type
);
AssemblyInfo Properties

The properties of the AssemblyInfo class are listed below. For a complete list of AssemblyInfo class members, see the AssemblyInfo Members topic.

Public Instance Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CodeBase</td>
<td>CodeBase of Assembly</td>
</tr>
<tr>
<td>Company</td>
<td>Company of Assembly</td>
</tr>
<tr>
<td>Copyright</td>
<td>Copyright of Assembly</td>
</tr>
<tr>
<td>Description</td>
<td>Description of Assembly</td>
</tr>
<tr>
<td>FullName</td>
<td>FullName of Assembly</td>
</tr>
<tr>
<td>Name</td>
<td>Name of Assembly</td>
</tr>
<tr>
<td>Product</td>
<td>Product of Assembly</td>
</tr>
<tr>
<td>Title</td>
<td>Title of Assembly</td>
</tr>
<tr>
<td>Version</td>
<td>Version of Assembly</td>
</tr>
</tbody>
</table>

See Also

AssemblyInfo Class | NetSpell.SpellChecker Namespace

Copyright (C) 2003 Paul Welter
AssemblyInfo.CodeBase Property

CodeBase of Assembly

[Visual Basic] Public ReadOnly Property CodeBase

[C#]
public string CodeBase {get;}

See Also

AssemblyInfo Class | NetSpell.SpellChecker Namespace

Copyright (C) 2003 Paul Welter
**AssemblyInfo.Company Property**

Company of Assembly

**[Visual Basic]**

```vbnet
Public Readonly Property Company
```

**[C#]**

```csharp
public string Company {get;}
```

**See Also**

[AssemblyInfo Class] | [NetSpell.SpellChecker Namespace]

**Copyright (C) 2003 Paul Welter**
AssemblyInfo.Copyright Property

Copyright of Assembly

[Visual Basic] Public ReadOnly Property Copyright

[C#]
public string Copyright {get;}

See Also

AssemblyInfo Class | NetSpell.SpellChecker Namespace

Copyright (C) 2003 Paul Welter
AssemblyInfo.Description Property

Description of Assembly

[Visual Basic]
Public Readonly Property Description

[C#]
public string Description {get;}

See Also

AssemblyInfo Class | NetSpell.SpellChecker Namespace

Copyright (C) 2003 Paul Welter
AssemblyInfo.FullName Property

FullName of Assembly

| [Visual Basic] Public Readonly Property FullName As [C#] public string FullName {get;} |

See Also

- AssemblyInfo Class | NetSpell.SpellChecker Namespace

Copyright (C) 2003 Paul Welter
AssemblyInfo.Name Property

Name of Assembly

[Visual Basic] Public ReadOnly Property Name

[C#]
public string Name {get;}

See Also

AssemblyInfo Class | NetSpell.SpellChecker Namespace

Copyright (C) 2003 Paul Welter
AssemblyInfo.Product Property

Product of Assembly

[Visual Basic] Public ReadOnly Property Product

[C#]
public string Product {get;}

See Also

AssemblyInfo Class | NetSpell.SpellChecker Namespace

Copyright (C) 2003 Paul Welter
AssemblyInfo.Title Property

Title of Assembly

[Visual Basic] Public ReadOnly Property Title

[C#]
public string Title {get;}

See Also

AssemblyInfo Class | NetSpell.SpellChecker Namespace

Copyright (C) 2003 Paul Welter
AssemblyInfo.Version Property

Version of Assembly

[Visual Basic] Public ReadOnly Property Version

[C#]
public string Version {get;}

See Also

AssemblyInfo Class | NetSpell.SpellChecker Namespace

Copyright (C) 2003 Paul Welter
ReplaceWordEventArgs Class

Summary description for ReplaceWordEventArgs.
For a list of all members of this type, see ReplaceWordEventArgs Members.

System.Object  EventArgs
    SpellingEventArgs
    ReplaceWordEventArgs

[Visual Basic]
Public Class ReplaceWordEventArgs
    Inherits SpellingEventArgs

[C#]
public class ReplaceWordEventArgs :
    SpellingEventArgs

Requirements

Namespace: NetSpell.SpellChecker

Assembly: NetSpell.SpellChecker (in NetSpell.SpellChecker.dll)

See Also

ReplaceWordEventArgs Members  | NetSpell.SpellChecker Namespace

Copyright (C) 2003 Paul Welter
### ReplaceWordEventArgs Members

#### ReplaceWordEventArgs overview

### Public Instance Constructors

<table>
<thead>
<tr>
<th>Constructor</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>ReplaceWordEventArgs Constructor</strong></td>
<td>Class sent to the event handler when the ReplacedWord Event is fired</td>
</tr>
</tbody>
</table>

### Public Instance Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>ReplacementWord</strong></td>
<td>The word to use in replacing the misspelled word</td>
</tr>
<tr>
<td><strong>TextIndex</strong> (inherited from SpellingEventArgs)</td>
<td>Text index of the WordEvent</td>
</tr>
<tr>
<td><strong>Word</strong> (inherited from SpellingEventArgs)</td>
<td>Word that caused the WordEvent</td>
</tr>
<tr>
<td><strong>WordIndex</strong> (inherited from SpellingEventArgs)</td>
<td>Word index of the WordEvent</td>
</tr>
</tbody>
</table>

### Public Instance Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Equals</strong> (inherited from Object)</td>
<td>Determines whether the specified <strong>Object</strong> is equal to the current <strong>Object</strong>.</td>
</tr>
<tr>
<td><strong>GetHashCode</strong> (inherited from Object)</td>
<td>Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table.</td>
</tr>
<tr>
<td><strong>GetType</strong> (inherited from Object)</td>
<td>Gets the <strong>Type</strong> of the current instance.</td>
</tr>
<tr>
<td><strong>ToString</strong> (inherited from Object)</td>
<td>Returns a <strong>String</strong> that represents the current <strong>Object</strong>.</td>
</tr>
</tbody>
</table>

### Protected Instance Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
</table>

| **Finalize** (inherited from **Object**) | Allows an **Object** to attempt to free resources and perform other cleanup operations before the **Object** is reclaimed by garbage collection. |
| **MemberwiseClone** (inherited from **Object**) | Creates a shallow copy of the current **Object**. |

**See Also**

[ReplaceWordEventArgs Class] | [NetSpell.SpellChecker Namespace]

*Copyright (C) 2003 Paul Welter*
ReplaceWordEventArgs Constructor

Class sent to the event handler when the ReplacedWord Event is fired

[Visual Basic]
Public Sub New(_
    ByVal replacementWord As String, _
    ByVal word As String, _
    ByVal wordIndex As Integer, _
    ByVal textIndex As Integer _
)

[C#]
public ReplaceWordEventArgs(
    string replacementWord,
    string word,
    int wordIndex,
    int textIndex
);

See Also
ReplaceWordEventArgs Class | NetSpell.SpellChecker Namespace

Copyright (C) 2003 Paul Welter
ReplaceEventArgs Properties

The properties of the ReplaceEventArgs class are listed below. For a complete list of ReplaceEventArgs class members, see the ReplaceEventArgs Members topic.

Public Instance Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ReplacementWord</td>
<td>The word to use in replacing the misspelled word</td>
</tr>
<tr>
<td>TextIndex</td>
<td>Text index of the WordEvent</td>
</tr>
<tr>
<td>Word</td>
<td>Word that caused the WordEvent</td>
</tr>
<tr>
<td>WordIndex</td>
<td>Word index of the WordEvent</td>
</tr>
</tbody>
</table>

See Also

ReplaceEventArgs Class | NetSpell.SpellChecker Namespace

Copyright (C) 2003 Paul Welter
ReplaceWordEventArgs.ReplacementWord Property

The word to use in replacing the misspelled word

[Visual Basic] Public ReadOnly Property ReplacementWord As String

[C#]
public string ReplacementWord {get;}

See Also

ReplaceWordEventArgs Class | NetSpell.SpellChecker Namespace

Copyright (C) 2003 Paul Welter
The Spelling class encapsulates the functions necessary to check the spelling of inputted text.

For a list of all members of this type, see Spelling Members.

System.Object MarshalByRefObject Component Spelling

[Visual Basic]
Public Class Spelling
    Inherits Component

[C#]
public class Spelling : Component

Requirements

Namespace: NetSpell.SpellChecker

Assembly: NetSpell.SpellChecker (in NetSpell.SpellChecker.dll)

See Also

Spelling Members | NetSpell.SpellChecker Namespace

Copyright (C) 2003 Paul Welter
# Spelling Members

## Spelling overview

### Public Instance Constructors

<table>
<thead>
<tr>
<th>Constructor</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Spelling</strong></td>
<td>Overloaded. Initializes a new instance of the Spelling class.</td>
</tr>
</tbody>
</table>

### Public Instance Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>AlertComplete</strong></td>
<td>Display the 'Spell Check Complete' alert.</td>
</tr>
<tr>
<td><strong>Container</strong> (inherited from Component)</td>
<td>Gets the IContainer that contains the Component.</td>
</tr>
<tr>
<td><strong>CurrentWord</strong></td>
<td>The current word being spell checked from the text property.</td>
</tr>
<tr>
<td><strong>Dictionary</strong></td>
<td>The WordDictionary object to use when spell checking.</td>
</tr>
<tr>
<td><strong>IgnoreAllCapsWords</strong></td>
<td>Ignore words with all capital letters when spell checking.</td>
</tr>
<tr>
<td><strong>IgnoreHtml</strong></td>
<td>Ignore html tags when spell checking.</td>
</tr>
<tr>
<td><strong>IgnoreList</strong></td>
<td>List of words to automatically ignore.</td>
</tr>
<tr>
<td><strong>IgnoreWordsWithDigits</strong></td>
<td>Ignore words with digits when spell checking.</td>
</tr>
<tr>
<td><strong>MaxSuggestions</strong></td>
<td>The maximum number of suggestions to generate.</td>
</tr>
<tr>
<td><strong>ReplaceList</strong></td>
<td>List of words and replacement values to automatically replace.</td>
</tr>
<tr>
<td><strong>ReplacementWord</strong></td>
<td>The word to used when replacing the misspelled word.</td>
</tr>
<tr>
<td><strong>ShowDialog</strong></td>
<td>Determines if the spell checker</td>
</tr>
</tbody>
</table>
should use its internal suggestions and options dialogs.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Site</strong> (inherited from Component)</td>
<td>Gets or sets the ISite of the Component.</td>
</tr>
<tr>
<td><strong>SuggestionForm</strong></td>
<td>The internal spelling suggestions dialog form</td>
</tr>
<tr>
<td><strong>SuggestionMode</strong></td>
<td>The suggestion strategy to use when generating suggestions</td>
</tr>
<tr>
<td><strong>Suggestions</strong></td>
<td>An array of word suggestions for the correct spelling of the misspelled word</td>
</tr>
<tr>
<td><strong>Text</strong></td>
<td>The text to spell check</td>
</tr>
<tr>
<td><strong>TextIndex</strong></td>
<td>TextIndex is the index of the current text being spell checked</td>
</tr>
<tr>
<td><strong>WordCount</strong></td>
<td>The number of words being spell checked</td>
</tr>
<tr>
<td><strong>WordIndex</strong></td>
<td>WordIndex is the index of the current word being spell checked</td>
</tr>
</tbody>
</table>

**Public Instance Methods**

- **CreateObjRef** (inherited from MarshalByRefObject) Creates an object that contains all the relevant information required to generate a proxy used to communicate with a remote object.
- **DeleteWord** Deletes the CurrentWord from the Text Property
- **Dispose** (inherited from Component) Overloaded. Releases all resources used by the Component.
- **EditDistance** Overloaded. Calculates the
<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>minimum number of change, inserts or deletes required to change firstWord into secondWord</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Equals (inherited from Object)</strong></td>
<td>Determines whether the specified <strong>Object</strong> is equal to the current <strong>Object</strong>.</td>
</tr>
<tr>
<td><strong>GetHashCode (inherited from Object)</strong></td>
<td>Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table.</td>
</tr>
<tr>
<td><strong>GetLifetimeService (inherited from MarshalByRefObject)</strong></td>
<td>Retrieves the current lifetime service object that controls the lifetime policy for this instance.</td>
</tr>
<tr>
<td><strong>GetType (inherited from Object)</strong></td>
<td>Gets the <strong>Type</strong> of the current instance.</td>
</tr>
<tr>
<td><strong>GetWordIndexFromTextIndex</strong></td>
<td>Gets the word index from the text index. Use this method to find a word based on the text position.</td>
</tr>
<tr>
<td><strong>IgnoreAllWord</strong></td>
<td>Ignores all instances of the CurrentWord in the Text Property</td>
</tr>
<tr>
<td><strong>IgnoreWord</strong></td>
<td>Ignores the instances of the CurrentWord in the Text Property</td>
</tr>
<tr>
<td><strong>InitializeLifetimeService (inherited from MarshalByRefObject)</strong></td>
<td>Obtains a lifetime service object to control the lifetime policy for this instance.</td>
</tr>
<tr>
<td><strong>ReplaceAllWord</strong></td>
<td>Overloaded. Replaces all instances of the CurrentWord in the Text Property</td>
</tr>
<tr>
<td><strong>ReplaceWord</strong></td>
<td>Overloaded. Replaces the instances of the CurrentWord in</td>
</tr>
<tr>
<td>Event</td>
<td>Description</td>
</tr>
<tr>
<td>-------</td>
<td>-------------</td>
</tr>
<tr>
<td><strong>SpellCheck</strong></td>
<td>Overloaded. Spell checks the words in the Text property starting at the WordIndex position.</td>
</tr>
<tr>
<td><strong>Suggest</strong></td>
<td>Overloaded. Populates the Suggestions property with word suggestions for the word</td>
</tr>
<tr>
<td><strong>TestWord</strong></td>
<td>Checks to see if the word is in the dictionary</td>
</tr>
<tr>
<td><strong>ToString</strong> (inherited from Component)</td>
<td></td>
</tr>
</tbody>
</table>

**Public Instance Events**

<table>
<thead>
<tr>
<th>Event</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>✍ <strong>DeletedWord</strong></td>
<td>This event is fired when a word is deleted</td>
</tr>
<tr>
<td>✍ <strong>Disposed</strong> (inherited from Component)</td>
<td>Adds an event handler to listen to the Disposed event on the component.</td>
</tr>
<tr>
<td>✍ <strong>DoubledWord</strong></td>
<td>This event is fired when word is detected two times in a row</td>
</tr>
<tr>
<td>✍ <strong>EndOfText</strong></td>
<td>This event is fired when the spell checker reaches the end of the text in the Text property</td>
</tr>
<tr>
<td>✍ <strong>IgnoredWord</strong></td>
<td>This event is fired when a word is skipped</td>
</tr>
<tr>
<td>✍ <strong>MisspelledWord</strong></td>
<td>This event is fired when the spell checker finds a word that is not in the dictionaries</td>
</tr>
<tr>
<td>✍ <strong>ReplacedWord</strong></td>
<td>This event is fired when a word is replace</td>
</tr>
</tbody>
</table>

**Protected Instance Properties**
### DesignMode (inherited from Component)
Gets a value that indicates whether the Component is currently in design mode.

### Events (inherited from Component)
Gets the list of event handlers that are attached to this Component.

## Protected Instance Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Dispose</strong></td>
<td>Overloaded. Clean up any resources being used.</td>
</tr>
<tr>
<td><strong>Finalize</strong> (inherited from Component)</td>
<td>Releases unmanaged resources and performs other cleanup operations before the Component is reclaimed by garbage collection.</td>
</tr>
<tr>
<td><strong>GetService</strong> (inherited from Component)</td>
<td>Returns an object that represents a service provided by the Component or by its Container.</td>
</tr>
<tr>
<td><strong>MemberwiseClone</strong> (inherited from Object)</td>
<td>Creates a shallow copy of the current Object.</td>
</tr>
<tr>
<td><strong>OnDeletedWord</strong></td>
<td>This is the method that is responsible for notifying receivers that the event occurred</td>
</tr>
<tr>
<td><strong>OnDoubledWord</strong></td>
<td>This is the method that is responsible for notifying receivers that the event occurred</td>
</tr>
<tr>
<td><strong>OnEndOfText</strong></td>
<td>This is the method that is responsible for notifying receivers that the event occurred</td>
</tr>
<tr>
<td><strong>OnIgnoredWord</strong></td>
<td>This is the method that is</td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>OnMisspelledWord</td>
<td>This is the method that is responsible for notifying receivers that the event occurred</td>
</tr>
<tr>
<td>OnReplacedWord</td>
<td>This is the method that is responsible for notifying receivers that the event occurred</td>
</tr>
</tbody>
</table>

See Also

- **Spelling Class** | **NetSpell.SpellChecker Namespace**

*Copyright (C) 2003 Paul Welter*
Spelling Constructor

Initializes a new instance of the SpellCheck class

Overload List

Initializes a new instance of the SpellCheck class

public Spelling();

Required for Windows.Forms Class Composition Designer support

public Spelling(IContainer);

See Also

Spelling Class | NetSpell.SpellChecker Namespace

Copyright (C) 2003 Paul Welter
Spelling Constructor ()

Initializes a new instance of the SpellCheck class

[Visual Basic] Overloads Public Sub New()

[C#]
public Spelling();

See Also

Spelling Class | NetSpell.SpellChecker Namespace | Spelling Constructor Overload List

Copyright (C) 2003 Paul Welter
Spelling Constructor (IContainer)

Required for Windows.Forms Class Composition Designer support

[Visual Basic] Overloads Public Sub New( _
    ByVal container As IContainer _
)

[C#]
public Spelling(
    IContainer container
);

See Also

Spelling Class | NetSpell.SpellChecker Namespace | Spelling Constructor Overload List

Copyright (C) 2003 Paul Welter
Spelling Properties

The properties of the **Spelling** class are listed below. For a complete list of **Spelling** class members, see the **Spelling Members** topic.

### Public Instance Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>AlertComplete</code></td>
<td>Displays the 'Spell Check Complete' alert.</td>
</tr>
<tr>
<td><code>Container</code></td>
<td>Gets the IContainer that contains the Component.</td>
</tr>
<tr>
<td><code>CurrentWord</code></td>
<td>The current word being spell checked from the text property.</td>
</tr>
<tr>
<td><code>Dictionary</code></td>
<td>The WordDictionary object to use when spell checking.</td>
</tr>
<tr>
<td><code>IgnoreAllCapsWords</code></td>
<td>Ignore words with all capital letters when spell checking.</td>
</tr>
<tr>
<td><code>IgnoreHtml</code></td>
<td>Ignore html tags when spell checking.</td>
</tr>
<tr>
<td><code>IgnoreList</code></td>
<td>List of words to automatically ignore.</td>
</tr>
<tr>
<td><code>IgnoreWordsWithDigits</code></td>
<td>Ignore words with digits when spell checking.</td>
</tr>
<tr>
<td><code>MaxSuggestions</code></td>
<td>The maximum number of suggestions to generate.</td>
</tr>
<tr>
<td><code>ReplaceList</code></td>
<td>List of words and replacement values to automatically replace.</td>
</tr>
<tr>
<td><code>ReplacementWord</code></td>
<td>The word to used when replacing the misspelled word.</td>
</tr>
<tr>
<td><code>ShowDialog</code></td>
<td>Determines if the spell checker should use its internal suggestions and options dialogs.</td>
</tr>
<tr>
<td><strong>Site</strong> (inherited from Component)</td>
<td>Gets or sets the <code>ISite</code> of the <code>Component</code>.</td>
</tr>
<tr>
<td><strong>SuggestionForm</strong></td>
<td>The internal spelling suggestions dialog form</td>
</tr>
<tr>
<td><strong>SuggestionMode</strong></td>
<td>The suggestion strategy to use when generating suggestions</td>
</tr>
<tr>
<td><strong>Suggestions</strong></td>
<td>An array of word suggestions for the correct spelling of the misspelled word</td>
</tr>
<tr>
<td><strong>Text</strong></td>
<td>The text to spell check</td>
</tr>
<tr>
<td><strong>TextIndex</strong></td>
<td>TextIndex is the index of the current text being spell checked</td>
</tr>
<tr>
<td><strong>WordCount</strong></td>
<td>The number of words being spell checked</td>
</tr>
<tr>
<td><strong>WordIndex</strong></td>
<td>WordIndex is the index of the current word being spell checked</td>
</tr>
</tbody>
</table>

### Protected Instance Properties

| **DesignMode** (inherited from Component) | Gets a value that indicates whether the `Component` is currently in design mode. |
| **Events** (inherited from Component) | Gets the list of event handlers that are attached to this `Component`. |

### See Also

- [Spelling Class](#) | [NetSpell.SpellChecker Namespace](#) |

---

**Copyright (C) 2003 Paul Welter**
Spelling.AlertComplete Property

Display the 'Spell Check Complete' alert.

[Visual Basic]
Public Property AlertComplete

[C#]
public bool AlertComplete {get; set;}

See Also

Spelling Class | NetSpell.SpellChecker Namespace

Copyright (C) 2003 Paul Welter
Spelling.CurrentWord Property

The current word being spell checked from the text property

[Visual Basic] Public ReadOnly Property CurrentWord

[C#]
public string CurrentWord {get;}

See Also

Spelling Class | NetSpell.SpellChecker Namespace

Copyright (C) 2003 Paul Welter
**Spelling.Dictionary Property**

The WordDictionary object to use when spell checking

[Visual Basic] Public Property Dictionary As }

[C#] public Dictionary.WordDictionary Dictionary }

See Also

[Spelling Class] [NetSpell.SpellChecker Namespace]

Copyright (C) 2003 Paul Welter
Spelling.IgnoreAllCapsWords Property

Ignore words with all capital letters when spell checking.

[Visual Basic] Public Property IgnoreAllCapsWords

[C#] public bool IgnoreAllCapsWords {get; set;}

See Also

Spelling Class | NetSpell.SpellChecker Namespace

Copyright (C) 2003 Paul Welter
Spelling.IgnoreHtml Property

Ignore html tags when spell checking

[Visual Basic] Public Property IgnoreHtml As !

[C#] public bool IgnoreHtml {get; set;}

See Also

Spelling Class | NetSpell.SpellChecker Namespace

Copyright (C) 2003 Paul Welter
Spelling.IgnoreList Property

List of words to automatically ignore

[Visual Basic] Public ReadOnly Property IgnoreList As ArrayList

[C#] public System.Collections.ArrayList IgnoreList

Remarks

When IgnoreAllWord is clicked, the CurrentWord is added to this list.

See Also

Spelling Class | NetSpell.SpellChecker Namespace

Copyright (C) 2003 Paul Welter
**Spelling.IgnoreWordsWithDigits Property**

Ignore words with digits when spell checking

**[Visual Basic]**

Public Property IgnoreWordsWithDigits

**[C#]**

public bool IgnoreWordsWithDigits {get; set;}

See Also

[Spelling Class] | [NetSpell.SpellChecker Namespace]

Copyright (C) 2003 Paul Welter
Spelling.MaxSuggestions Property

The maximum number of suggestions to generate

[Visual Basic] Public Property MaxSuggestions

[C#]
public int MaxSuggestions {get; set;}

See Also

Spelling Class | NetSpell.SpellChecker Namespace

Copyright (C) 2003 Paul Welter
Spelling.ReplaceList Property

List of words and replacement values to automatically replace

[Visual Basic] Public ReadOnly Property ReplaceList

[C#] public System.Collections.Hashtable ReplaceList

Remarks

When ReplaceAllWord is clicked, the CurrentWord is added to this list

See Also

Spelling Class | NetSpell.SpellChecker Namespace

Copyright (C) 2003 Paul Welter
The word to used when replacing the misspelled word

[Visual Basic] Public Property ReplacementWord

[C#]
public string ReplacementWord {get; set;}

See Also

Spelling Class | NetSpell.SpellChecker Namespace | ReplaceAllWord | ReplaceWord

Copyright (C) 2003 Paul Welter
Spelling.ShowDialog Property

Determines if the spell checker should use its internal suggestions and options dialogs.

[Visual Basic]
Public Property ShowDialog As Boolean

[C#]
public bool ShowDialog {get; set;}

See Also
Spelling Class  |  NetSpell.SpellChecker Namespace

Copyright (C) 2003 Paul Welter
The internal spelling suggestions dialog form

[Visual Basic] Public ReadOnly Property SuggestionForm

[C#] public Forms.SuggestionForm SuggestionForm { get; }

See Also

Spelling Class | NetSpell.SpellChecker Namespace

Copyright (C) 2003 Paul Welter
### Spelling.SuggestionMode Property

The suggestion strategy to use when generating suggestions

<table>
<thead>
<tr>
<th>[Visual Basic] Public Property SuggestionMode</th>
</tr>
</thead>
</table>

| [C#] public Spelling.SuggestionEnum SuggestionMode |

See Also

- [Spelling Class](#) | [NetSpell.SpellChecker Namespace](#)

---

Copyright (C) 2003 Paul Welter
**Spelling.Suggestions Property**

An array of word suggestions for the correct spelling of the misspelled word

[Visual Basic] Public ReadOnly Property Suggestions

[C#]
public System.Collections.ArrayList Suggestions {get;}

See Also

[Spelling Class] | [NetSpell.SpellChecker Namespace] | [Suggest] | [SpellCheck] | [MaxSuggestions]

---

Copyright (C) 2003 Paul Welter
The text to spell check

[Visual Basic] Public Property Text As String

[C#]
public string Text {get; set;}

See Also

Spelling Class | NetSpell.SpellChecker Namespace

Copyright (C) 2003 Paul Welter
Spelling.TextIndex Property

TextIndex is the index of the current text being spell checked

[Visual Basic] Public ReadOnly Property TextIndex

[C#]
public int TextIndex {get;}

See Also

Spelling Class | NetSpell.SpellChecker Namespace

Copyright (C) 2003 Paul Welter
Spelling.WordCount Property

The number of words being spell checked

[Visual Basic]
Public Readonly Property WordCount

[C#]
public int WordCount {get;}

See Also

Spelling Class | NetSpell.SpellChecker Namespace

Copyright (C) 2003 Paul Welter
Spelling.WordIndex Property

WordIndex is the index of the current word being spell checked

[Visual Basic] Public Property WordIndex As Integer

[C#]
public int WordIndex {get; set;}

See Also

Spelling Class | NetSpell.SpellChecker Namespace

Copyright (C) 2003 Paul Welter
The methods of the **Spelling** class are listed below. For a complete list of **Spelling** class members, see the **Spelling Members** topic.

### Public Instance Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>CreateObjRef</strong> (inherited from MarshalByRefObject)</td>
<td>Creates an object that contains all the relevant information required to generate a proxy used to communicate with a remote object.</td>
</tr>
<tr>
<td><strong>DeleteWord</strong></td>
<td>Deletes the CurrentWord from the Text Property</td>
</tr>
<tr>
<td><strong>Dispose</strong> (inherited from Component)</td>
<td>Overloaded. Releases all resources used by the <strong>Component</strong>.</td>
</tr>
<tr>
<td><strong>EditDistance</strong></td>
<td>Overloaded. Calculates the minimum number of change, inserts or deletes required to change firstWord into secondWord</td>
</tr>
<tr>
<td><strong>Equals</strong> (inherited from <strong>Object</strong>)</td>
<td>Determines whether the specified <strong>Object</strong> is equal to the current <strong>Object</strong>.</td>
</tr>
<tr>
<td><strong>GetHashCode</strong> (inherited from <strong>Object</strong>)</td>
<td>Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table.</td>
</tr>
<tr>
<td><strong>GetLifetimeService</strong> (inherited from MarshalByRefObject)</td>
<td>Retrieves the current lifetime service object that controls the lifetime policy for this instance.</td>
</tr>
<tr>
<td><strong>GetType</strong> (inherited from <strong>Object</strong>)</td>
<td>Gets the <strong>Type</strong> of the current instance.</td>
</tr>
<tr>
<td><strong>GetWordIndexFromTextIndex</strong></td>
<td>Gets the word index from the</td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td><strong>IgnoreAllWord</strong></td>
<td>Ignores all instances of the CurrentWord in the Text Property</td>
</tr>
<tr>
<td><strong>IgnoreWord</strong></td>
<td>Ignores the instances of the CurrentWord in the Text Property</td>
</tr>
<tr>
<td><strong>InitializeLifetimeService</strong></td>
<td>(inherited from MarshalByRefObject) Obtains a lifetime service object to control the lifetime policy for this instance.</td>
</tr>
<tr>
<td><strong>ReplaceAllWord</strong></td>
<td>Overloaded. Replaces all instances of the CurrentWord in the Text Property</td>
</tr>
<tr>
<td><strong>ReplaceWord</strong></td>
<td>Overloaded. Replaces the instances of the CurrentWord in the Text Property</td>
</tr>
<tr>
<td><strong>SpellCheck</strong></td>
<td>Overloaded. Spell checks the words in the Text property starting at the WordIndex position.</td>
</tr>
<tr>
<td><strong>Suggest</strong></td>
<td>Overloaded. Populates the Suggestions property with word suggestions for the word</td>
</tr>
<tr>
<td><strong>TestWord</strong></td>
<td>Checks to see if the word is in the dictionary</td>
</tr>
<tr>
<td><strong>ToString</strong> (inherited from Component)</td>
<td></td>
</tr>
</tbody>
</table>

**Protected Instance Methods**

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Dispose</strong></td>
<td>Overloaded. Clean up any resources being used.</td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td><strong>Finalize</strong> (inherited from Component)</td>
<td>Releases unmanaged resources and performs other cleanup operations before the Component is reclaimed by garbage collection.</td>
</tr>
<tr>
<td><strong>GetService</strong> (inherited from Component)</td>
<td>Returns an object that represents a service provided by the Component or by its Container.</td>
</tr>
<tr>
<td><strong>MemberwiseClone</strong> (inherited from Object)</td>
<td>Creates a shallow copy of the current Object.</td>
</tr>
<tr>
<td><strong>OnDeletedWord</strong></td>
<td>This is the method that is responsible for notifying receivers that the event occurred.</td>
</tr>
<tr>
<td><strong>OnDoubledWord</strong></td>
<td>This is the method that is responsible for notifying receivers that the event occurred.</td>
</tr>
<tr>
<td><strong>OnEndOfText</strong></td>
<td>This is the method that is responsible for notifying receivers that the event occurred.</td>
</tr>
<tr>
<td><strong>OnIgnoredWord</strong></td>
<td>This is the method that is responsible for notifying receivers that the event occurred.</td>
</tr>
<tr>
<td><strong>OnMisspelledWord</strong></td>
<td>This is the method that is responsible for notifying receivers that the event occurred.</td>
</tr>
<tr>
<td><strong>OnReplacedWord</strong></td>
<td>This is the method that is responsible for notifying receivers that the event occurred.</td>
</tr>
</tbody>
</table>
See Also

Spelling Class | NetSpell.SpellChecker Namespace

Copyright (C) 2003 Paul Welter
Spelling.DeleteWord Method

Deletes the CurrentWord from the Text Property

[Visual Basic]
Public Sub DeleteWord()

[C#]
public void DeleteWord();

Remarks

Note, calling ReplaceWord with the ReplacementWord property set to an empty string has the same behavior as DeleteWord.

See Also

Spelling Class | NetSpell.SpellChecker Namespace

Copyright (C) 2003 Paul Welter
**Spelling.Dispose Method**

Clean up any resources being used.

**Overload List**

Inherited from Component.

```csharp
public void Dispose();
```

Clean up any resources being used.

```csharp
protected override void Dispose(bool);
```

**See Also**

- [Spelling Class](#) | [NetSpell.SpellChecker Namespace](#)

---

**Copyright (C) 2003 Paul Welter**
Clean up any resources being used.

```vbnet
Overrides Overloads Protected Sub Dispose(ByVal disposing As Boolean)
```

```csharp
protected override void Dispose(bool disposing)
```

See Also
- Spelling Class
- NetSpell.SpellChecker Namespace
- Spelling.Dispose Overload List

Copyright (C) 2003 Paul Welter
Spelling.EditDistance Method

Calculates the minimum number of change, inserts or deletes required to change firstWord into secondWord

Overload List

Calculates the minimum number of change, inserts or deletes required to change firstWord into secondWord

public int EditDistance(string,string);

Calculates the minimum number of change, inserts or deletes required to change firstWord into secondWord

public int EditDistance(string,string,bool);

See Also

Spelling Class | NetSpell.SpellChecker Namespace

Copyright (C) 2003 Paul Welter
Spelling.EditDistance Method (String, String)

Calculates the minimum number of change, inserts or deletes required to change firstWord into secondWord

```visualbasic
Overloads Public Function EditDistance(  
    ByVal source As String,  
    ByVal target As String  
) As Integer
```

```csharp
public int EditDistance(  
    string source,  
    string target
);
```

Parameters

source

The first word to calculate

target

The second word to calculate

Return Value

The number of edits to make firstWord equal secondWord

Remarks

This method automatically gives priority to matching the first and last char

See Also

Spelling Class | NetSpell.SpellChecker Namespace | Spelling.EditDistance Overload List

Copyright (C) 2003 Paul Welter
Spelling.EditDistance Method (String, String, Boolean)

Calculates the minimum number of change, inserts or deletes required to change firstWord into secondWord

**Visual Basic**

```vbnet
Overloads Public Function EditDistance(
    ByVal source As String, 
    ByVal target As String, 
    ByVal positionPriority As Boolean 
) As Integer
```

**C#**

```csharp
public int EditDistance(
    string source, 
    string target, 
    bool positionPriority 
);
```

**Parameters**

- **source**
  The first word to calculate

- **target**
  The second word to calculate

- **positionPriority**
  set to true if the first and last char should have priority

**Return Value**

The number of edits to make firstWord equal secondWord

**See Also**

- Spelling Class | NetSpell.SpellChecker Namespace | Spelling.EditDistance Overload List
Spelling.GetWordIndexFromTextIndex Method

Gets the word index from the text index. Use this method to find a word based on the text position.

[Visual Basic]
Public Function GetWordIndexFromTextIndex(
    ByVal textIndex As Integer
) As Integer

[C#]
public int GetWordIndexFromTextIndex(
    int textIndex
);

Parameters

textIndex
The index to search for

Return Value
The word index that the text index falls on

See Also

Spelling Class | NetSpell.SpellChecker Namespace

Copyright (C) 2003 Paul Welter
Spelling.IgnoreAllWord Method

Ignores all instances of the CurrentWord in the Text Property

[Visual Basic] Public Sub IgnoreAllWord()
[C#] public void IgnoreAllWord();

See Also

Spelling Class | NetSpell.SpellChecker Namespace

Copyright (C) 2003 Paul Welter
Spelling.IgnoreWord Method

Ignores the instances of the CurrentWord in the Text Property

[Visual Basic]
Public Sub IgnoreWord()

[C#]
public void IgnoreWord();

Remarks
Must call SpellCheck after call this method to resume spell checking

See Also
Spelling Class | NetSpell.SpellChecker Namespace

Copyright (C) 2003 Paul Welter
Spelling.OnDeletedWord Method

This is the method that is responsible for notifying receivers that the event occurred

[Visual Basic] Overridable Protected Sub OnDeletedWord(_
    ByVal e As SpellingEventArgs _
)

[C#]
protected virtual void OnDeletedWord(
    SpellingEventArgs e
);

See Also

Spelling Class | NetSpell.SpellChecker Namespace

Copyright (C) 2003 Paul Welter
This is the method that is responsible for notifying receivers that the event occurred

[Visual Basic]
Public Overridable Protected Sub OnDoubledWord(ByVal e As SpellingEventArgs)

[C#]
protected virtual void OnDoubledWord(SpellingEventArgs e);

See Also

Spelling Class | NetSpell.SpellChecker Namespace

Copyright (C) 2003 Paul Welter
Spelling.OnEndOfText Method

This is the method that is responsible for notifying receivers that the event occurred

[Visual Basic] Overridable Protected Sub OnEndOfText(
    ByVal e As EventArgs
)

[C#]
protected virtual void OnEndOfText(
    EventArgs e
);

See Also

Spelling Class | NetSpell.SpellChecker Namespace

Copyright (C) 2003 Paul Welter
**Spelling.OnIgnoredWord Method**

This is the method that is responsible for notifying receivers that the event occurred

```visualbasic
Overridable Protected Sub OnIgnoredWord(
    ByVal e As SpellingEventArgs)
```

```csharp
protected virtual void OnIgnoredWord(
    SpellingEventArgs e);
```

**See Also**

- [Spelling Class](#) | [NetSpell.SpellChecker Namespace](#)

*Copyright (C) 2003 Paul Welter*
Spelling.OnMisspelledWord Method

This is the method that is responsible for notifying receivers that the event occurred

[Visual Basic] Overridable Protected Sub OnMisspelledWord( ByVal e As SpellingEventArgs )

[C#] protected virtual void OnMisspelledWord( SpellingEventArgs e );

See Also

Spelling Class | NetSpell.SpellChecker Namespace

Copyright (C) 2003 Paul Welter
This is the method that is responsible for notifying receivers that the event occurred

```
[Visual Basic] Overridable Protected Sub OnReplacedWord( ByVal e As ReplaceWordEventArgs )
)
```

```
[C#]
protected virtual void OnReplacedWord( ReplaceWordEventArgs e );
```

See Also

Spelling Class | NetSpell.SpellChecker Namespace

Copyright (C) 2003 Paul Welter
Spelling.ReplaceAllWord Method

Replaces all instances of the CurrentWord in the Text Property

Overload List

Replaces all instances of the CurrentWord in the Text Property

public void ReplaceAllWord();

Replaces all instances of the CurrentWord in the Text Property

public void ReplaceAllWord(string);

See Also

Spelling Class | NetSpell.SpellChecker Namespace

Copyright (C) 2003 Paul Welter
Spelling.ReplaceAllWord Method ()

Replaces all instances of the CurrentWord in the Text Property

[Visual Basic] Overloads Public Sub ReplaceAllWord()

[C#] public void ReplaceAllWord();

See Also

Spelling Class | NetSpell.SpellChecker Namespace | Spelling.ReplaceAllWord Overload List

Copyright (C) 2003 Paul Welter
Spelling.ReplaceAllWord Method (String)

Replaces all instances of the CurrentWord in the Text Property

```csharp
public void ReplaceAllWord(string replacementWord);
```

Parameters

replacementWord

The word to replace the CurrentWord with

See Also

Spelling Class | NetSpell.SpellChecker Namespace | Spelling.ReplaceAllWord Overload List

Copyright (C) 2003 Paul Welter
Spelling.ReplaceWord Method

Replaces the instances of the CurrentWord in the Text Property

Overload List

Replaces the instances of the CurrentWord in the Text Property

public void ReplaceWord();

Replaces the instances of the CurrentWord in the Text Property

public void ReplaceWord(string);

See Also

Spelling Class | NetSpell.SpellChecker Namespace

Copyright (C) 2003 Paul Welter
Replaces the instances of the CurrentWord in the Text Property

**[Visual Basic]**

Overloads Public Sub ReplaceWord()

**[C#]**

public void ReplaceWord();

See Also

Spelling Class | NetSpell.SpellChecker Namespace | Spelling.ReplaceWord Overload List

Copyright (C) 2003 Paul Welter
Spelling.ReplaceWord Method (String)

Replaces the instances of the CurrentWord in the Text Property

[Visual Basic] Overloads Public Sub ReplaceWord
ByVal replacementWord As String
"

[C#]
public void ReplaceWord(
    string replacementWord
    );

Parameters

replacementWord

The word to replace the CurrentWord with

See Also

Spelling Class | NetSpell.SpellChecker Namespace | Spelling.ReplaceWord Overload List

Copyright (C) 2003 Paul Welter
Spelling.SpellCheck Method

Spell checks the words in the **Text** property starting at the **WordIndex** position.

**Overload List**

Spell checks the words in the **Text** property starting at the **WordIndex** position.

```csharp
public bool SpellCheck();
```

Spell checks the words in the **Text** property starting at the **WordIndex** position. This overload takes in the **WordIndex** to start checking from.

```csharp
public bool SpellCheck(int);
```

Spell checks a range of words in the **Text** property starting at the **WordIndex** position and ending at **endWordIndex**.

```csharp
public bool SpellCheck(int,int);
```

Spell checks the words in the **Text** property starting at the **WordIndex** position. This overload takes in the **text** to spell check

```csharp
public bool SpellCheck(string);
```

Spell checks the words in the **Text** property starting at the **WordIndex** position. This overload takes in the **text** to check and the **WordIndex** to start checking from.

```csharp
public bool SpellCheck(string,int);
```

**See Also**

<table>
<thead>
<tr>
<th>Spelling Class</th>
<th>NetSpell.SpellChecker Namespace</th>
<th>CurrentWord</th>
<th>WordIndex</th>
</tr>
</thead>
</table>

---

Copyright (C) 2003 Paul Welter
**Spelling.SpellCheck Method ()**

Spell checks the words in the Text property starting at the WordIndex position.

[C#]

```csharp
public bool SpellCheck();
```

**Return Value**

Returns true if there is a word found in the text that is not in the dictionaries

**See Also**

[Spelling Class] | [NetSpell.SpellChecker Namespace] | [Spelling.SpellCheck Overload List] | [CurrentWord] | [WordIndex]

---

Copyright (C) 2003 Paul Welter
Spelling.SpellCheck Method (Int32)

Spell checks the words in the Text property starting at the WordIndex position. This overload takes in the WordIndex to start checking from.

[Visual Basic] Overloads Public Function SpellCheck(
    ByVal startWordIndex As Integer _
) As Boolean

[C#] public bool SpellCheck(
    int startWordIndex
);

Parameters

startWordIndex

The index of the word to start checking from.

Return Value

Returns true if there is a word found in the text that is not in the dictionaries

See Also

Spelling Class | NetSpell.SpellChecker Namespace | Spelling.SpellCheck Overload List | CurrentWord | WordIndex

Copyright (C) 2003 Paul Welter
Spelling.SpellCheck Method (Int32, Int32)

Spell checks a range of words in the Text property starting at the WordIndex position and ending at endWordIndex.

[Visual Basic] Overloads Public Function SpellCheck(_
    ByVal startWordIndex As Integer, _
    ByVal endWordIndex As Integer _
) As Boolean

[C#] public bool SpellCheck(
    int startWordIndex, _
    int endWordIndex
);

Parameters

startWordIndex

The index of the word to start checking from.

endWordIndex

The index of the word to end checking with.

Return Value

Returns true if there is a word found in the text that is not in the dictionaries

See Also

Spelling Class | NetSpell.SpellChecker Namespace | Spelling.SpellCheck Overload List | CurrentWord | WordIndex

Copyright (C) 2003 Paul Welter
Spelling.SpellCheck Method (String)

Spell checks the words in the Text property starting at the WordIndex position. This overload takes in the text to spell check.

```visual-basic
Overloads Public Function SpellCheck(
    ByVal text As String
) As Boolean
```

```csharp
public bool SpellCheck(
    string text
);```

Parameters

- **text**
  - The text to spell check

Return Value

- Returns true if there is a word found in the text that is not in the dictionaries

See Also

- Spelling Class | NetSpell.SpellChecker Namespace | Spelling.SpellCheck Overload List | CurrentWord | WordIndex

Copyright (C) 2003 Paul Welter
Spell checks the words in the Text property starting at the WordIndex position. This overload takes in the text to check and the WordIndex to start checking from.

**Parameters**

- **text**
  
  The text to spell check

- **startWordIndex**
  
  The index of the word to start checking from

**Return Value**

Returns true if there is a word found in the text that is not in the dictionaries

**See Also**

- Spelling Class
- NetSpell.SpellChecker Namespace
- Spelling.SpellCheck Overload List
- CurrentWord
- WordIndex

Copyright (C) 2003 Paul Welter
Spelling.Suggest Method

Populates the Suggestions property with word suggestions for the CurrentWord

Overload List

Populates the Suggestions property with word suggestions for the CurrentWord

    public void Suggest();

Populates the Suggestions property with word suggestions for the word

    public void Suggest(string);

See Also

Spelling Class | NetSpell.SpellChecker Namespace | CurrentWord | Suggestions | TestWord

------------------------------------------------------------------------

Copyright (C) 2003 Paul Welter
Spelling.Suggest Method ()

Populates the Suggestions property with word suggestions for the CurrentWord.

[Visual Basic] Overloads Public Sub Suggest()

[C#] public void Suggest();

Remarks

TestWord must have been called before calling this method.

See Also

Spelling Class | NetSpell.SpellChecker Namespace | Spelling.Suggest Overload List | CurrentWord | Suggestions | TestWord

Copyright (C) 2003 Paul Welter
Populates the Suggestions property with word suggestions for the word.

[Visual Basic]
Overloads Public Sub Suggest(
    ByVal word As String
)

[C#]
public void Suggest(
    string word
);

Parameters

word
The word to generate suggestions on

Remarks

This method sets the Text property to the word. Then calls TestWord on the word to generate the need information for suggestions. Note that the Text, CurrentWord and WordIndex properties are set when calling this method.

See Also

Spelling Class | NetSpell.SpellChecker Namespace | Spelling.Suggest Overload List | CurrentWord | Suggestions | TestWord

Copyright (C) 2003 Paul Welter
Spelling.TestWord Method

Checks to see if the word is in the dictionary

[Visual Basic] Public Function TestWord(_
    ByVal word As String _
) As Boolean

[C#] public bool TestWord(_
    string word
);

Parameters

word

The word to check

Return Value

Returns true if word is found in dictionary

See Also

Spelling Class | NetSpell.SpellChecker Namespace

Copyright (C) 2003 Paul Welter
Spelling Events

The events of the **Spelling** class are listed below. For a complete list of **Spelling** class members, see the **Spelling Members** topic.

Public Instance Events

<table>
<thead>
<tr>
<th>Event</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DeletedWord</td>
<td>This event is fired when a word is deleted</td>
</tr>
<tr>
<td>Disposed (inherited from Component)</td>
<td>Adds an event handler to listen to the <strong>Disposed</strong> event on the component.</td>
</tr>
<tr>
<td>DoubledWord</td>
<td>This event is fired when word is detected two times in a row</td>
</tr>
<tr>
<td>EndOfText</td>
<td>This event is fired when the spell checker reaches the end of the text in the Text property</td>
</tr>
<tr>
<td>IgnoredWord</td>
<td>This event is fired when a word is skipped</td>
</tr>
<tr>
<td>MisspelledWord</td>
<td>This event is fired when the spell checker finds a word that is not in the dictionaries</td>
</tr>
<tr>
<td>ReplacedWord</td>
<td>This event is fired when a word is replace</td>
</tr>
</tbody>
</table>

See Also

[Spelling Class](#) | [NetSpell.SpellChecker Namespace](#)

---

Copyright (C) 2003 Paul Welter
**Spelling.DeletedWord Event**

This event is fired when a word is deleted

[Visual Basic]  
```
Public Event DeletedWord As DeletedWordEventHandler
```

[C#]  
```
public event DeletedWordEventHandler DeletedWord;
```

**Event Data**

The event handler receives an argument of type `SpellingEventArgs` containing data related to this event. The following `SpellingEventArgs` properties provide information specific to this event.

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>TextIndex</code></td>
<td>Text index of the WordEvent</td>
</tr>
<tr>
<td><code>Word</code></td>
<td>Word that caused the WordEvent</td>
</tr>
<tr>
<td><code>WordIndex</code></td>
<td>Word index of the WordEvent</td>
</tr>
</tbody>
</table>

**Remarks**

Use this event to update the parent text

**See Also**

- [Spelling Class](#)
- [NetSpell.SpellChecker Namespace](#)

*Copyright (C) 2003 Paul Welter*
Spelling.DoubledWord Event

This event is fired when word is detected two times in a row

[Visual Basic] Public Event DoubledWord As DoubledWordEventHandler

[C#] public event DoubledWordEventHandler DoubledWord;

Event Data

The event handler receives an argument of type SpellingEventArgs containing data related to this event. The following SpellingEventArgs properties provide information specific to this event.

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TextIndex</td>
<td>Text index of the WordEvent</td>
</tr>
<tr>
<td>Word</td>
<td>Word that caused the WordEvent</td>
</tr>
<tr>
<td>WordIndex</td>
<td>Word index of the WordEvent</td>
</tr>
</tbody>
</table>

See Also

Spelling Class | NetSpell.SpellChecker Namespace

Copyright (C) 2003 Paul Welter
Spelling.EndOfText Event

This event is fired when the spell checker reaches the end of the text in the Text property.

[Visual Basic] Public Event EndOfText As EndOfTextEventHandler

[C#] public event EndOfTextEventHandler EndOfText;

See Also

Spelling Class | NetSpell.SpellChecker Namespace

Copyright (C) 2003 Paul Welter
Spelling.IgnoredWord Event

This event is fired when a word is skipped.

[Visual Basic] Public Event IgnoredWord As IgnoredWordEventHandler

[C#]
public event IgnoredWordEventHandler IgnoredWord;

Event Data

The event handler receives an argument of type SpellingEventArgs containing data related to this event. The following SpellingEventArgs properties provide information specific to this event.

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TextIndex</td>
<td>Text index of the WordEvent</td>
</tr>
<tr>
<td>Word</td>
<td>Word that caused the WordEvent</td>
</tr>
<tr>
<td>WordIndex</td>
<td>Word index of the WordEvent</td>
</tr>
</tbody>
</table>

See Also

Spelling Class | NetSpell.SpellChecker Namespace

Copyright (C) 2003 Paul Welter
Spelling.MisspelledWord Event

This event is fired when the spell checker finds a word that is not in the dictionaries

[Visual Basic] Public Event MisspelledWord As

[C#] public event MisspelledWordEventHandler MisspelledWord;

Event Data

The event handler receives an argument of type SpellingEventArgs containing data related to this event. The following SpellingEventArgs properties provide information specific to this event.

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TextIndex</td>
<td>Text index of the WordEvent</td>
</tr>
<tr>
<td>Word</td>
<td>Word that caused the WordEvent</td>
</tr>
<tr>
<td>WordIndex</td>
<td>Word index of the WordEvent</td>
</tr>
</tbody>
</table>

See Also

Spelling Class | NetSpell.SpellChecker Namespace

Copyright (C) 2003 Paul Welter
Spelling.ReplacedWord Event

This event is fired when a word is replace

[Visual Basic] Public Event ReplacedWord As ReplacedWordEventHandler

[C#] public event ReplacedWordEventHandler ReplacedWord;

Event Data

The event handler receives an argument of type ReplaceWordEventArgs containing data related to this event. The following ReplaceWordEventArgs properties provide information specific to this event.

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ReplacementWord</td>
<td>The word to use in replacing the misspelled word</td>
</tr>
<tr>
<td>TextIndex</td>
<td>Text index of the WordEvent</td>
</tr>
<tr>
<td>Word</td>
<td>Word that caused the WordEvent</td>
</tr>
<tr>
<td>WordIndex</td>
<td>Word index of the WordEvent</td>
</tr>
</tbody>
</table>

Remarks

Use this event to update the parent text

See Also

Spelling Class | NetSpell.SpellChecker Namespace

Copyright (C) 2003 Paul Welter
Spelling.DeletedWordEventHandler Delegate

This represents the delegate method prototype that event receivers must implement

[Visual Basic]
Public Delegate Sub Spelling.DeletedWordEventHandler(
  ByVal sender As Object,
  ByVal e As SpellingEventArgs
)

[C#]
public delegate void Spelling.DeletedWordEventHandler(
  object sender,
  SpellingEventArgs e
);

Requirements
Namespace: NetSpell.SpellChecker
Assembly: NetSpell.SpellChecker (in NetSpell.SpellChecker.dll)

See Also
NetSpell.SpellChecker Namespace

Copyright (C) 2003 Paul Welter
Spelling.DoubledWordEventHandler Delegate

This represents the delegate method prototype that event receivers must implement

```vbnet
Public Delegate Sub Spelling.DoubledWordEventHandler(
    ByVal sender As Object,
    ByVal e As SpellingEventArgs)
```

```csharp
public delegate void Spelling.DoubledWordEventHandler(
    object sender, SpellingEventArgs e);
```

Requirements

- **Namespace**: [NetSpell.SpellChecker](#)
- **Assembly**: NetSpell.SpellChecker (in NetSpell.SpellChecker.dll)

See Also

- [NetSpell.SpellChecker Namespace](#)

Copyright (C) 2003 Paul Welter
Spelling.EndOfTextEventHandler Delegate

This represents the delegate method prototype that event receivers must implement

```[Visual Basic]Public Delegate Sub Spelling.EndOfTextEventHandler(  ByVal sender As Object, ByVal e As EventArgs)_()
```

```[C#]public delegate void Spelling.EndOfTextEventHandler(object sender, EventArgs e);
```

Requirements

Namespace: NetSpell.SpellChecker

Assembly: NetSpell.SpellChecker (in NetSpell.SpellChecker.dll)

See Also

NetSpell.SpellChecker Namespace

Copyright (C) 2003 Paul Welter
Spelling.IgnoredWordEventHandler Delegate

This represents the delegate method prototype that event receivers must implement

[Visual Basic]
Public Delegate Sub Spelling.IgnoredWordEventHandler(
    ByVal sender As Object, _
    ByVal e As SpellingEventArgs _
)

[C#]
public delegate void Spelling.IgnoredWordEventHandler(
    object sender,
    SpellingEventArgs e
);

Requirements

Namespace: NetSpell.SpellChecker

Assembly: NetSpell.SpellChecker (in NetSpell.SpellChecker.dll)

See Also

NetSpell.SpellChecker Namespace

Copyright (C) 2003 Paul Welter
Spelling.MisspelledWordEventHandler Delegate

This represents the delegate method prototype that event receivers must implement

[Visual Basic] Public Delegate Sub Spelling.MisspelledWordEventHandler(_ ByVal sender As Object, ByVal e As SpellingEventArgs _)

[C#] public delegate void Spelling.MisspelledWordEventHandler(object sender, SpellingEventArgs e);

Requirements

Namespace: NetSpell.SpellChecker

Assembly: NetSpell.SpellChecker (in NetSpell.SpellChecker.dll)

See Also

NetSpell.SpellChecker Namespace

Copyright (C) 2003 Paul Welter
Spelling.ReplacedWordEventHandler Delegate

This represents the delegate method prototype that event receivers must implement

[Visual Basic]
Public Delegate Sub Spelling.ReplacedWordEventHandler(_
   ByVal sender As Object, _
   ByVal e As ReplaceWordEventArgs _
)

[C#]
public delegate void Spelling.ReplacedWordEventHandler(_
   object sender, 
   ReplaceWordEventArgs e
);

Requirements

Namespace: NetSpell.SpellChecker

Assembly: NetSpell.SpellChecker (in NetSpell.SpellChecker.dll)

See Also

NetSpell.SpellChecker Namespace

Copyright (C) 2003 Paul Welter
Spelling.SuggestionEnum Enumeration

The suggestion strategy to use when generating suggestions

[Visual Basic]Public Enum Spelling.SuggestionEnum

[C#]
public enum Spelling.SuggestionEnum

Members

<table>
<thead>
<tr>
<th>Member Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PhoneticNearMiss</td>
<td>Combines the phonetic and near miss strategies</td>
</tr>
<tr>
<td>Phonetic</td>
<td>The phonetic strategy generates suggestions by word sound</td>
</tr>
<tr>
<td>NearMiss</td>
<td>The near miss strategy generates suggestion by replacing, removing, adding chars to make words</td>
</tr>
</tbody>
</table>

Requirements

Namespace: NetSpell.SpellChecker

Assembly: NetSpell.SpellChecker (in NetSpell.SpellChecker.dll)

See Also

NetSpell.SpellChecker Namespace

Copyright (C) 2003 Paul Welter
SpellingEventArgs Class

Class sent to the event handler when the DoubleWord or MisspelledWord event occurs

For a list of all members of this type, see SpellingEventArgs Members.

System.Object  EventArgs
  SpellingEventArgs

[Visual Basic]
Public Class SpellingEventArgs
Inherits EventArgs

[C#]
public class SpellingEventArgs : EventArgs

Requirements

Namespace: NetSpell.SpellChecker

Assembly: NetSpell.SpellChecker (in NetSpell.SpellChecker.dll)

See Also

SpellingEventArgs Members  |  NetSpell.SpellChecker Namespace

Copyright (C) 2003 Paul Welter
### SpellingEventArgs Members

<table>
<thead>
<tr>
<th>SpellingEventArgs overview</th>
</tr>
</thead>
</table>

#### Public Instance Constructors

<table>
<thead>
<tr>
<th>Constructor</th>
<th>Constructor used to pass in properties</th>
</tr>
</thead>
</table>

#### Public Instance Properties

<table>
<thead>
<tr>
<th>TextIndex</th>
<th>Text index of the WordEvent</th>
</tr>
</thead>
<tbody>
<tr>
<td>Word</td>
<td>Word that caused the WordEvent</td>
</tr>
<tr>
<td>WordIndex</td>
<td>Word index of the WordEvent</td>
</tr>
</tbody>
</table>

#### Public Instance Methods

| Equals (inherited from Object) | Determines whether the specified Object is equal to the current Object. |
| GetHashCode (inherited from Object) | Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table. |
| GetType (inherited from Object) | Gets the Type of the current instance. |
| ToString (inherited from Object) | Returns a String that represents the current Object. |

#### Protected Instance Methods

| Finalize (inherited from Object) | Allows an Object to attempt to free resources and perform other cleanup operations before the Object is reclaimed by garbage collection. |
| **MemberwiseClone** (inherited from **Object**) | Creates a shallow copy of the current **Object**. |

See Also

- **SpellingEventArgs Class** | **NetSpell.SpellChecker Namespace**

**Copyright (C) 2003 Paul Welter**
SpellingEventArgs Constructor

Constructor used to pass in properties

[Visual Basic]
Public Sub New( _
  ByVal word As String, _
  ByVal wordIndex As Integer, _
  ByVal textIndex As Integer _
)

[C#]
public SpellingEventArgs(
  string word,
  int wordIndex,
  int textIndex
);

See Also

SpellingEventArgs Class | NetSpell.SpellChecker Namespace

Copyright (C) 2003 Paul Welter
SpellingEventArgs Properties

The properties of the **SpellingEventArgs** class are listed below. For a complete list of **SpellingEventArgs** class members, see the **SpellingEventArgs Members** topic.

**Public Instance Properties**

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>TextIndex</strong></td>
<td>Text index of the WordEvent</td>
</tr>
<tr>
<td><strong>Word</strong></td>
<td>Word that caused the WordEvent</td>
</tr>
<tr>
<td><strong>WordIndex</strong></td>
<td>Word index of the WordEvent</td>
</tr>
</tbody>
</table>

**See Also**

[SpellingEventArgs Class](#) | [NetSpell.SpellChecker Namespace](#)

Copyright (C) 2003 Paul Welter
SpellingEventArgs.TextIndex Property

Text index of the WordEvent

[Visual Basic] Public ReadOnly Property TextIndex As Integer

[C#]
public int TextIndex {get;}

See Also

SpellingEventArgs Class | NetSpell.SpellChecker Namespace

Copyright (C) 2003 Paul Welter
SpellingEventArgs.Word Property

Word that caused the WordEvent

[Visual Basic]
Public ReadOnly Property Word As String

[C#]
public string Word {get;}

See Also

SpellingEventArgs Class | NetSpell.SpellChecker Namespace

Copyright (C) 2003 Paul Welter
SpellingEventArgs.WordIndex Property

Word index of the WordEvent

[Visual Basic] Public ReadOnly Property WordIndex

[C#] public int WordIndex {get;}

See Also

SpellingEventArgs Class | NetSpell.SpellChecker Namespace

Copyright (C) 2003 Paul Welter
NetSpell.SpellChecker.Dictionary Namespace

Classes

<table>
<thead>
<tr>
<th>Class</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Word</strong></td>
<td>The Word class represents a base word in the dictionary</td>
</tr>
<tr>
<td><strong>WordDictionary</strong></td>
<td>The WordDictionary class contains all the logic for managing the word list.</td>
</tr>
</tbody>
</table>
Word Class

The Word class represents a base word in the dictionary.

For a list of all members of this type, see Word Members.

System.Object  Word

[Visual Basic]
Public Class Word
Implements IComparable

[C#]
public class Word : IComparable

Requirements


Assembly: NetSpell.SpellChecker (in NetSpell.SpellChecker.dll)

See Also

Word Members | NetSpell.SpellChecker.Dictionary Namespace

Copyright (C) 2003 Paul Welter
Word Members

**Word overview**

**Public Instance Constructors**

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>Word</code></td>
<td>Overloaded. Initiates a new instance of the Word class.</td>
</tr>
</tbody>
</table>

**Public Instance Properties**

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>AffixKeys</code></td>
<td>The affix keys that can be applied to this base word</td>
</tr>
<tr>
<td><code>Index</code></td>
<td>The index position of where this word appears</td>
</tr>
<tr>
<td><code>PhoneticCode</code></td>
<td>The phonetic code for this word</td>
</tr>
<tr>
<td><code>Text</code></td>
<td>The string for the base word</td>
</tr>
</tbody>
</table>

**Public Instance Methods**

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>CompareTo</code></td>
<td>Sorts a collection of words by EditDistance</td>
</tr>
<tr>
<td><code>Equals</code> (inherited from <code>Object</code>)</td>
<td>Determines whether the specified <code>Object</code> is equal to the current <code>Object</code>.</td>
</tr>
<tr>
<td><code>GetHashCode</code> (inherited from <code>Object</code>)</td>
<td>Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table.</td>
</tr>
<tr>
<td><code>GetType</code> (inherited from <code>Object</code>)</td>
<td>Gets the <code>Type</code> of the current instance.</td>
</tr>
<tr>
<td><code>ToString</code></td>
<td>Converts the word object to a string</td>
</tr>
</tbody>
</table>

**Protected Instance Methods**

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>Finalize</code> (inherited from <code>Object</code>)</td>
<td>Allows an <code>Object</code> to attempt to</td>
</tr>
<tr>
<td><strong>Object</strong></td>
<td>free resources and perform other cleanup operations before the <strong>Object</strong> is reclaimed by garbage collection.</td>
</tr>
<tr>
<td><strong>MemberwiseClone</strong> (inherited from <strong>Object</strong>)</td>
<td>Creates a shallow copy of the current <strong>Object</strong>.</td>
</tr>
</tbody>
</table>

**See Also**


**Copyright (C) 2003 Paul Welter**
**Word Constructor**

Initializes a new instance of the class

**Overload List**

Initializes a new instance of the class

```csharp
public Word();
```

Initializes a new instance of the class

```csharp
public Word(string);
```

Initializes a new instance of the class

```csharp
public Word(string,string);
```

Initializes a new instance of the class

```csharp
public Word(string,string,string);
```

**See Also**


Copyright (C) 2003 Paul Welter
**Word Constructor ()**

Initializes a new instance of the class

[Visual Basic] *Overloads Public Sub New()*

[C#]  
*public Word();*

See Also


---

Copyright (C) 2003 Paul Welter
Word Constructor (String, String, String)

Initializes a new instance of the class

[Visual Basic] Overloads Public Sub New( _
    ByVal text As String, _
    ByVal affixKeys As String, _
    ByVal phoneticCode As String _
)

[C#]
public Word(
    string text,
    string affixKeys,
    string phoneticCode
);

Parameters

text
The string for the base word

affixKeys
The affix keys that can be applied to this base word

phoneticCode
The phonetic code for this word

See Also


Copyright (C) 2003 Paul Welter
Word Constructor (String, String)

Initializes a new instance of the class

**[Visual Basic]**

```vbnet
Overloads Public Sub New(_
    ByVal text As String, _
    ByVal affixKeys As String _
)
```

**[C#]**

```csharp
public Word(
    string text,
    string affixKeys
);
```

**Parameters**

*text*

The string for the base word

*affixKeys*

The affix keys that can be applied to this base word

**See Also**


---

Copyright (C) 2003 Paul Welter
Word Constructor (String)

Initializes a new instance of the class

**[Visual Basic]**

```
Overloads Public Sub New(_
    ByVal text As String _
)
```

**[C#]**

```
public Word(
    string text
);
```

**Parameters**

*text*

The string for the base word

**See Also**


---

[Copyright (C) 2003 Paul Welter](#)
The properties of the **Word** class are listed below. For a complete list of **Word** class members, see the [Word Members](#) topic.

### Public Instance Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>AffixKeys</strong></td>
<td>The affix keys that can be applied to this base word</td>
</tr>
<tr>
<td><strong>Index</strong></td>
<td>The index position of where this word appears</td>
</tr>
<tr>
<td><strong>PhoneticCode</strong></td>
<td>The phonetic code for this word</td>
</tr>
<tr>
<td><strong>Text</strong></td>
<td>The string for the base word</td>
</tr>
</tbody>
</table>

### See Also


---

Copyright (C) 2003 Paul Welter
Word.AffixKeys Property

The affix keys that can be applied to this base word

[Visual Basic] Public Property AffixKeys As String

[C#]
public string AffixKeys {get; set;}

See Also

Word Class | NetSpell.SpellChecker.Dictionary Namespace

Copyright (C) 2003 Paul Welter
Word.Index Property

The index position of where this word appears

[Visual Basic] Public Property Index As Integer

[C#]
public int Index {get; set;}

See Also

Word Class | NetSpell.SpellChecker.Dictionary Namespace

Copyright (C) 2003 Paul Welter
The phonetic code for this word

[Visual Basic] Public Property PhoneticCode As

[C#]
public string PhoneticCode {get; set;}

See Also

Word Class | NetSpell.SpellChecker.Dictionary Namespace

Copyright (C) 2003 Paul Welter
Word.Text Property

The string for the base word

[Visual Basic] Public Property Text As String

[C#]
public string Text {get; set;}

See Also

Word Class | NetSpell.SpellChecker.Dictionary Namespace

Copyright (C) 2003 Paul Welter
The methods of the **Word** class are listed below. For a complete list of **Word** class members, see the [Word Members topic](#).

### Public Instance Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>CompareTo</strong></td>
<td>Sorts a collection of words by EditDistance</td>
</tr>
<tr>
<td><strong>Equals</strong> (inherited from <strong>Object</strong>)</td>
<td>Determines whether the specified <strong>Object</strong> is equal to the current <strong>Object</strong>.</td>
</tr>
<tr>
<td><strong>GetHashCode</strong> (inherited from <strong>Object</strong>)</td>
<td>Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table.</td>
</tr>
<tr>
<td><strong>GetType</strong> (inherited from <strong>Object</strong>)</td>
<td>Gets the <strong>Type</strong> of the current instance.</td>
</tr>
<tr>
<td><strong>ToString</strong></td>
<td>Converts the word object to a string</td>
</tr>
</tbody>
</table>

### Protected Instance Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Finalize</strong> (inherited from <strong>Object</strong>)</td>
<td>Allows an <strong>Object</strong> to attempt to free resources and perform other cleanup operations before the <strong>Object</strong> is reclaimed by garbage collection.</td>
</tr>
<tr>
<td><strong>MemberwiseClone</strong> (inherited from <strong>Object</strong>)</td>
<td>Creates a shallow copy of the current <strong>Object</strong>.</td>
</tr>
</tbody>
</table>

### See Also

Word.CompareTo Method

Sorts a collection of words by EditDistance

[Visual Basic] NotOverridable Public Function CompareTo(
    ByVal obj As Object
) As Integer Implements IComparable.CompareTo

[C#]
public int CompareTo(
    object obj
);

Implements
    IComparable.CompareTo

Remarks
    The compare sorts in desc order, largest EditDistance first

See Also
    Word Class | NetSpell.SpellChecker.Dictionary Namespace

Copyright (C) 2003 Paul Welter
Word.ToString Method

Converts the word object to a string

[Visual Basic] Overrides Public Function ToString()

[C#]
public override string ToString();

Return Value

Returns the Text Property contents

See Also

Word Class | NetSpell.SpellChecker.Dictionary Namespace

Copyright (C) 2003 Paul Welter
The WordDictionary class contains all the logic for managing the word list.

For a list of all members of this type, see WordDictionary Members.

System.Object  MarshalByRefObject  Component  
WordDictionary

[Visual Basic]
Public Class WordDictionary 
Inherits Component

[C#]
public class WordDictionary : Component

Requirements

Assembly: NetSpell.SpellChecker (in NetSpell.SpellChecker.dll)

See Also


Copyright (C) 2003 Paul Welter
## WordDictionary Members

### WordDictionary overview

### Public Instance Constructors

<table>
<thead>
<tr>
<th>Constructor</th>
<th>Description</th>
</tr>
</thead>
</table>

### Public Instance Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BaseWords</td>
<td>The collection of base words for the dictionary</td>
</tr>
<tr>
<td>Container</td>
<td>Gets the IContainer that contains the Component.</td>
</tr>
<tr>
<td>Copyright</td>
<td>Copyright text for the dictionary</td>
</tr>
<tr>
<td>DictionaryFile</td>
<td>The file name for the main dictionary</td>
</tr>
<tr>
<td>DictionaryFolder</td>
<td>Folder containing the dictionaries</td>
</tr>
<tr>
<td>EnableUserFile</td>
<td>Set this to true to automatically create a user dictionary when a word is added.</td>
</tr>
<tr>
<td>Initialized</td>
<td>True if the dictionary has been initialized</td>
</tr>
<tr>
<td>PhoneticRules</td>
<td>Collection of phonetic rules for this dictionary</td>
</tr>
<tr>
<td>PrefixRules</td>
<td>Collection of affix prefixes for the base words in this dictionary</td>
</tr>
<tr>
<td>ReplaceCharacters</td>
<td>List of characters to use when generating suggestions using the near miss strategy</td>
</tr>
<tr>
<td>Site</td>
<td>Gets or sets the ISite of the Component.</td>
</tr>
</tbody>
</table>
### SuffixRules
Collection of affix suffixes for the base words in this dictionary

### TryCharacters
List of characters to try when generating suggestions using the near miss strategy

### UserFile
The file name for the user word list for this dictionary

### UserWords
List of user entered words in this dictionary

---

**Public Instance Methods**

- **Add**
  Adds a word to the user list

- **Clear**
  Clears the user list of words

- **Contains**
  Searches all contained word lists for word

- **CreateObjRef** (inherited from MarshalByRefObject)
  Creates an object that contains all the relevant information required to generate a proxy used to communicate with a remote object.

- **Dispose** (inherited from Component)
  Overloaded. Releases all resources used by the Component.

- **Equals** (inherited from Object)
  Determines whether the specified Object is equal to the current Object.

- **ExpandWord**
  Expands an affix compressed base word

- **GetHashCode** (inherited from Object)
  Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table.

- **GetLifetimeService** (inherited)
  Retrieves the current lifetime
<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MarshalByRefObject</td>
<td>service object that controls the lifetime policy for this instance.</td>
</tr>
<tr>
<td>GetType (inherited from Object)</td>
<td>Gets the Type of the current instance.</td>
</tr>
<tr>
<td>Initialize</td>
<td>Initializes the dictionary by loading and parsing the dictionary file and the user file.</td>
</tr>
<tr>
<td>InitializeLifetimeService (inherited from MarshalByRefObject)</td>
<td>Obtains a lifetime service object to control the lifetime policy for this instance.</td>
</tr>
<tr>
<td>PhoneticCode</td>
<td>Generates a phonetic code of how the word sounds</td>
</tr>
<tr>
<td>Remove</td>
<td>Removes a word from the user list</td>
</tr>
<tr>
<td>ToString (inherited from Component)</td>
<td></td>
</tr>
</tbody>
</table>

**Public Instance Events**

<table>
<thead>
<tr>
<th>Event</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Disposed (inherited from Component)</td>
<td>Adds an event handler to listen to the Disposed event on the component.</td>
</tr>
</tbody>
</table>

**Protected Instance Properties**

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DesignMode (inherited from Component)</td>
<td>Gets a value that indicates whether the Component is currently in design mode.</td>
</tr>
<tr>
<td>Events (inherited from Component)</td>
<td>Gets the list of event handlers that are attached to this Component.</td>
</tr>
</tbody>
</table>

**Protected Instance Methods**

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dispose</td>
<td>Overloaded. Clean up any resources being used.</td>
</tr>
<tr>
<td><strong>Finalize</strong> (inherited from Component)</td>
<td>Releases unmanaged resources and performs other cleanup operations before the <strong>Component</strong> is reclaimed by garbage collection.</td>
</tr>
<tr>
<td><strong>GetService</strong> (inherited from Component)</td>
<td>Returns an object that represents a service provided by the <strong>Component</strong> or by its <strong>Container</strong>.</td>
</tr>
<tr>
<td><strong>MemberwiseClone</strong> (inherited from Object)</td>
<td>Creates a shallow copy of the current <strong>Object</strong>.</td>
</tr>
</tbody>
</table>

**See Also**


[Copyright (C) 2003 Paul Welter]
**WordDictionary Constructor**

Initializes a new instance of the class

**Overload List**

Initializes a new instance of the class

```csharp
public WordDictionary();
```

Initializes a new instance of the class

```csharp
public WordDictionary(IContainer);
```

**See Also**


---

[Copyright (C) 2003 Paul Welter]
**WordDictionary Constructor ()**

Initializes a new instance of the class

[Visual Basic] Overloads Public Sub New()

[C#]
public WordDictionary();

See Also


Copyright (C) 2003 Paul Welter
WordDictionary Constructor (IContainer)

Initializes a new instance of the class

[C#]
public WordDictionary(
    IContainer container
);

[Visual Basic]
Overloads Public Sub New( _
    ByVal container As IContainer _
)

See Also

Copyright (C) 2003 Paul Welter
The properties of the **WordDictionary** class are listed below. For a complete list of **WordDictionary** class members, see the **WordDictionary Members** topic.

## Public Instance Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>BaseWords</strong></td>
<td>The collection of base words for the dictionary</td>
</tr>
<tr>
<td><strong>Container</strong></td>
<td>(inherited from Component) Gets the <strong>IContainer</strong> that contains the <strong>Component</strong>.</td>
</tr>
<tr>
<td><strong>Copyright</strong></td>
<td>Copyright text for the dictionary</td>
</tr>
<tr>
<td><strong>DictionaryFile</strong></td>
<td>The file name for the main dictionary</td>
</tr>
<tr>
<td><strong>DictionaryFolder</strong></td>
<td>Folder containing the dictionaries</td>
</tr>
<tr>
<td><strong>EnableUserFile</strong></td>
<td>Set this to true to automatically create a user dictionary when a word is added.</td>
</tr>
<tr>
<td><strong>Initialized</strong></td>
<td>True if the dictionary has been initialized</td>
</tr>
<tr>
<td><strong>PhoneticRules</strong></td>
<td>Collection of phonetic rules for this dictionary</td>
</tr>
<tr>
<td><strong>PrefixRules</strong></td>
<td>Collection of affix prefixes for the base words in this dictionary</td>
</tr>
<tr>
<td><strong>ReplaceCharacters</strong></td>
<td>List of characters to use when generating suggestions using the near miss strategy</td>
</tr>
<tr>
<td><strong>Site</strong></td>
<td>(inherited from Component) Gets or sets the <strong>ISite</strong> of the <strong>Component</strong>.</td>
</tr>
<tr>
<td><strong>SuffixRules</strong></td>
<td>Collection of affix suffixes for the base words in this dictionary</td>
</tr>
<tr>
<td>Property</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td><strong>TryCharacters</strong></td>
<td>List of characters to try when generating suggestions using the near miss strategy</td>
</tr>
<tr>
<td><strong>UserFile</strong></td>
<td>The file name for the user word list for this dictionary</td>
</tr>
<tr>
<td><strong>UserWords</strong></td>
<td>List of user entered words in this dictionary</td>
</tr>
</tbody>
</table>

**Protected Instance Properties**

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>DesignMode</strong></td>
<td>(inherited from Component) Gets a value that indicates whether the Component is currently in design mode.</td>
</tr>
<tr>
<td><strong>Events</strong></td>
<td>(inherited from Component) Gets the list of event handlers that are attached to this Component.</td>
</tr>
</tbody>
</table>

**See Also**


---

Copyright (C) 2003 Paul Welter
**WordDictionary.BaseWords Property**

The collection of base words for the dictionary

**[Visual Basic]**
```
Public ReadOnly Property BaseWords
```

**[C#]**
```
public System.Collections.Hashtable BaseWords
```

See Also


**Copyright (C) 2003 Paul Welter**
WordDictionary.Copyright Property

Copyright text for the dictionary

[Visual Basic] Public Property Copyright As String

[C#]

class String Copyright {get; set;}

See Also


Copyright (C) 2003 Paul Welter
WordDictionary.DictionaryFile Property

The file name for the main dictionary

[Visual Basic] Public Property DictionaryFile

[C#] public string DictionaryFile {get; set;}

See Also


Copyright (C) 2003 Paul Welter
**WordDictionary.DictionaryFolder Property**

Folder containing the dictionaries

[Visual Basic] `Public Property DictionaryFolder As [Visual Basic]`  
[C#] `public string DictionaryFolder {get; set;}`

See Also


Copyright (C) 2003 Paul Welter
**WordDictionary.EnableUserFile Property**

Set this to true to automatically create a user dictionary when a word is added.

[Visual Basic]

```vbnet
Public Property EnableUserFile
```

[C#]

```csharp
public bool EnableUserFile {get; set;}
```

**Remarks**

This should be set to false in a web environment

**See Also**


---

Copyright (C) 2003 Paul Welter
WordDictionary.Initialized Property

True if the dictionary has been initialized

[Visual Basic]
Public ReadOnly Property Initialized

[C#]
public bool Initialized {get;}

See Also


Copyright (C) 2003 Paul Welter
WordDictionary.PhoneticRules Property

Collection of phonetic rules for this dictionary

[Visual Basic] Public ReadOnly Property PhoneticRules As...

[C#]

public Phonetic.PhoneticRuleCollection PhoneticRules {get;}

See Also


Copyright (C) 2003 Paul Welter
**WordDictionary.PrefixRules Property**

Collection of affix prefixes for the base words in this dictionary

**[Visual Basic]**

Public ReadOnly Property PrefixRules

**[C#]**

public Affix.AffixRuleCollection PrefixRules

See Also


Copyright (C) 2003 Paul Welter
WordDictionary.ReplaceCharacters Property

List of characters to use when generating suggestions using the near miss strategy

[Visual Basic]
Public ReadOnly Property ReplaceCharacters as String

[C#]
public System.Collections.ArrayList ReplaceCharacters

See Also

Namespace

Copyright (C) 2003 Paul Welter
WordDictionary.SuffixRules Property

Collection of affix suffixes for the base words in this dictionary

[Visual Basic] Public Readonly Property SuffixRules

[C#]
public Affix.AffixRuleCollection SuffixRules

See Also


Copyright (C) 2003 Paul Welter
List of characters to try when generating suggestions using the near miss strategy

[Visual Basic] Public Property TryCharacters

[C#]
public string TryCharacters {get; set;}

See Also


Copyright (C) 2003 Paul Welter
WordDictionary.UserFile Property

The file name for the user word list for this dictionary

[Visual Basic] Public Property UserFile As String

[C#]
public string UserFile {get; set;}

See Also


Copyright (C) 2003 Paul Welter
**WordDictionary.UserWords Property**

List of user entered words in this dictionary

**[Visual Basic]**
```
Public ReadOnly Property UserWords
```

**[C#]**
```
public System.Collections.Hashtable UserWords
```

See Also


*Copyright (C) 2003 Paul Welter*
WordDictionary Methods

The methods of the **WordDictionary** class are listed below. For a complete list of **WordDictionary** class members, see the **WordDictionary Members** topic.

Public Instance Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Add</strong></td>
<td>Adds a word to the user list</td>
</tr>
<tr>
<td><strong>Clear</strong></td>
<td>Clears the user list of words</td>
</tr>
<tr>
<td><strong>Contains</strong></td>
<td>Searches all contained word lists for word</td>
</tr>
<tr>
<td><strong>CreateObjRef</strong></td>
<td>(inherited from <strong>MarshalByRefObject</strong>) Creates an object that contains all the relevant information required to generate a proxy used to communicate with a remote object.</td>
</tr>
<tr>
<td><strong>Dispose</strong></td>
<td>(inherited from <strong>Component</strong>) Overloaded. Releases all resources used by the <strong>Component</strong>.</td>
</tr>
<tr>
<td><strong>Equals</strong></td>
<td>(inherited from <strong>Object</strong>) Determines whether the specified <strong>Object</strong> is equal to the current <strong>Object</strong>.</td>
</tr>
<tr>
<td><strong>ExpandWord</strong></td>
<td>Expands an affix compressed base word</td>
</tr>
<tr>
<td><strong>GetHashCode</strong></td>
<td>(inherited from <strong>Object</strong>) Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table.</td>
</tr>
<tr>
<td><strong>GetLifetimeService</strong></td>
<td>(inherited from <strong>MarshalByRefObject</strong>) Retrieves the current lifetime service object that controls the lifetime policy for this instance.</td>
</tr>
<tr>
<td><strong>GetType</strong></td>
<td>(inherited from <strong>Object</strong>) Gets the <strong>Type</strong> of the current instance.</td>
</tr>
<tr>
<td><strong>Initialize</strong></td>
<td>Initializes the dictionary by loading and parsing the dictionary file and the user file.</td>
</tr>
<tr>
<td>---------------</td>
<td>------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td><strong>InitializeLifetimeService</strong> (\text{(inherited from MarshalByRefObject)})</td>
<td>Obtains a lifetime service object to control the lifetime policy for this instance.</td>
</tr>
<tr>
<td><strong>PhoneticCode</strong></td>
<td>Generates a phonetic code of how the word sounds</td>
</tr>
<tr>
<td><strong>Remove</strong></td>
<td>Removes a word from the user list</td>
</tr>
<tr>
<td><strong>ToString</strong> (\text{(inherited from Component)})</td>
<td></td>
</tr>
</tbody>
</table>

### Protected Instance Methods

<table>
<thead>
<tr>
<th><strong>Dispose</strong></th>
<th>Overloaded. Clean up any resources being used.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Finalize</strong> (\text{(inherited from Component)})</td>
<td>Releases unmanaged resources and performs other cleanup operations before the Component is reclaimed by garbage collection.</td>
</tr>
<tr>
<td><strong>GetService</strong> (\text{(inherited from Component)})</td>
<td>Returns an object that represents a service provided by the Component or by its Container.</td>
</tr>
<tr>
<td><strong>MemberwiseClone</strong> (\text{(inherited from Object)})</td>
<td>Creates a shallow copy of the current Object.</td>
</tr>
</tbody>
</table>

### See Also


Copyright (C) 2003 Paul Welter
WordDictionary.Add Method

Adds a word to the user list

[Visual Basic]
Public Sub Add(ByVal word As String)

[C#]
public void Add(string word);

Parameters

word

The word to add

Remarks

This method is only affects the user word list

See Also


Copyright (C) 2003 Paul Welter
WordDictionary.Clear Method

Clears the user list of words

[Visual Basic] Public Sub Clear()

[C#]
public void Clear();

Remarks
This method is only affects the user word list

See Also


Copyright (C) 2003 Paul Welter
WordDictionary.Contains Method

Searches all contained word lists for word

[Visual Basic] Public Function Contains( _
  ByVal word As String _
) As Boolean

[C#]
public bool Contains(
  string word
);

Parameters

word
The word to search for

Return Value

Returns true if word is found

See Also


Copyright (C) 2003 Paul Welter
WordDictionary.Dispose Method

Clean up any resources being used.

Overload List

Inherited from Component.

public void Dispose();

Clean up any resources being used.

protected override void Dispose(bool);

See Also


Copyright (C) 2003 Paul Welter
WordDictionary.Dispose Method (Boolean)

Clean up any resources being used.

[Visual Basic] Overrides Overloads Protected Sub Dispose(ByVal disposing As Boolean)

[C#]
protected override void Dispose(bool disposing);

See Also


Copyright (C) 2003 Paul Welter
**WordDictionary.ExpandWord Method**

Expands an affix compressed base word

```visual-basic
Public Function ExpandWord(
    ByVal word As Word
) As ArrayList
```

```csharp
public ArrayList ExpandWord(
    Word word
);
```

**Parameters**

*word*

The word to expand

**Return Value**

A System.Collections.ArrayList of words expanded from base word

**See Also**


---

Copyright (C) 2003 Paul Welter
WordDictionary.Initialize Method

Initializes the dictionary by loading and parsing the dictionary file and the user file.

[Visual Basic]
Public Sub Initialize()

[C#]
public void Initialize();

See Also


Copyright (C) 2003 Paul Welter
Generates a phonetic code of how the word sounds

[Visual Basic] Public Function PhoneticCode(_
    ByVal word As String _
) As String

[C#]
public string PhoneticCode(
    string word
);

Parameters

word
The word to generated the sound code from

Return Value

A code of how the word sounds

See Also


Copyright (C) 2003 Paul Welter
**WordDictionary.Remove Method**

Removes a word from the user list

**Parameters**

- **word**
  - The word to remove

**Remarks**

This method is only affects the user word list

**See Also**


__________

Copyright (C) 2003 Paul Welter

### Classes

<table>
<thead>
<tr>
<th>Class</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>AffixEntry</strong></td>
<td>Rule Entry for expanding base words</td>
</tr>
<tr>
<td><strong>AffixEntryCollection</strong></td>
<td>A collection that stores 'AffixEntry' objects.</td>
</tr>
<tr>
<td><strong>AffixEntryEnumerator</strong></td>
<td>A strongly typed enumerator for 'AffixEntryCollection'</td>
</tr>
<tr>
<td><strong>AffixRule</strong></td>
<td>Rule for expanding base words</td>
</tr>
<tr>
<td><strong>AffixRuleCollection</strong></td>
<td>A dictionary collection that stores 'AffixRule' objects.</td>
</tr>
<tr>
<td><strong>AffixRuleEnumerator</strong></td>
<td>A strongly typed enumerator for 'AffixRuleCollection'</td>
</tr>
<tr>
<td><strong>AffixUtility</strong></td>
<td>Summary description for AffixUtility.</td>
</tr>
</tbody>
</table>

*Copyright (C) 2003 Paul Welter*
**AffixEntry Class**

Rule Entry for expanding base words

For a list of all members of this type, see **AffixEntry Members**.

**System.Object**  **AffixEntry**

```plaintext
[Visual Basic]
Public Class AffixEntry

[C#]
public class AffixEntry
```

Requirements

- **Assembly**: NetSpell.SpellChecker (in NetSpell.SpellChecker.dll)

See Also

- **AffixEntry Members**  |  NetSpell.SpellChecker.Dictionary.Affix

**Namespace**

---

Copyright (C) 2003 Paul Welter
**AffixEntry Members**

**AffixEntry overview**

### Public Instance Constructors

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>AffixEntry Constructor</strong></td>
<td>Initializes a new instance of the class</td>
</tr>
</tbody>
</table>

### Public Instance Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>AddCharacters</strong></td>
<td>The characters to add to the string</td>
</tr>
<tr>
<td><strong>Condition</strong></td>
<td>The condition to be met in order to add characters</td>
</tr>
<tr>
<td><strong>ConditionCount</strong></td>
<td>The number of conditions that must be met</td>
</tr>
<tr>
<td><strong>StripCharacters</strong></td>
<td>The characters to remove before adding characters</td>
</tr>
</tbody>
</table>

### Public Instance Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Equals</strong> (inherited from Object)</td>
<td>Determines whether the specified <strong>Object</strong> is equal to the current <strong>Object</strong>.</td>
</tr>
<tr>
<td><strong>GetHashCode</strong> (inherited from Object)</td>
<td>Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table.</td>
</tr>
<tr>
<td><strong>GetType</strong> (inherited from Object)</td>
<td>Gets the <strong>Type</strong> of the current instance.</td>
</tr>
<tr>
<td><strong>ToString</strong> (inherited from Object)</td>
<td>Returns a <strong>String</strong> that represents the current <strong>Object</strong>.</td>
</tr>
</tbody>
</table>

### Protected Instance Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Finalize</strong> (inherited from Object)</td>
<td>Allows an <strong>Object</strong> to attempt to</td>
</tr>
</tbody>
</table>

free resources and perform other cleanup operations before the `Object` is reclaimed by garbage collection.

| MemberwiseClone (inherited from Object) | Creates a shallow copy of the current `Object`. |

**See Also**

- [AffixEntry Class](#)

---

Copyright (C) 2003 Paul Welter
AffixEntry Constructor

Initializes a new instance of the class

[Visual Basic] Public Sub New()

[C#]
public AffixEntry();

See Also


Copyright (C) 2003 Paul Welter
AffixEntry Properties

The properties of the AffixEntry class are listed below. For a complete list of AffixEntry class members, see the AffixEntry Members topic.

Public Instance Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>AddCharacters</code></td>
<td>The characters to add to the string</td>
</tr>
<tr>
<td><code>Condition</code></td>
<td>The condition to be met in order to add characters</td>
</tr>
<tr>
<td><code>ConditionCount</code></td>
<td>The number of conditions that must be met</td>
</tr>
<tr>
<td><code>StripCharacters</code></td>
<td>The characters to remove before adding characters</td>
</tr>
</tbody>
</table>

See Also


Copyright (C) 2003 Paul Welter
The characters to add to the string

[Visual Basic] Public Property AddCharacters

[C#]
public string AddCharacters {get; set;}

See Also


Copyright (C) 2003 Paul Welter
AffixEntry.Condition Property

The condition to be met in order to add characters

[Visual Basic] Public Property Condition As Integer()

[C#]
public int[] Condition {get; set;}

See Also


Copyright (C) 2003 Paul Welter
AffixEntry.ConditionCount Property

The number of conditions that must be met

[Visual Basic] Public Property ConditionCount

[C#]
public int ConditionCount {get; set;}

See Also


Copyright (C) 2003 Paul Welter
AffixEntry.StripCharacters Property

The characters to remove before adding characters

[Visual Basic] Public Property StripCharacters

[C#]
public string StripCharacters {get; set;}

See Also


Copyright (C) 2003 Paul Welter
AffixEntryCollection Class

A collection that stores 'AffixEntry' objects.
For a list of all members of this type, see AffixEntryCollection Members.

System.Object  CollectionBase
    AffixEntryCollection

[Visual Basic]
Public Class AffixEntryCollection
    Inherits CollectionBase

[C#]
public class AffixEntryCollection : CollectionBase

Requirements

Assembly: NetSpell.SpellChecker (in NetSpell.SpellChecker.dll)

See Also


Copyright (C) 2003 Paul Welter
## AffixEntryCollection Members

### AffixEntryCollection overview

#### Public Instance Constructors

<table>
<thead>
<tr>
<th>Method (Type)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AffixEntryCollection</td>
<td>Overloaded. Initializes a new instance of the AffixEntryCollection class.</td>
</tr>
</tbody>
</table>

#### Public Instance Properties

<table>
<thead>
<tr>
<th>Property (Type)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Count (inherited from CollectionBase)</td>
<td>Gets the number of elements contained in the CollectionBase instance.</td>
</tr>
<tr>
<td>Item</td>
<td>Represents the 'AffixEntry' item at the specified index position.</td>
</tr>
</tbody>
</table>

#### Public Instance Methods

<table>
<thead>
<tr>
<th>Method (Type)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Add</td>
<td>Adds a 'AffixEntry' item with the specified value to the AffixEntryCollection</td>
</tr>
<tr>
<td>AddRange</td>
<td>Overloaded. Copies the elements of an array at the end of this instance of AffixEntryCollection.</td>
</tr>
<tr>
<td>Clear (inherited from CollectionBase)</td>
<td>Removes all objects from the CollectionBase instance.</td>
</tr>
<tr>
<td>Contains</td>
<td>Gets a value indicating whether the AffixEntryCollection contains the specified value.</td>
</tr>
<tr>
<td>CopyTo</td>
<td>Copies the AffixEntryCollection values to a one-dimensional System.Array instance starting at the specified array index.</td>
</tr>
<tr>
<td>Equals (inherited from Object)</td>
<td>Determines whether the</td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>--------</td>
<td>-------------</td>
</tr>
<tr>
<td>GetEnumerator</td>
<td>Returns an enumerator that can be used to iterate through the 'AffixEntryCollection'.</td>
</tr>
<tr>
<td>GetHashCode (inherited from Object)</td>
<td>Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table.</td>
</tr>
<tr>
<td>GetType (inherited from Object)</td>
<td>Gets the Type of the current instance.</td>
</tr>
<tr>
<td>IndexOf</td>
<td>Returns the index of a 'AffixEntry' object in the collection.</td>
</tr>
<tr>
<td>Insert</td>
<td>Inserts an existing 'AffixEntry' into the collection at the specified index.</td>
</tr>
<tr>
<td>Remove</td>
<td>Removes a specific item from the 'AffixEntryCollection'.</td>
</tr>
<tr>
<td>RemoveAt (inherited from CollectionBase)</td>
<td>Removes the element at the specified index of the CollectionBase instance.</td>
</tr>
<tr>
<td>ToString (inherited from Object)</td>
<td>Returns a String that represents the current Object.</td>
</tr>
</tbody>
</table>

**Protected Instance Properties**

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>InnerList (inherited from CollectionBase)</td>
<td>Gets an ArrayList containing the list of elements in the CollectionBase instance.</td>
</tr>
<tr>
<td>List (inherited from CollectionBase)</td>
<td>Gets an IList containing the list of elements in the CollectionBase instance.</td>
</tr>
</tbody>
</table>

**Protected Instance Methods**
<table>
<thead>
<tr>
<th>Method</th>
<th>(inherited from)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Finalize</strong></td>
<td>Object</td>
<td>Allows an Object to attempt to free resources and perform other cleanup operations before the Object is reclaimed by garbage collection.</td>
</tr>
<tr>
<td><strong>MemberwiseClone</strong></td>
<td>Object</td>
<td>Creates a shallow copy of the current Object.</td>
</tr>
<tr>
<td><strong>OnClear</strong></td>
<td>CollectionBase</td>
<td>Performs additional custom processes when clearing the contents of the CollectionBase instance.</td>
</tr>
<tr>
<td><strong>OnClearComplete</strong></td>
<td>CollectionBase</td>
<td>Performs additional custom processes after clearing the contents of the CollectionBase instance.</td>
</tr>
<tr>
<td><strong>OnInsert</strong></td>
<td>CollectionBase</td>
<td>Performs additional custom processes before inserting a new element into the CollectionBase instance.</td>
</tr>
<tr>
<td><strong>OnInsertComplete</strong></td>
<td>CollectionBase</td>
<td>Performs additional custom processes after inserting a new element into the CollectionBase instance.</td>
</tr>
<tr>
<td><strong>OnRemove</strong></td>
<td>CollectionBase</td>
<td>Performs additional custom processes when removing an element from the CollectionBase instance.</td>
</tr>
<tr>
<td><strong>OnRemoveComplete</strong></td>
<td>CollectionBase</td>
<td>Performs additional custom processes after removing an element from the CollectionBase instance.</td>
</tr>
<tr>
<td><strong>OnSet</strong></td>
<td>CollectionBase</td>
<td>Performs additional custom processes before setting a value in the CollectionBase instance.</td>
</tr>
</tbody>
</table>
### Explicit Interface Implementations

<table>
<thead>
<tr>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ICollection.CopyTo</td>
<td>Performs additional custom processes after setting a value in the CollectionBase instance.</td>
</tr>
<tr>
<td>IList.Add</td>
<td>Performs additional custom processes when validating a value.</td>
</tr>
<tr>
<td>IList.Contains</td>
<td></td>
</tr>
<tr>
<td>IList.IndexOf</td>
<td></td>
</tr>
<tr>
<td>IList.Insert</td>
<td></td>
</tr>
<tr>
<td>IList.Remove</td>
<td></td>
</tr>
</tbody>
</table>

### See Also

[AffixEntryCollection Class](http://example.com) | [NetSpell.SpellChecker.Dictionary.Affix Namespace](http://example.com)
**AffixEntryCollection Constructor**

Initializes a new instance of 'AffixEntryCollection'.

**Overload List**

Initializes a new instance of 'AffixEntryCollection'.

- `public AffixEntryCollection();`

Initializes a new instance of 'AffixEntryCollection' with an array of 'AffixEntry' objects.

- `public AffixEntryCollection(AffixEntry[]);`

Initializes a new instance of 'AffixEntryCollection' based on an already existing instance.

- `public AffixEntryCollection(AffixEntryCollection);`

**See Also**


---

**Copyright (C) 2003 Paul Welter**
# AffixEntryCollection Constructor ()

Initializes a new instance of 'AffixEntryCollection'.

<table>
<thead>
<tr>
<th>[Visual Basic] Overloads Public Sub New()</th>
</tr>
</thead>
<tbody>
<tr>
<td>[C#] public AffixEntryCollection();</td>
</tr>
</tbody>
</table>

## See Also


---

Copyright (C) 2003 Paul Welter
AffixEntryCollection Constructor (AffixEntryCollection)

Initializes a new instance of 'AffixEntryCollection' based on an already existing instance.

[Visual Basic]Overloads Public Sub New( _
    ByVal value As AffixEntryCollection _
)

[C#]
public AffixEntryCollection(
    AffixEntryCollection value
);

Parameters

value
A 'AffixEntryCollection' from which the contents is copied

See Also

Namespace | AffixEntryCollection Constructor Overload List

Copyright (C) 2003 Paul Welter
AffixEntryCollection Constructor (AffixEntry[])  

Initializes a new instance of 'AffixEntryCollection' with an array of 'AffixEntry' objects.

[Visual Basic] Overloads Public Sub New(  
    ByVal value As AffixEntry()  
)

[C#]  
public AffixEntryCollection(  
    AffixEntry[] value  
);

Parameters

value  
An array of 'AffixEntry' objects with which to initialize the collection

See Also


Copyright (C) 2003 Paul Welter
AffixEntryCollection Properties

The properties of the **AffixEntryCollection** class are listed below. For a complete list of **AffixEntryCollection** class members, see the **AffixEntryCollection Members** topic.

### Public Instance Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Count</strong></td>
<td>Gets the number of elements contained in the <strong>CollectionBase</strong> instance.</td>
</tr>
<tr>
<td><strong>Item</strong></td>
<td>Represents the 'AffixEntry' item at the specified index position.</td>
</tr>
</tbody>
</table>

### Protected Instance Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>InnerList</strong></td>
<td>Gets an <strong>ArrayList</strong> containing the list of elements in the <strong>CollectionBase</strong> instance.</td>
</tr>
<tr>
<td><strong>List</strong></td>
<td>Gets an <strong>IList</strong> containing the list of elements in the <strong>CollectionBase</strong> instance.</td>
</tr>
</tbody>
</table>

See Also

**AffixEntryCollection Class** | **NetSpell.SpellChecker.Dictionary.Affix Namespace**

---

Copyright (C) 2003 Paul Welter
AffixEntryCollection.Item Property

Represents the 'AffixEntry' item at the specified index position.

[Visual Basic] Public Default Property Item(
    ByVal index As Integer
) As AffixEntry

[C#] public AffixEntry this[int index] {get; set;}

Parameters

index
The zero-based index of the entry to locate in the collection.

Property Value
The entry at the specified index of the collection.

See Also


Copyright (C) 2003 Paul Welter
AffixEntryCollection Methods

The methods of the AffixEntryCollection class are listed below. For a complete list of AffixEntryCollection class members, see the AffixEntryCollection Members topic.

### Public Instance Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Add</strong></td>
<td>Adds a 'AffixEntry' item with the specified value to the 'AffixEntryCollection'</td>
</tr>
<tr>
<td><strong>AddRange</strong></td>
<td>Overloaded. Copies the elements of an array at the end of this instance of 'AffixEntryCollection'.</td>
</tr>
<tr>
<td><strong>Clear</strong> (inherited from CollectionBase)</td>
<td>Removes all objects from the CollectionBase instance.</td>
</tr>
<tr>
<td><strong>Contains</strong></td>
<td>Gets a value indicating whether the 'AffixEntryCollection' contains the specified value.</td>
</tr>
<tr>
<td><strong>CopyTo</strong></td>
<td>Copies the 'AffixEntryCollection' values to a one-dimensional System.Array instance starting at the specified array index.</td>
</tr>
<tr>
<td><strong>Equals</strong> (inherited from Object)</td>
<td>Determines whether the specified Object is equal to the current Object.</td>
</tr>
<tr>
<td><strong>GetEnumerator</strong></td>
<td>Returns an enumerator that can be used to iterate through the 'AffixEntryCollection'.</td>
</tr>
<tr>
<td><strong>GetHashCode</strong> (inherited from Object)</td>
<td>Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table.</td>
</tr>
<tr>
<td><strong>GetType</strong> (inherited from Object)</td>
<td>Gets the Type of the current</td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------</td>
<td>--------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td><strong>IndexOf</strong></td>
<td>Returns the index of a 'AffixEntry' object in the collection.</td>
</tr>
<tr>
<td><strong>Insert</strong></td>
<td>Inserts an existing 'AffixEntry' into the collection at the specified index.</td>
</tr>
<tr>
<td><strong>Remove</strong></td>
<td>Removes a specific item from the 'AffixEntryCollection'.</td>
</tr>
<tr>
<td><strong>RemoveAt</strong></td>
<td>Removes the element at the specified index of the CollectionBase instance.</td>
</tr>
<tr>
<td><strong>ToString</strong></td>
<td>Returns a String that represents the current Object.</td>
</tr>
</tbody>
</table>

**Protected Instance Methods**

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Finalize</strong></td>
<td>Allows an Object to attempt to free resources and perform other cleanup operations before the Object is reclaimed by garbage collection.</td>
</tr>
<tr>
<td><strong>MemberwiseClone</strong></td>
<td>Creates a shallow copy of the current Object.</td>
</tr>
<tr>
<td><strong>OnClear</strong></td>
<td>Performs additional custom processes when clearing the contents of the CollectionBase instance.</td>
</tr>
<tr>
<td><strong>OnClearComplete</strong></td>
<td>Performs additional custom processes after clearing the contents of the CollectionBase instance.</td>
</tr>
<tr>
<td><strong>OnInsert</strong></td>
<td>Performs additional custom processes before inserting a new element into the CollectionBase instance.</td>
</tr>
<tr>
<td>Event</td>
<td>Description</td>
</tr>
<tr>
<td>-------</td>
<td>-------------</td>
</tr>
<tr>
<td><strong>OnInsertComplete</strong> (inherited from CollectionBase)</td>
<td>Performs additional custom processes after inserting a new element into the CollectionBase instance.</td>
</tr>
<tr>
<td><strong>OnRemove</strong> (inherited from CollectionBase)</td>
<td>Performs additional custom processes when removing an element from the CollectionBase instance.</td>
</tr>
<tr>
<td><strong>OnRemoveComplete</strong> (inherited from CollectionBase)</td>
<td>Performs additional custom processes after removing an element from the CollectionBase instance.</td>
</tr>
<tr>
<td><strong>OnSet</strong> (inherited from CollectionBase)</td>
<td>Performs additional custom processes before setting a value in the CollectionBase instance.</td>
</tr>
<tr>
<td><strong>OnSetComplete</strong> (inherited from CollectionBase)</td>
<td>Performs additional custom processes after setting a value in the CollectionBase instance.</td>
</tr>
<tr>
<td><strong>OnValidate</strong> (inherited from CollectionBase)</td>
<td>Performs additional custom processes when validating a value.</td>
</tr>
</tbody>
</table>

**Explicit Interface Implementations**

<table>
<thead>
<tr>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ICollection.CopyTo</td>
<td>(inherited from CollectionBase)</td>
</tr>
<tr>
<td>IList.Add</td>
<td>(inherited from CollectionBase)</td>
</tr>
<tr>
<td>IList.Contains</td>
<td>(inherited from CollectionBase)</td>
</tr>
<tr>
<td>IList.IndexOf</td>
<td>(inherited from CollectionBase)</td>
</tr>
<tr>
<td><strong>IList.Insert</strong> (inherited from CollectionBase)</td>
<td></td>
</tr>
<tr>
<td><strong>IList.Remove</strong> (inherited from CollectionBase)</td>
<td></td>
</tr>
</tbody>
</table>

See Also


Copyright (C) 2003 Paul Welter
AffixEntryCollection.Add Method

Adds a 'AffixEntry' item with the specified value to the 'AffixEntryCollection'

[Visual Basic]
Public Function Add( _
    ByVal value As AffixEntry _
) As Integer

[C#]
public int Add(
    AffixEntry value
);

Parameters

value
    The 'AffixEntry' to add.

Return Value

The index at which the new element was inserted.

See Also


Copyright (C) 2003 Paul Welter
AffixEntryCollection.AddRange Method

Copies the elements of an array at the end of this instance of 'AffixEntryCollection'.

Overload List

Copies the elements of an array at the end of this instance of 'AffixEntryCollection'.

```csharp
public void AddRange(AffixEntry[]);
```

Adds the contents of another 'AffixEntryCollection' at the end of this instance.

```csharp
public void AddRange(AffixEntryCollection);
```

See Also


Copyright (C) 2003 Paul Welter
AffixEntryCollection.AddRange Method (AffixEntry[])  

Copies the elements of an array at the end of this instance of 'AffixEntryCollection'.

[Visual Basic]
Overloads Public Sub AddRange(
   ByVal value As AffixEntry() 
)

[C#]
public void AddRange(
   AffixEntry[] value
);

Parameters

value
An array of 'AffixEntry' objects to add to the collection.

See Also


Copyright (C) 2003 Paul Welter
AffixEntryCollection.AddRange Method (AffixEntryCollection)

Adds the contents of another 'AffixEntryCollection' at the end of this instance.

[Visual Basic] Overloads Public Sub AddRange(
    ByVal value As AffixEntryCollection
)

[C#]
public void AddRange(
    AffixEntryCollection value
);

Parameters

value
A 'AffixEntryCollection' containing the objects to add to the collection.

See Also

Namespace | AffixEntryCollection.AddRange Overload List

Copyright (C) 2003 Paul Welter
AffixEntryCollection.Contains Method

Gets a value indicating whether the 'AffixEntryCollection' contains the specified value.

[Visual Basic]
Public Function Contains( _
    ByVal value As AffixEntry _
) As Boolean

[C#]
public bool Contains(
    AffixEntry value
);

Parameters

value
   The item to locate.

Return Value

True if the item exists in the collection; false otherwise.

See Also


Copyright (C) 2003 Paul Welter
AffixEntryCollection.CopyTo Method

Copies the 'AffixEntryCollection' values to a one-dimensional System.Array instance starting at the specified array index.

[Visual Basic]
Public Sub CopyTo( _
    ByVal array As AffixEntry(), _
    ByVal index As Integer _
)

[C#]
public void CopyTo(  
    AffixEntry[] array,  
    int index  
);

Parameters

array
   The one-dimensional System.Array that represents the copy destination.

index
   The index in the array where copying begins.

See Also


Copyright (C) 2003 Paul Welter
**AffixEntryCollection.GetEnumerator Method**

Returns an enumerator that can be used to iterate through the 'AffixEntryCollection'.

[Visual Basic]  
```vbnet
Public Function GetEnumerator() As New public AffixEntryEnumerator
```

[C#]  
```csharp
new public AffixEntryEnumerator GetEnumerator();
```

See Also

- [AffixEntryCollection Class](#)  

---

Copyright (C) 2003 Paul Welter
AffixEntryCollection.IndexOf Method

Returns the index of a 'AffixEntry' object in the collection.

[Visual Basic]
Public Function IndexOf( _
    ByVal value As AffixEntry _
) As Integer

[C#]
public int IndexOf(
    AffixEntry value
);

Parameters

value
    The 'AffixEntry' object whose index will be retrieved.

Return Value

If found, the index of the value; otherwise, -1.

See Also


Copyright (C) 2003 Paul Welter
AffixEntryCollection.Insert Method

 Inserts an existing 'AffixEntry' into the collection at the specified index.

 [Visual Basic] Public Sub Insert( _
    ByVal index As Integer, _
    ByVal value As AffixEntry _
) _

 [C#] public void Insert( _
    int index, _
    AffixEntry value _
); _

Parameters

index
  The zero-based index where the new item should be inserted.

value
  The item to insert.

See Also


Copyright (C) 2003 Paul Welter
AffixEntryCollection.Remove Method

Removes a specific item from the 'AffixEntryCollection'.

[Visual Basic] Public Sub Remove( ByVal value As AffixEntry )

[C#] public void Remove( AffixEntry value );

Parameters

value
The item to remove from the 'AffixEntryCollection'.

See Also


Copyright (C) 2003 Paul Welter
AffixEntryEnumerator Class

A strongly typed enumerator for 'AffixEntryCollection'

For a list of all members of this type, see AffixEntryEnumerator Members.

System.Object  AffixEntryEnumerator

[Visual Basic]
Public Class AffixEntryEnumerator
    Implements IEnumerator

[C#]
public class AffixEntryEnumerator : IEnumerator

Requirements


Assembly: NetSpell.SpellChecker (in NetSpell.SpellChecker.dll)

See Also


Copyright (C) 2003 Paul Welter
### AffixEntryEnumerator Members

#### AffixEntryEnumerator overview

#### Public Instance Constructors

<table>
<thead>
<tr>
<th>Constructor</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>AffixEntryEnumerator Constructor</strong></td>
<td>Enumerator constructor</td>
</tr>
</tbody>
</table>

#### Public Instance Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Current</strong></td>
<td>Gets the current element from the collection (strongly typed)</td>
</tr>
</tbody>
</table>

#### Public Instance Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Equals</strong> (inherited from Object)</td>
<td>Determines whether the specified <strong>Object</strong> is equal to the current <strong>Object</strong>.</td>
</tr>
<tr>
<td><strong>GetHashCode</strong> (inherited from Object)</td>
<td>Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table.</td>
</tr>
<tr>
<td><strong>GetType</strong> (inherited from Object)</td>
<td>Gets the <strong>Type</strong> of the current instance.</td>
</tr>
<tr>
<td><strong>MoveNext</strong></td>
<td>Advances the enumerator to the next element of the collection</td>
</tr>
<tr>
<td><strong>Reset</strong></td>
<td>Sets the enumerator to the first element in the collection</td>
</tr>
<tr>
<td><strong>ToString</strong> (inherited from Object)</td>
<td>Returns a <strong>String</strong> that represents the current <strong>Object</strong>.</td>
</tr>
</tbody>
</table>

#### Protected Instance Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Finalize</strong> (inherited from Object)</td>
<td>Allows an <strong>Object</strong> to attempt to free resources and perform other cleanup operations before the <strong>Object</strong> is reclaimed by</td>
</tr>
</tbody>
</table>
garbage collection.

| MemberwiseClone (inherited from Object) | Creates a shallow copy of the current Object. |

Private Instance Properties

| System.Collections.IEnumerator.Current |

Explicit Interface Implementations

| IEnumerator.MoveNext advances the enumerator to the next element of the collection |
| IEnumerator.Reset sets the enumerator to the first element in the collection |

See Also


Copyright (C) 2003 Paul Welter
AffixEntryEnumerator Constructor

Enumerator constructor

[Visual Basic] Public Sub New( _
    ByVal mappings As AffixEntryCollection _
)

[C#] public AffixEntryEnumerator( 
    AffixEntryCollection mappings 
);

See Also


Copyright (C) 2003 Paul Welter
## AffixEntryEnumerator Properties

The properties of the `AffixEntryEnumerator` class are listed below. For a complete list of `AffixEntryEnumerator` class members, see the [AffixEntryEnumerator Members](#) topic.

<table>
<thead>
<tr>
<th>Public Instance Properties</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><code>Current</code></td>
<td>Gets the current element from the collection (strongly typed)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Private Instance Properties</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><code>System.Collections.IEnumerator.Current</code></td>
<td></td>
</tr>
</tbody>
</table>

### See Also


---

**Copyright (C) 2003 Paul Welter**
AffixEntryEnumerator.Current Property

Gets the current element from the collection (strongly typed)

[Visual Basic] Public ReadOnly Property Current

[C#] public AffixEntry Current {get;}

See Also


Copyright (C) 2003 Paul Welter

[Visual Basic] 
NotOverridable Private ReadOnly Property System.Collections.IEnumerator.Current

[C#]
private object System.Collections.IEnumerator.Current

See Also


Copyright (C) 2003 Paul Welter
AffixEntryEnumerator Methods

The methods of the AffixEntryEnumerator class are listed below. For a complete list of AffixEntryEnumerator class members, see the AffixEntryEnumerator Members topic.

Public Instance Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Equals</strong> <em>(inherited from Object)</em></td>
<td>Determines whether the specified Object is equal to the current Object.</td>
</tr>
<tr>
<td><strong>GetHashCode</strong> <em>(inherited from Object)</em></td>
<td>Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table.</td>
</tr>
<tr>
<td><strong>GetType</strong> <em>(inherited from Object)</em></td>
<td>Gets the Type of the current instance.</td>
</tr>
<tr>
<td><strong>MoveNext</strong></td>
<td>Advances the enumerator to the next element of the collection</td>
</tr>
<tr>
<td><strong>Reset</strong></td>
<td>Sets the enumerator to the first element in the collection</td>
</tr>
<tr>
<td><strong>ToString</strong> <em>(inherited from Object)</em></td>
<td>Returns a String that represents the current Object.</td>
</tr>
</tbody>
</table>

Protected Instance Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Finalize</strong> <em>(inherited from Object)</em></td>
<td>Allows an Object to attempt to free resources and perform other cleanup operations before the Object is reclaimed by garbage collection.</td>
</tr>
<tr>
<td><strong>MemberwiseClone</strong> <em>(inherited from Object)</em></td>
<td>Creates a shallow copy of the current Object.</td>
</tr>
</tbody>
</table>

Explicit Interface Implementations

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
</table>
| **IEnumerator.MoveNext**   | Advances the enumerator to the }
See Also


Copyright (C) 2003 Paul Welter
AffixEntryEnumerator.MoveNext Method

Advances the enumerator to the next element of the collection

[Visual Basic] Public Function MoveNext() As Boolean

[C#]
public bool MoveNext();

See Also


Copyright (C) 2003 Paul Welter
**AffixEntryEnumerator.Reset Method**

Sets the enumerator to the first element in the collection

[Visual Basic] 
```vbnet
Public Sub Reset()
```

[C#] 
```csharp
public void Reset();
```

See Also


---

Copyright (C) 2003 Paul Welter
**AffixEntryEnumerator.IEnumerator.MoveNext Method**

Advances the enumerator to the next element of the collection

```visual-basic
Function MoveNext() As Boolean
    Implements IEnumerator.MoveNext
```

```csharp
bool IEnumerator.MoveNext();
```

**Implements**

`IEnumerator.MoveNext`

**See Also**


---

Copyright (C) 2003 Paul Welter
AffixEntryEnumerator.IEnumerable.Reset Method

Sets the enumerator to the first element in the collection

[Visual Basic] Sub Reset() Implements _
IEnumerator.Reset

[C#]
void IEnumerator.Reset();

Implements
IEnumerator.Reset

See Also
Namespace

Copyright (C) 2003 Paul Welter
**AffixRule Class**

Rule for expanding base words

For a list of all members of this type, see **AffixRule Members**.

**System.Object** **AffixRule**

```vbnet
Public Class AffixRule
```

```csharp
public class AffixRule
```

**Requirements**

**Namespace:** NetSpell.SpellChecker.Dictionary.Affix

**Assembly:** NetSpell.SpellChecker (in NetSpell.SpellChecker.dll)

**See Also**


---

**Copyright (C) 2003 Paul Welter**
## AffixRule Members

### AffixRule overview

**Public Instance Constructors**

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>AffixRule Constructor</strong></td>
<td>Initializes a new instance of the class</td>
</tr>
</tbody>
</table>

**Public Instance Properties**

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>AffixEntries</strong></td>
<td>Collection of text entries that make up this rule</td>
</tr>
<tr>
<td><strong>AllowCombine</strong></td>
<td>Allow combining prefix and suffix</td>
</tr>
<tr>
<td><strong>Name</strong></td>
<td>Name of the Affix rule</td>
</tr>
</tbody>
</table>

**Public Instance Methods**

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Equals</strong> (inherited from <strong>Object</strong>)</td>
<td>Determines whether the specified <strong>Object</strong> is equal to the current <strong>Object</strong>.</td>
</tr>
<tr>
<td><strong>GetHashCode</strong> (inherited from <strong>Object</strong>)</td>
<td>Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table.</td>
</tr>
<tr>
<td><strong>GetType</strong> (inherited from <strong>Object</strong>)</td>
<td>Gets the <strong>Type</strong> of the current instance.</td>
</tr>
<tr>
<td><strong>ToString</strong> (inherited from <strong>Object</strong>)</td>
<td>Returns a <strong>String</strong> that represents the current <strong>Object</strong>.</td>
</tr>
</tbody>
</table>

### Protected Instance Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Finalize</strong> (inherited from <strong>Object</strong>)</td>
<td>Allows an <strong>Object</strong> to attempt to free resources and perform other cleanup operations before the <strong>Object</strong> is reclaimed by garbage collection.</td>
</tr>
<tr>
<td><strong>MemberwiseClone</strong> (inherited from <strong>Object</strong>)</td>
<td>Creates a shallow copy of the current <strong>Object</strong>.</td>
</tr>
</tbody>
</table>

**See Also**


[Copyright (C) 2003 Paul Welter]
AffixRule Constructor

Initializes a new instance of the class

[Visual Basic] Public Sub New()

[C#]
public AffixRule();

See Also


Copyright (C) 2003 Paul Welter
AffixRule Properties

The properties of the AffixRule class are listed below. For a complete list of AffixRule class members, see the AffixRule Members topic.

Public Instance Properties

<table>
<thead>
<tr>
<th>AffixEntries</th>
<th>Collection of text entries that make up this rule</th>
</tr>
</thead>
<tbody>
<tr>
<td>AllowCombine</td>
<td>Allow combining prefix and suffix</td>
</tr>
<tr>
<td>Name</td>
<td>Name of the Affix rule</td>
</tr>
</tbody>
</table>

See Also


Copyright (C) 2003 Paul Welter
AffixRule.AffixEntries Property

Collection of text entries that make up this rule

[Visual Basic] Public Property AffixEntries As

[C#] public AffixEntryCollection AffixEntries {get;

See Also


Copyright (C) 2003 Paul Welter
AffixRule.AllowCombine Property

Allow combining prefix and suffix

[Visual Basic] Public Property AllowCombine As

[C#]
public bool AllowCombine {get; set;}

See Also


Copyright (C) 2003 Paul Welter
AffixRule.Name Property

Name of the Affix rule

[Visual Basic] Public Property Name As String

[C#]
public string Name {get; set;}

See Also


Copyright (C) 2003 Paul Welter
AffixRuleCollection Class

A dictionary collection that stores 'AffixRule' objects.

For a list of all members of this type, see AffixRuleCollection Members.

**System.Object**  AffixRuleCollection

```
[Visual Basic]
Public Class AffixRuleCollection
    Implements IDictionary, ICollection, IEnumerable, ICloneable

[C#]
public class AffixRuleCollection :
    IDictionary, ICollection, IEnumerable, ICloneable
```

**Requirements**

**Namespace:** NetSpell.SpellChecker.Dictionary.Affix

**Assembly:** NetSpell.SpellChecker (in NetSpell.SpellChecker.dll)

**See Also**

________________________________________________________________________

Copyright (C) 2003 Paul Welter
### AffixRuleCollection Members

#### AffixRuleCollection overview

#### Public Static (Shared) Methods

<table>
<thead>
<tr>
<th>Synchronized</th>
<th>Returns a synchronized (thread-safe) wrapper for the AffixRuleCollection.</th>
</tr>
</thead>
</table>

#### Public Instance Constructors

<table>
<thead>
<tr>
<th>AffixRuleCollection</th>
<th>Overloaded. Initializes a new instance of the AffixRuleCollection class.</th>
</tr>
</thead>
</table>

#### Public Instance Properties

<table>
<thead>
<tr>
<th>Count</th>
<th>Gets the number of elements contained in the AffixRuleCollection.</th>
</tr>
</thead>
<tbody>
<tr>
<td>IsFixedSize</td>
<td>gets a value indicating whether the AffixRuleCollection has a fixed size.</td>
</tr>
<tr>
<td>IsReadOnly</td>
<td>gets a value indicating whether the AffixRuleCollection is read-only.</td>
</tr>
<tr>
<td>IsSynchronized</td>
<td>Gets a value indicating whether access to the AffixRuleCollection is synchronized (thread-safe).</td>
</tr>
<tr>
<td>Item</td>
<td>Gets or sets the element with the specified key.</td>
</tr>
<tr>
<td>Keys</td>
<td>gets an ICollection containing the keys of the AffixRuleCollection.</td>
</tr>
<tr>
<td>SyncRoot</td>
<td>Gets an object that can be used</td>
</tr>
</tbody>
</table>
to synchronize access to the AffixRuleCollection.

**Values**
gets an ICollection containing the values in the AffixRuleCollection.

### Public Instance Methods

- **Add**
  adds an element with the provided key and value to the AffixRuleCollection.

- **Clear**
  removes all elements from the AffixRuleCollection.

- **Clone**
  Creates a new object that is a copy of the current instance.

- **Contains**
  Determines whether the AffixRuleCollection contains an element with the specified key.

- **ContainsKey**
  Determines whether the AffixRuleCollection contains a specific key.

- **ContainsValue**
  Determines whether the AffixRuleCollection contains a specific value.

- **CopyTo**
  copies the elements of the AffixRuleCollection to an Array, starting at a particular Array index.

- **Equals** *(inherited from Object)*
  Determines whether the specified Object is equal to the current Object.

- **GetEnumerator**
  Returns an enumerator that can be used to iterate through the 'AffixRuleCollection'.
<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GetHashCode</td>
<td>Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table.</td>
</tr>
<tr>
<td>GetType</td>
<td>Gets the <strong>Type</strong> of the current instance.</td>
</tr>
<tr>
<td>Remove</td>
<td>Removes the element with the specified key from the AffixRuleCollection.</td>
</tr>
<tr>
<td>ToString</td>
<td>Returns a <strong>String</strong> that represents the current <strong>Object</strong>.</td>
</tr>
</tbody>
</table>

**Protected Instance Fields**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>innerHash</td>
<td>Internal Hashtable</td>
</tr>
</tbody>
</table>

**Protected Instance Methods**

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Finalize</td>
<td>Allows an <strong>Object</strong> to attempt to free resources and perform other cleanup operations before the <strong>Object</strong> is reclaimed by garbage collection.</td>
</tr>
<tr>
<td>MemberwiseClone</td>
<td>Creates a shallow copy of the current <strong>Object</strong>.</td>
</tr>
</tbody>
</table>

**Private Instance Properties**

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>System.Collections.IDictionary.Item</td>
<td></td>
</tr>
</tbody>
</table>

**Explicit Interface Implementations**

<table>
<thead>
<tr>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ICloneable.Clone</td>
<td></td>
</tr>
<tr>
<td>IDictionary.Add</td>
<td></td>
</tr>
<tr>
<td>IDictionary.Contains</td>
<td></td>
</tr>
<tr>
<td>IDictionary.GetEnumerator</td>
<td></td>
</tr>
</tbody>
</table>
**See Also**


Copyright (C) 2003 Paul Welter
AffixRuleCollection Constructor

Initializes a new instance of 'AffixRuleCollection'.

Overload List

Initializes a new instance of 'AffixRuleCollection'.

public AffixRuleCollection();

Initializes a new instance of 'AffixRuleCollection'.

public AffixRuleCollection(AffixRuleCollection);

Initializes a new instance of 'AffixRuleCollection'.

public AffixRuleCollection(IDictionary);

Initializes a new instance of 'AffixRuleCollection'.

public AffixRuleCollection(IDictionary, IHashCodeProvider, IComparer);

Initializes a new instance of 'AffixRuleCollection'.

public AffixRuleCollection(IDictionary, float);

Initializes a new instance of 'AffixRuleCollection'.

public AffixRuleCollection(IDictionary, float, IHashCodeProvider, IComparer);

Initializes a new instance of 'AffixRuleCollection'.

public AffixRuleCollection(IHashCodeProvider, IComparer);

Initializes a new instance of 'AffixRuleCollection'.

public AffixRuleCollection(int);

Initializes a new instance of 'AffixRuleCollection'.

public AffixRuleCollection(int, IHashCodeProvider, IComparer);

Initializes a new instance of 'AffixRuleCollection'.

public AffixRuleCollection(int, int);

Initializes a new instance of 'AffixRuleCollection'.

public AffixRuleCollection(int, float, IHashCodeProvider, IComparer);

See Also
### AffixRuleCollection Constructor ()

Initializes a new instance of 'AffixRuleCollection'.

<table>
<thead>
<tr>
<th>[Visual Basic] Overloads Public Sub New()</th>
</tr>
</thead>
<tbody>
<tr>
<td>[C#] public AffixRuleCollection();</td>
</tr>
</tbody>
</table>

**See Also**

- AffixRuleCollection Class
- AffixRuleCollection Constructor Overload List

---

Copyright (C) 2003 Paul Welter
Initializes a new instance of 'AffixRuleCollection'.

Parameters

original

A 'AffixRuleCollection' from which the contents is copied

See Also


Copyright (C) 2003 Paul Welter
AffixRuleCollection Constructor (IDictionary)

Initializes a new instance of 'AffixRuleCollection'.

[Visual Basic]
Overloads Public Sub New( _
    ByVal dictionary As IDictionary _
)

[C#]
public AffixRuleCollection(
    IDictionary dictionary
);

Parameters

dictionary

The IDictionary to copy to a new 'AffixRuleCollection'.

See Also


Copyright (C) 2003 Paul Welter
AffixRuleCollection Constructor (Int32)

Initializes a new instance of 'AffixRuleCollection'.

[Visual Basic] Overloads Public Sub New( _
    ByVal capacity As Integer _
)

[C#]
public AffixRuleCollection(
    int capacity
);

Parameters

capacity

The approximate number of elements that the 'AffixRuleCollection' can initially contain.

See Also


Copyright (C) 2003 Paul Welter
AffixRuleCollection Constructor (IDictionary, Single)

Initializes a new instance of 'AffixRuleCollection'.

[Visual Basic]
Overloads Public Sub New( _
    ByVal dictionary As IDictionary, _
    ByVal loadFactor As Single _
)

[C#]
public AffixRuleCollection(  
    IDictionary dictionary,  
    float loadFactor  
);  

Parameters

dictionary

The IDictionary to copy to a new 'AffixRuleCollection'.

loadFactor

A number in the range from 0.1 through 1.0 indicating the maximum ratio of elements to buckets.

See Also


Copyright (C) 2003 Paul Welter
AffixRuleCollection Constructor (IHashCodeProvider, IComparer)

Initializes a new instance of 'AffixRuleCollection'.

[Visual Basic] Overloads Public Sub New( _
    ByVal codeProvider As IHashCodeProvider, _
    ByVal comparer As IComparer _
)

[C#]
public AffixRuleCollection(
    IHashCodeProvider codeProvider,
    IComparer comparer
);

Parameters

codeProvider

The IHashCodeProvider that supplies the hash codes for all keys in the 'AffixRuleCollection'.

comparer

The IComparer to use to determine whether two keys are equal.

See Also


Copyright (C) 2003 Paul Welter
**AffixRuleCollection Constructor (Int32, Int32)**

Initializes a new instance of 'AffixRuleCollection'.

```csharp
public AffixRuleCollection(
    int capacity,
    int loadFactor
);
```

### Parameters

**capacity**

The approximate number of elements that the 'AffixRuleCollection' can initially contain.

**loadFactor**

A number in the range from 0.1 through 1.0 indicating the maximum ratio of elements to buckets.

### See Also


---

**Copyright (C) 2003 Paul Welter**
NetSpell Documentation - v.2.1.7
AffixRuleCollection Constructor (IDictionary, IHashCodeProvider, IComparer)

Initializes a new instance of 'AffixRuleCollection'.

[Visual Basic]
Overloads Public Sub New( _
    ByVal dictionary As IDictionary, _
    ByVal codeProvider As IHashCodeProvider, _
    ByVal comparer As IComparer _
)

[C#]
public AffixRuleCollection(
    IDictionary dictionary,
    IHashCodeProvider codeProvider,
    IComparer comparer
);

Parameters

dictionary
    The IDictionary to copy to a new 'AffixRuleCollection'.

codeProvider
    The IHashCodeProvider that supplies the hash codes for all keys
    in the 'AffixRuleCollection'.

comparer
    The IComparer to use to determine whether two keys are equal.

See Also

Namespace | AffixRuleCollection Constructor Overload List

Copyright (C) 2003 Paul Welter
AffixRuleCollection Constructor (Int32, IHashCodeProvider, IComparer)

Initializes a new instance of 'AffixRuleCollection'.

[Visual Basic] Overloads Public Sub New( _
    ByVal capacity As Integer, _
    ByVal codeProvider As IHashCodeProvider, _
    ByVal comparer As IComparer _
)

[C#] public AffixRuleCollection( _
    int capacity, _
    IHashCodeProvider codeProvider, _
    IComparer comparer _
);

Parameters

capacity

The approximate number of elements that the 'AffixRuleCollection' can initially contain.

codeProvider

The IHashCodeProvider that supplies the hash codes for all keys in the 'AffixRuleCollection'.

comparer

The IComparer to use to determine whether two keys are equal.

See Also


Copyright (C) 2003 Paul Welter
**AffixRuleCollection Constructor (IDictionary, Single, IHashCodeProvider, IComparer)**

Initializes a new instance of 'AffixRuleCollection'.

**Visual Basic**

```vbnet
Overloads Public Sub New(
    ByVal dictionary As IDictionary,
    ByVal loadFactor As Single,
    ByVal codeProvider As IHashCodeProvider,
    ByVal comparer As IComparer)
```

**C#**

```csharp
public AffixRuleCollection(
    IDictionary dictionary,
    float loadFactor,
    IHashCodeProvider codeProvider,
    IComparer comparer
);
```

**Parameters**

- **dictionary**
  
  The IDictionary to copy to a new 'AffixRuleCollection'.

- **loadFactor**
  
  A number in the range from 0.1 through 1.0 indicating the maximum ratio of elements to buckets.

- **codeProvider**
  
  The IHashCodeProvider that supplies the hash codes for all keys in the 'AffixRuleCollection'.

- **comparer**
  
  The IComparer to use to determine whether two keys are equal.

**See Also**
Namespace | AffixRuleCollection Constructor Overload List

Copyright (C) 2003 Paul Welter
## AffixRuleCollection Constructor (Int32, Single, IHashCodeProvider, IComparer)

Initializes a new instance of 'AffixRuleCollection'.

### [Visual Basic]
```vbnet
Overloads Public Sub New( 
    ByVal capacity As Integer, 
    ByVal loadFactor As Single, 
    ByVal codeProvider As IHashCodeProvider, 
    ByVal comparer As IComparer 
)
```

### [C#]
```csharp
public AffixRuleCollection(
    int capacity,
    float loadFactor,
    IHashCodeProvider codeProvider,
    IComparer comparer
);
```

### Parameters

**capacity**

The approximate number of elements that the 'AffixRuleCollection' can initially contain.

**loadFactor**

A number in the range from 0.1 through 1.0 indicating the maximum ratio of elements to buckets.

**codeProvider**

The IHashCodeProvider that supplies the hash codes for all keys in the 'AffixRuleCollection'.

**comparer**

The IComparer to use to determine whether two keys are equal.
See Also

Namespace | AffixRuleCollection Constructor Overload List

Copyright (C) 2003 Paul Welter
AffixRuleCollection Fields

The fields of the `AffixRuleCollection` class are listed below. For a complete list of `AffixRuleCollection` class members, see the `AffixRuleCollection Members` topic.

Protected Instance Fields

| `innerHash` | Internal Hasetable |

See Also


Copyright (C) 2003 Paul Welter
AffixRuleCollection.innerHash Field

Internal Hashtable

[Visual Basic] Protected innerHash As Hashtable

[C#] protected Hashtable innerHash;

See Also


Copyright (C) 2003 Paul Welter
AffixRuleCollection Properties

The properties of the AffixRuleCollection class are listed below. For a complete list of AffixRuleCollection class members, see the AffixRuleCollection Members topic.

Public Instance Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Count</td>
<td>Gets the number of elements contained in the AffixRuleCollection.</td>
</tr>
<tr>
<td>IsFixedSize</td>
<td>gets a value indicating whether the AffixRuleCollection has a fixed size.</td>
</tr>
<tr>
<td>IsReadOnly</td>
<td>gets a value indicating whether the AffixRuleCollection is read-only.</td>
</tr>
<tr>
<td>IsSynchronized</td>
<td>Gets a value indicating whether access to the AffixRuleCollection is synchronized (thread-safe).</td>
</tr>
<tr>
<td>Item</td>
<td>Gets or sets the element with the specified key.</td>
</tr>
<tr>
<td>Keys</td>
<td>gets an ICollection containing the keys of the AffixRuleCollection.</td>
</tr>
<tr>
<td>SyncRoot</td>
<td>Gets an object that can be used to synchronize access to the AffixRuleCollection.</td>
</tr>
<tr>
<td>Values</td>
<td>gets an ICollection containing the values in the AffixRuleCollection.</td>
</tr>
</tbody>
</table>

Private Instance Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>System.Collections.IDictionary.Item</td>
<td></td>
</tr>
</tbody>
</table>
See Also


Copyright (C) 2003 Paul Welter
AffixRuleCollection.Count Property

Gets the number of elements contained in the AffixRuleCollection.

[Visual Basic] NotOverridable Public ReadOnly Property Count As Implements _ ICollection.Count

[C#] public int Count {get;}

Implements

ICollection.Count

See Also


Copyright (C) 2003 Paul Welter
AffixRuleCollection.IsFixedSize Property

gets a value indicating whether the AffixRuleCollection has a fixed size.

[Visual Basic] NotOverridable Public ReadOnly Property IsFixedSize As Implements _
  IDictionary.IsFixedSize

[C#]
public bool IsFixedSize {get;}

Implements

IDictionary.IsFixedSize

See Also


Copyright (C) 2003 Paul Welter
**AffixRuleCollection.IsReadOnly Property**

gets a value indicating whether the AffixRuleCollection is read-only.

```csharp
public bool IsReadOnly {get;}
```

**Implements**

`IDictionary.IsReadOnly`

**See Also**


---

Copyright (C) 2003 Paul Welter
AffixRuleCollection.IsSynchronized Property

Gets a value indicating whether access to the AffixRuleCollection is synchronized (thread-safe).

[Visual Basic] NotOverridable Public ReadOnly Property IsSynchronized As Implements _ ICollection.IsSynchronized

[C#] public bool IsSynchronized {get;}

Implements

ICollection.IsSynchronized

See Also


Copyright (C) 2003 Paul Welter
AffixRuleCollection.Item Property

Gets or sets the element with the specified key.

[Visual Basic] Public Default Property Item(
    ByVal key As String)
) As AffixRule

[C#] public AffixRule this[string key]

Property Value

The key of the element to get or set.

See Also


Copyright (C) 2003 Paul Welter
AffixRuleCollection.Keys Property

gets an ICollection containing the keys of the AffixRuleCollection.

[Visual Basic]

_NotOverridable Public ReadOnly Property Keys As IDictionary.Keys_

[C#]

public System.Collections.ICollection Keys {get;}

Implements

IDictionary.Keys

See Also


Copyright (C) 2003 Paul Welter
**AffixRuleCollection.SyncRoot Property**

Gets an object that can be used to synchronize access to the AffixRuleCollection.

**[Visual Basic]**

```vbnet
NotOverridable Public ReadOnly Property SyncRoot As Implements _
   ICollection.SyncRoot
```

**[C#]**

```csharp
public object SyncRoot {get;}
```

**Implements**

`ICollection.SyncRoot`

**See Also**


---

`Copyright (C) 2003 Paul Welter`
AffixRuleCollection.System.Collections.IDictionary.Item Property

| [Visual Basic] | NotOverridable Private Default _ ByVal key As Object _ ) As Object |
| [C#] | private object this[ object key ] {get; set;} |

See Also


Copyright (C) 2003 Paul Welter
AffixRuleCollection.Values Property

gets an ICollection containing the values in the AffixRuleCollection.

[Visual Basic] NotOverridable Public ReadOnly Property Values As Implements IDictionary.Values

[C#] public System.Collections.ICollection Values

Implements

IDictionary.Values

See Also


Copyright (C) 2003 Paul Welter
## AffixRuleCollection Methods

The methods of the **AffixRuleCollection** class are listed below. For a complete list of **AffixRuleCollection** class members, see the [AffixRuleCollection Members](#) topic.

### Public Static (Shared) Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>Synchronized</code></td>
<td>Returns a synchronized (thread-safe) wrapper for the AffixRuleCollection.</td>
</tr>
</tbody>
</table>

### Public Instance Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>Add</code></td>
<td>adds an element with the provided key and value to the AffixRuleCollection.</td>
</tr>
<tr>
<td><code>Clear</code></td>
<td>removes all elements from the AffixRuleCollection.</td>
</tr>
<tr>
<td><code>Clone</code></td>
<td>Creates a new object that is a copy of the current instance.</td>
</tr>
<tr>
<td><code>Contains</code></td>
<td>Determines whether the AffixRuleCollection contains an element with the specified key.</td>
</tr>
<tr>
<td><code>ContainsKey</code></td>
<td>Determines whether the AffixRuleCollection contains a specific key.</td>
</tr>
<tr>
<td><code>ContainsValue</code></td>
<td>Determines whether the AffixRuleCollection contains a specific value.</td>
</tr>
<tr>
<td><code>CopyTo</code></td>
<td>copies the elements of the AffixRuleCollection to an Array, starting at a particular Array index.</td>
</tr>
<tr>
<td><code>Equals</code> (inherited from <strong>Object</strong>)</td>
<td>Determines whether the specified <strong>Object</strong> is equal to the</td>
</tr>
</tbody>
</table>
### GetEnumerator

Returns an enumerator that can be used to iterate through the 'AffixRuleCollection'.

### GetHashCode (inherited from Object)

Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table.

### GetType (inherited from Object)

Gets the Type of the current instance.

### Remove

Removes the element with the specified key from the AffixRuleCollection.

### ToString (inherited from Object)

Returns a String that represents the current Object.

### Protected Instance Methods

#### Finalize (inherited from Object)

Allows an Object to attempt to free resources and perform other cleanup operations before the Object is reclaimed by garbage collection.

#### MemberwiseClone (inherited from Object)

Creates a shallow copy of the current Object.

### Explicit Interface Implementations

- ICloneable.Clone
- IDictionary.Add
- IDictionary.Contains
- IDictionary.GetEnumerator
- IDictionary.Remove
- IEnumerable.GetEnumerator
See Also


Copyright (C) 2003 Paul Welter
**AffixRuleCollection.Add Method**

adds an element with the provided key and value to the AffixRuleCollection.

```visual-basic
Public Sub Add( _
    ByVal key As String, _
    ByVal value As AffixRule _
)
```

```csharp
public void Add(
    string key,
    AffixRule value
);
```

**Parameters**

- **key**
  
  The string Object to use as the key of the element to add.

- **value**
  
  The AffixRule Object to use as the value of the element to add.

**See Also**


**Copyright (C) 2003 Paul Welter**
removes all elements from the AffixRuleCollection.

[Visual Basic] NotOverridable Public Sub Clear()
  Implements _
  IDictionary.Clear

[C#]
public void Clear();

Implements
  IDictionary.Clear

See Also

Copyright (C) 2003 Paul Welter
AffixRuleCollection.Clone Method

Contains a new object that is a copy of the current instance.

[Visual Basic]
Public Function Clone() As AffixRuleCollection

[C#]
public AffixRuleCollection Clone();

Return Value
A new object that is a copy of this instance.

See Also
Namespace

Copyright (C) 2003 Paul Welter
**AffixRuleCollection.Contains Method**

Determines whether the AffixRuleCollection contains an element with the specified key.

**[Visual Basic]**
```vbnet
Public Function Contains( ByVal key As String ) As Boolean
```

**[C#]**
```csharp
public bool Contains( string key );
```

**Parameters**

*key*

The key to locate in the AffixRuleCollection.

**Return Value**

true if the AffixRuleCollection contains an element with the key; otherwise, false.

**See Also**


---

**Copyright (C) 2003 Paul Welter**
**AffixRuleCollection.ContainsKey Method**

Determines whether the AffixRuleCollection contains a specific key.

**[Visual Basic]**

```
Public Function ContainsKey(
    ByVal key As String
) As Boolean
```

**[C#]**

```
public bool ContainsKey(
    string key
);
```

### Parameters

- **key**
  
  The key to locate in the AffixRuleCollection.

### Return Value

- true if the AffixRuleCollection contains an element with the specified key; otherwise, false.

### See Also


---

*Copyright (C) 2003 Paul Welter*
AffixRuleCollection.ContainsValue Method

Determines whether the AffixRuleCollection contains a specific value.

[Visual Basic] Public Function ContainsValue(
    ByVal value As AffixRule
) As Boolean

[C#] public bool ContainsValue(
    AffixRule value
);

Parameters

value

The value to locate in the AffixRuleCollection. The value can be a null reference (Nothing in Visual Basic).

Return Value

true if the AffixRuleCollection contains an element with the specified value; otherwise, false.

See Also


Copyright (C) 2003 Paul Welter
AffixRuleCollection.CopyTo Method

copies the elements of the AffixRuleCollection to an Array, starting at a particular Array index.

[Visual Basic] NotOverridable Public Sub CopyTo
    ByVal array As _
    ByVal index As _
) Implements _
    ICollection.CopyTo

[C#]
public void CopyTo(
    Array array,
    int index
);

Parameters

array

The one-dimensional Array that is the destination of the elements copied from AffixRuleCollection. The Array must have zero-based indexing.

index

The zero-based index in array at which copying begins.

Implements

IICollection.CopyTo

See Also


Copyright (C) 2003 Paul Welter
AffixRuleCollection.GetEnumerator Method

Returns an enumerator that can be used to iterate through the 'AffixRuleCollection'.

[Visual Basic] Public Function GetEnumerator()

[C#] public AffixRuleEnumerator GetEnumerator();

See Also


Copyright (C) 2003 Paul Welter
AffixRuleCollection.Remove Method

Removes the element with the specified key from the AffixRuleCollection.

[Visual Basic]
Public Sub Remove( _
    ByVal key As String _
)

[C#]
public void Remove(
    string key
);

Parameters

key
    The key of the element to remove

See Also


Copyright (C) 2003 Paul Welter
AffixRuleCollection.Synchronized Method

Returns a synchronized (thread-safe) wrapper for the AffixRuleCollection.

**[Visual Basic]**
```vbnet
Public Shared Function Synchronized(
    ByVal nonSync As AffixRuleCollection
) As AffixRuleCollection
```

**[C#]**
```csharp
public static AffixRuleCollection Synchronized(
    AffixRuleCollection nonSync
);
```

**Parameters**

*nonSync*

The AffixRuleCollection to synchronize.

**See Also**


---

Copyright (C) 2003 Paul Welter
AffixRuleCollection.IDictionary.Add Method

[Visual Basic] Sub Add( _
    ByVal key As Object, _
    ByVal value As Object _
) Implements _
    IDictionary.Add

[C#]

    void IDictionary.Add(
        object key,
        object value
    );

Implements
    IDictionary.Add

See Also
    Namespace

Copyright (C) 2003 Paul Welter
AffixRuleCollection.IDictionary.Contains Method

[Visual Basic] Function Contains( _
    ByVal key As Object _
) As Boolean Implements IDictionary.Contains

[C#]
bool IDictionary.Contains(
    object key
);

Implements
IDictionary.Contains

See Also

Copyright (C) 2003 Paul Welter
AffixRuleCollection.IDictionary.GetEnumerator Method

[Visual Basic] Function GetEnumerator() As IDictionary.GetEnumerator
  Implements IDictionary.GetEnumerator

[C#]
IDictionaryEnumerator IDictionary.GetEnumerator();

Implements
  IDictionary.GetEnumerator

See Also

Copyright (C) 2003 Paul Welter
AffixRuleCollection.IDictionary.Remove Method

**[Visual Basic]**

```vbnet
Sub Remove(ByVal key As Object)
    Implements IDictionary.Remove
End Sub
```

**[C#]**

```csharp
void IDictionary.Remove(object key);
```

**Implements**

`IDictionary.Remove`

**See Also**


---

Copyright (C) 2003 Paul Welter
**AffixRuleCollection.IEnumerable.GetEnumerator Method**

[Visual Basic]```vbnet
Function GetEnumerator() As IEnumerator
  Implements IEnumerable.GetEnumerator
End Function
```

[C#]```csharp
IEnumerator IEnumerable.GetEnumerator();
```

**Implements**

IEnumerable.GetEnumerator

**See Also**


---

Copyright (C) 2003 Paul Welter
AffixRuleCollection.ICloneable.Clone Method

[Visual Basic] Function Clone() As Object Implements _ ICloneable.Clone

[C#] object ICloneable.Clone();

Implements
ICloneable.Clone

See Also

Copyright (C) 2003 Paul Welter
AffixRuleEnumerator Class

A strongly typed enumerator for 'AffixRuleCollection'

For a list of all members of this type, see AffixRuleEnumerator Members.

System.Object  AffixRuleEnumerator

[Visual Basic]
Public Class AffixRuleEnumerator
Implements IDictionaryEnumerator, IEnumerator

[C#]
public class AffixRuleEnumerator :
IDictionaryEnumerator, IEnumerator

Requirements


Assembly: NetSpell.SpellChecker (in NetSpell.SpellChecker.dll)

See Also

AffixRuleEnumerator Members  |

Copyright (C) 2003 Paul Welter
**AffixRuleEnumerator Members**

**AffixRuleEnumerator overview**

**Public Instance Properties**

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Current</strong></td>
<td>Gets the current element from the collection</td>
</tr>
<tr>
<td><strong>Entry</strong></td>
<td>gets both the key and the value of the current AffixRuleCollection entry.</td>
</tr>
<tr>
<td><strong>Key</strong></td>
<td>gets the key of the current AffixRuleCollection entry.</td>
</tr>
<tr>
<td><strong>Value</strong></td>
<td>gets the value of the current AffixRuleCollection entry.</td>
</tr>
</tbody>
</table>

**Public Instance Methods**

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Equals</strong> (inherited from Object)</td>
<td>Determines whether the specified Object is equal to the current Object.</td>
</tr>
<tr>
<td><strong>GetHashCode</strong> (inherited from Object)</td>
<td>Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table.</td>
</tr>
<tr>
<td><strong>GetType</strong> (inherited from Object)</td>
<td>Gets the Type of the current instance.</td>
</tr>
<tr>
<td><strong>MoveNext</strong></td>
<td>Advances the enumerator to the next element of the collection</td>
</tr>
<tr>
<td><strong>Reset</strong></td>
<td>Sets the enumerator to the first element in the collection</td>
</tr>
<tr>
<td><strong>ToString</strong> (inherited from Object)</td>
<td>Returns a String that represents the current Object.</td>
</tr>
</tbody>
</table>

**Protected Instance Methods**
<table>
<thead>
<tr>
<th><strong>Finalize</strong> (inherited from <strong>Object</strong>)</th>
<th>Allows an <strong>Object</strong> to attempt to free resources and perform other cleanup operations before the <strong>Object</strong> is reclaimed by garbage collection.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>MemberwiseClone</strong> (inherited from <strong>Object</strong>)</td>
<td>Creates a shallow copy of the current <strong>Object</strong>.</td>
</tr>
</tbody>
</table>

**Private Instance Properties**

<table>
<thead>
<tr>
<th>System.Collections.IDictionaryEnumerator.Key</th>
</tr>
</thead>
<tbody>
<tr>
<td>System.Collections.IDictionaryEnumerator.Value</td>
</tr>
</tbody>
</table>

**See Also**


Copyright (C) 2003 Paul Welter
AffixRuleEnumerator Properties

The properties of the `AffixRuleEnumerator` class are listed below. For a complete list of `AffixRuleEnumerator` class members, see the `AffixRuleEnumerator Members` topic.

### Public Instance Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Current</strong></td>
<td>Gets the current element from the collection</td>
</tr>
<tr>
<td><strong>Entry</strong></td>
<td>gets both the key and the value of the current AffixRuleCollection entry.</td>
</tr>
<tr>
<td><strong>Key</strong></td>
<td>gets the key of the current AffixRuleCollection entry.</td>
</tr>
<tr>
<td><strong>Value</strong></td>
<td>gets the value of the current AffixRuleCollection entry.</td>
</tr>
</tbody>
</table>

### Private Instance Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>System.Collections.IDictionaryEnumerator.Key</code></td>
<td></td>
</tr>
<tr>
<td><code>System.Collections.IDictionaryEnumerator.Value</code></td>
<td></td>
</tr>
</tbody>
</table>

See Also

AffixRuleEnumerator.Current Property

Gets the current element from the collection

[Visual Basic] NotOverridable Public ReadOnly Property Current As IEnumerator.Current

[C#] public object Current {get;}

Implements
IEumerator.Current

See Also

Copyright (C) 2003 Paul Welter
AffixRuleEnumerator.Entry Property

gets both the key and the value of the current AffixRuleCollection entry.

[Visual Basic] NotOverridable Public ReadOnly Property Entry As Implements _
IDictionaryEnumerator.Entry

[C#]
public System.Collections.DictionaryEntry Entry {get;}

Implements
IDictionaryEnumerator.Entry

See Also

Namespace

Copyright (C) 2003 Paul Welter
AffixRuleEnumerator.Key Property

gets the key of the current AffixRuleCollection entry.

[Visual Basic] Public ReadOnly Property Key As String

[C#] public string Key {get;}

See Also


Copyright (C) 2003 Paul Welter

[Visual Basic] NotOverridable Private ReadOnly Property System.Collections.IDictionaryEnumerator.Key As

[C#]
private object System.Collections.IDictionaryEnumerator.Key {get;}

See Also


Copyright (C) 2003 Paul Welter

[Visual Basic] NotOverridable Private Readonly

[C#] private object System.Collections.IDictionaryEnumerator.Value

See Also


Copyright (C) 2003 Paul Welter
AffixRuleEnumerator.Value Property

gets the value of the current AffixRuleCollection entry.

[Visual Basic] Public ReadOnly Property Value

[C#]
public AffixRule Value {get;}

See Also


Copyright (C) 2003 Paul Welter
AffixRuleEnumerator Methods

The methods of the AffixRuleEnumerator class are listed below. For a complete list of AffixRuleEnumerator class members, see the AffixRuleEnumerator Members topic.

Public Instance Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Equals</strong> (inherited from Object)</td>
<td>Determines whether the specified Object is equal to the current Object.</td>
</tr>
<tr>
<td><strong>GetHashCode</strong> (inherited from Object)</td>
<td>Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table.</td>
</tr>
<tr>
<td><strong>GetType</strong> (inherited from Object)</td>
<td>Gets the Type of the current instance.</td>
</tr>
<tr>
<td><strong>MoveNext</strong></td>
<td>Advances the enumerator to the next element of the collection</td>
</tr>
<tr>
<td><strong>Reset</strong></td>
<td>Sets the enumerator to the first element in the collection</td>
</tr>
<tr>
<td><strong>ToString</strong> (inherited from Object)</td>
<td>Returns a String that represents the current Object.</td>
</tr>
</tbody>
</table>

Protected Instance Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Finalize</strong> (inherited from Object)</td>
<td>Allows an Object to attempt to free resources and perform other cleanup operations before the Object is reclaimed by garbage collection.</td>
</tr>
<tr>
<td><strong>MemberwiseClone</strong> (inherited from Object)</td>
<td>Creates a shallow copy of the current Object.</td>
</tr>
</tbody>
</table>

See Also

## AffixRuleEnumerator.MoveNext Method

Advances the enumerator to the next element of the collection

### [Visual Basic]

```
NotOverridable Public Function MoveNext() Implements _
IEnumerator.MoveNext
```

### [C#]

```
public bool MoveNext();
```

### Implements

- `IEnumerator.MoveNext`

### See Also


---

Copyright (C) 2003 Paul Welter
**AffixRuleEnumerator.Reset Method**

Sets the enumerator to the first element in the collection

**[Visual Basic]**
```
NotOverridable Public Sub Reset()
Implements _
IEnumerator.Reset
```

**[C#]**
```
public void Reset();
```

**Implements**

IEnumerator.Reset

**See Also**


**Copyright (C) 2003 Paul Welter**
AffixUtility Class

Summary description for AffixUtility.

For a list of all members of this type, see AffixUtility Members.

System.Object  AffixUtility

[Visual Basic]
NotInheritable Public Class AffixUtility

[C#]
public sealed class AffixUtility

Requirements

Assembly: NetSpell.SpellChecker (in NetSpell.SpellChecker.dll)

See Also


Copyright (C) 2003 Paul Welter
### AffixUtility Members

#### AffixUtility overview

<table>
<thead>
<tr>
<th>Public Static (Shared) Methods</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>AddPrefix</strong></td>
<td>Adds a prefix to a word</td>
</tr>
<tr>
<td><strong>AddSuffix</strong></td>
<td>Adds a suffix to a word</td>
</tr>
<tr>
<td><strong>EncodeConditions</strong></td>
<td>Generates the condition character array</td>
</tr>
<tr>
<td><strong>RemovePrefix</strong></td>
<td>Removes the affix prefix rule entry for the word if valid</td>
</tr>
<tr>
<td><strong>RemoveSuffix</strong></td>
<td>Removes the affix suffix rule entry for the word if valid</td>
</tr>
</tbody>
</table>

#### Public Instance Constructors

<table>
<thead>
<tr>
<th>Public Instance Constructors</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>AffixUtility Constructor</strong></td>
<td>Initializes a new instance of the class</td>
</tr>
</tbody>
</table>

#### Public Instance Methods

<table>
<thead>
<tr>
<th>Public Instance Methods</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Equals</strong> (inherited from <strong>Object</strong>)</td>
<td>Determines whether the specified <strong>Object</strong> is equal to the current <strong>Object</strong>.</td>
</tr>
<tr>
<td><strong>GetHashCode</strong> (inherited from <strong>Object</strong>)</td>
<td>Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table.</td>
</tr>
<tr>
<td><strong>GetType</strong> (inherited from <strong>Object</strong>)</td>
<td>Gets the <strong>Type</strong> of the current instance.</td>
</tr>
<tr>
<td><strong>ToString</strong> (inherited from <strong>Object</strong>)</td>
<td>Returns a <strong>String</strong> that represents the current <strong>Object</strong>.</td>
</tr>
</tbody>
</table>

#### Protected Instance Methods

<table>
<thead>
<tr>
<th>Protected Instance Methods</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Finalize</strong> (inherited from <strong>Object</strong>)</td>
<td>Allows an <strong>Object</strong> to attempt to</td>
</tr>
<tr>
<td><strong>Object)</strong></td>
<td>free resources and perform other cleanup operations before the <strong>Object</strong> is reclaimed by garbage collection.</td>
</tr>
<tr>
<td><strong>MemberwiseClone</strong> (inherited from <strong>Object)</strong></td>
<td>Creates a shallow copy of the current <strong>Object</strong>.</td>
</tr>
</tbody>
</table>

**See Also**


Copyright (C) 2003 Paul Welter
**AffixUtility Constructor**

Initializes a new instance of the class

```Visual Basic
Public Sub New()
```

```C#
public AffixUtility();
```

See Also

**AffixUtility Methods**

The methods of the **AffixUtility** class are listed below. For a complete list of **AffixUtility** class members, see the **AffixUtility Members** topic.

**Public Static (Shared) Methods**

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AddPrefix</td>
<td>Adds a prefix to a word</td>
</tr>
<tr>
<td>AddSuffix</td>
<td>Adds a suffix to a word</td>
</tr>
<tr>
<td>EncodeConditions</td>
<td>Generates the condition character array</td>
</tr>
<tr>
<td>RemovePrefix</td>
<td>Removes the affix prefix rule entry for the word if valid</td>
</tr>
<tr>
<td>RemoveSuffix</td>
<td>Removes the affix suffix rule entry for the word if valid</td>
</tr>
</tbody>
</table>

**Public Instance Methods**

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Equals (inherited from <strong>Object</strong>)</td>
<td>Determines whether the specified <strong>Object</strong> is equal to the current <strong>Object</strong>.</td>
</tr>
<tr>
<td>GetHashCode (inherited from <strong>Object</strong>)</td>
<td>Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table.</td>
</tr>
<tr>
<td>GetType (inherited from <strong>Object</strong>)</td>
<td>Gets the <strong>Type</strong> of the current instance.</td>
</tr>
<tr>
<td>ToString (inherited from <strong>Object</strong>)</td>
<td>Returns a <strong>String</strong> that represents the current <strong>Object</strong>.</td>
</tr>
</tbody>
</table>

**Protected Instance Methods**

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Finalize (inherited from <strong>Object</strong>)</td>
<td>Allows an <strong>Object</strong> to attempt to free resources and perform other cleanup operations before the <strong>Object</strong> is reclaimed by</td>
</tr>
<tr>
<td><strong>MemberwiseClone</strong> (inherited from <strong>Object</strong>)</td>
<td>Creates a shallow copy of the current <strong>Object</strong>.</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>-----------------------------------------------</td>
</tr>
</tbody>
</table>

**See Also**

- AffixUtility Class

Copyright (C) 2003 Paul Welter
AffixUtility.AddPrefix Method

Adds a prefix to a word

[Visual Basic]
Public Shared Function AddPrefix(
    ByVal word As String, _
    ByVal rule As AffixRule _) As String

[C#]
public static string AddPrefix(
    string word,
    AffixRule rule
);
AffixUtility.AddSuffix Method

Adds a suffix to a word

[Visual Basic]
Public Shared Function AddSuffix(
    ByVal word As String,
    ByVal rule As AffixRule
) As String

[C#]
public static string AddSuffix(
    string word,
    AffixRule rule
);

Parameters

word
The word to get the suffix added to

rule
The AffixRule to use when adding the suffix

Return Value
The word with the suffix added

See Also
Copyright (C) 2003 Paul Welter
Generates the condition character array

[Visual Basic] Public Shared Sub EncodeConditions(_
   ByVal conditionText As String, _
   ByVal entry As AffixEntry _
)

[C#]
public static void EncodeConditions(_
   string conditionText, _
   AffixEntry entry
);

Parameters

conditionText
the text form of the conditions

entry
The AffixEntry to add the condition array to

See Also


Copyright (C) 2003 Paul Welter
AffixUtility.RemovePrefix Method

Removes the affix prefix rule entry for the word if valid

```csharp
public static string RemovePrefix(
    string word, AffixEntry entry
);
```

Parameters

- **word**
  - The word to be modified
- **entry**
  - The affix rule entry to use

Return Value

The word after affix removed. Will be the same word if affix could not be removed.

Remarks

This method does not verify that the returned word is a valid word, only that the affix can be removed

See Also


Copyright (C) 2003 Paul Welter
AffixUtility.RemoveSuffix Method

Removes the affix suffix rule entry for the word if valid

**Parameters**

- `word` The word to be modified
- `entry` The affix rule entry to use

**Return Value**

The word after affix removed. Will be the same word if affix could not be removed.

**Remarks**

This method does not verify that the returned word is a valid word, only that the affix can be removed

**See Also**


---

Copyright (C) 2003 Paul Welter
## Classes

<table>
<thead>
<tr>
<th>Class</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PhoneticRule</td>
<td>This class hold the settings for a phonetic rule</td>
</tr>
<tr>
<td>PhoneticRuleCollection</td>
<td>A collection that stores 'PhoneticRule' objects.</td>
</tr>
<tr>
<td>PhoneticRuleEnumerator</td>
<td>A strongly typed enumerator for 'PhoneticRuleCollection'</td>
</tr>
<tr>
<td>PhoneticUtility</td>
<td>This class holds helper methods for phonetic encoding</td>
</tr>
</tbody>
</table>

*Copyright (C) 2003 Paul Welter*
PhoneticRule Class

This class hold the settings for a phonetic rule
For a list of all members of this type, see PhoneticRule Members.

System.Object PhoneticRule

[Visual Basic]
Public Class PhoneticRule

[C#]
public class PhoneticRule

Requirements

Assembly: NetSpell.SpellChecker (in NetSpell.SpellChecker.dll)

See Also


Copyright (C) 2003 Paul Welter
PhoneticRule Members

**PhoneticRule overview**

**Public Instance Constructors**

<table>
<thead>
<tr>
<th>Constructor</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>PhoneticRule Constructor</strong></td>
<td>Initializes a new instance of the class</td>
</tr>
</tbody>
</table>

**Public Instance Properties**

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>BeginningOnly</strong></td>
<td>True if this rule should be applied to the beginning only</td>
</tr>
<tr>
<td><strong>Condition</strong></td>
<td>The ascii condition array</td>
</tr>
<tr>
<td><strong>ConditionCount</strong></td>
<td>The number of conditions</td>
</tr>
<tr>
<td><strong>ConsumeCount</strong></td>
<td>The number of chars to consume with this rule</td>
</tr>
<tr>
<td><strong>EndOnly</strong></td>
<td>True if this rule should be applied to the end only</td>
</tr>
<tr>
<td><strong>Priority</strong></td>
<td>The priority of this rule</td>
</tr>
<tr>
<td><strong>ReplaceMode</strong></td>
<td>True if this rule should run in replace mode</td>
</tr>
<tr>
<td><strong>ReplaceString</strong></td>
<td>The string to use when replacing</td>
</tr>
</tbody>
</table>

**Public Instance Methods**

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Equals</strong> (inherited from <strong>Object</strong>)</td>
<td>Determines whether the specified <strong>Object</strong> is equal to the current <strong>Object</strong>.</td>
</tr>
<tr>
<td><strong>GetHashCode</strong> (inherited from <strong>Object</strong>)</td>
<td>Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table.</td>
</tr>
<tr>
<td><strong>GetType</strong> (inherited from <strong>Object</strong>)</td>
<td>Gets the <strong>Type</strong> of the current instance.</td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>--------</td>
<td>-------------</td>
</tr>
<tr>
<td><strong>ToString</strong> (inherited from Object)</td>
<td>Returns a <strong>String</strong> that represents the current <strong>Object</strong>.</td>
</tr>
</tbody>
</table>

**Protected Instance Methods**

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Finalize</strong> (inherited from Object)</td>
<td>Allows an <strong>Object</strong> to attempt to free resources and perform other cleanup operations before the <strong>Object</strong> is reclaimed by garbage collection.</td>
</tr>
<tr>
<td><strong>MemberwiseClone</strong> (inherited from Object)</td>
<td>Creates a shallow copy of the current <strong>Object</strong>.</td>
</tr>
</tbody>
</table>

**See Also**

---  

**Copyright (C) 2003 Paul Welter**
PhoneticRule Constructor

Initializes a new instance of the class

[Visual Basic] Public Sub New()

[C#]
public PhoneticRule();

See Also

Copyright (C) 2003 Paul Welter
PhoneticRule Properties

The properties of the **PhoneticRule** class are listed below. For a complete list of **PhoneticRule** class members, see the [PhoneticRule Members](#) topic.

**Public Instance Properties**

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>BeginningOnly</strong></td>
<td>True if this rule should be applied to the beginning only</td>
</tr>
<tr>
<td><strong>Condition</strong></td>
<td>The ascii condition array</td>
</tr>
<tr>
<td><strong>ConditionCount</strong></td>
<td>The number of conditions</td>
</tr>
<tr>
<td><strong>ConsumeCount</strong></td>
<td>The number of chars to consume with this rule</td>
</tr>
<tr>
<td><strong>EndOnly</strong></td>
<td>True if this rule should be applied to the end only</td>
</tr>
<tr>
<td><strong>Priority</strong></td>
<td>The priority of this rule</td>
</tr>
<tr>
<td><strong>ReplaceMode</strong></td>
<td>True if this rule should run in replace mode</td>
</tr>
<tr>
<td><strong>ReplaceString</strong></td>
<td>The string to use when replacing</td>
</tr>
</tbody>
</table>

See Also


---

Copyright (C) 2003 Paul Welter
PhoneticRule.BeginningOnly Property

True if this rule should be applied to the beginning only

[Visual Basic] Public Property BeginningOnly

[C#] public bool BeginningOnly {get; set;}

See Also

Namespace

Copyright (C) 2003 Paul Welter
PhoneticRule.Condition Property

The ascii condition array

[Visual Basic] Public ReadOnly Property Condition As [C#] public int[] Condition {get;}

See Also


Copyright (C) 2003 Paul Welter
**PhoneticRule.ConditionCount Property**

The number of conditions

[Visual Basic] Public Property ConditionCount

[C#] public int ConditionCount {get; set;}

See Also


Copyright (C) 2003 Paul Welter
PhoneticRule.ConsumeCount Property

The number of chars to consume with this rule

[Visual Basic] Public Property ConsumeCount As

[C#]
public int ConsumeCount {get; set;}

See Also


Copyright (C) 2003 Paul Welter
PhoneticRule.EndOnly Property

True if this rule should be applied to the end only

[Visual Basic] Public Property EndOnly As Boolean

[C#] public bool EndOnly {get; set;}

See Also

Copyright (C) 2003 Paul Welter
PhoneticRule.Priority Property

The priority of this rule

[Visual Basic] Public Property Priority As Integer

[C#]
public int Priority {get; set;}

See Also


Copyright (C) 2003 Paul Welter
**PhoneticRule.ReplaceMode Property**

True if this rule should run in replace mode

```[Visual Basic]Public Property ReplaceMode As```

```[C#]public bool ReplaceMode {get; set;}```

See Also


Copyright (C) 2003 Paul Welter
**PhoneticRule.ReplaceString Property**

The string to use when replacing

[Visual Basic] `Public Property ReplaceString` /

[C#] `public string ReplaceString {get; set;}`

See Also


Namespace

Copyright (C) 2003 Paul Welter
PhoneticRuleCollection Class

A collection that stores 'PhoneticRule' objects.

For a list of all members of this type, see PhoneticRuleCollection Members.

System.Object  CollectionBase  PhoneticRuleCollection

[Visual Basic]
Public Class PhoneticRuleCollection
    Inherits CollectionBase

[C#]
public class PhoneticRuleCollection : CollectionBase

Requirements


Assembly: NetSpell.SpellChecker (in NetSpell.SpellChecker.dll)

See Also

PhoneticRuleCollection Members  |

Copyright (C) 2003 Paul Welter
# PhoneticRuleCollection Members

## PhoneticRuleCollection overview

### Public Instance Constructors

| **PhoneticRuleCollection** | Overloaded. Initializes a new instance of the PhoneticRuleCollection class. |

### Public Instance Properties

| **Count** (inherited from CollectionBase) | Gets the number of elements contained in the CollectionBase instance. |

| **Item** | Represents the 'PhonicRule' item at the specified index position. |

### Public Instance Methods

| **Add** | Adds a 'PhonicRule' item with the specified value to the 'PhonicRuleCollection' |

| **AddRange** | Overloaded. Copies the elements of an array at the end of this instance of 'PhonicRuleCollection'. |

| **Clear** (inherited from CollectionBase) | Removes all objects from the CollectionBase instance. |

| **Contains** | Gets a value indicating whether the 'PhonicRuleCollection' contains the specified value. |

<p>| <strong>CopyTo</strong> | Copies the 'PhonicRuleCollection' values to a one-dimensional System.Array instance starting at the specified array index. |</p>
<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Equals</strong> (inherited from <strong>Object</strong>)</td>
<td>Determines whether the specified <strong>Object</strong> is equal to the current <strong>Object</strong>.</td>
</tr>
<tr>
<td><strong>GetEnumerator</strong></td>
<td>Returns an enumerator that can be used to iterate through the 'PhoneticRuleCollection'.</td>
</tr>
<tr>
<td><strong>GetHashCode</strong> (inherited from <strong>Object</strong>)</td>
<td>Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table.</td>
</tr>
<tr>
<td><strong>GetType</strong> (inherited from <strong>Object</strong>)</td>
<td>Gets the <strong>Type</strong> of the current instance.</td>
</tr>
<tr>
<td><strong>IndexOf</strong></td>
<td>Returns the index of a 'PhoneticRule' object in the collection.</td>
</tr>
<tr>
<td><strong>Insert</strong></td>
<td>Inserts an existing 'PhoneticRule' into the collection at the specified index.</td>
</tr>
<tr>
<td><strong>Remove</strong></td>
<td>Removes a specific item from the 'PhoneticRuleCollection'.</td>
</tr>
<tr>
<td><strong>RemoveAt</strong> (inherited from <strong>CollectionBase</strong>)</td>
<td>Removes the element at the specified index of the <strong>CollectionBase</strong> instance.</td>
</tr>
<tr>
<td><strong>ToString</strong> (inherited from <strong>Object</strong>)</td>
<td>Returns a <strong>String</strong> that represents the current <strong>Object</strong>.</td>
</tr>
</tbody>
</table>

**Protected Instance Properties**

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>InnerList</strong> (inherited from <strong>CollectionBase</strong>)</td>
<td>Gets an <strong>ArrayList</strong> containing the list of elements in the <strong>CollectionBase</strong> instance.</td>
</tr>
<tr>
<td><strong>List</strong> (inherited from <strong>CollectionBase</strong>)</td>
<td>Gets an <strong>IList</strong> containing the list of elements in the <strong>CollectionBase</strong> instance.</td>
</tr>
</tbody>
</table>
## Protected Instance Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Finalize</strong> (inherited from <strong>Object</strong>)</td>
<td>Allows an <strong>Object</strong> to attempt to free resources and perform other cleanup operations before the <strong>Object</strong> is reclaimed by garbage collection.</td>
</tr>
<tr>
<td><strong>MemberwiseClone</strong> (inherited from <strong>Object</strong>)</td>
<td>Creates a shallow copy of the current <strong>Object</strong>.</td>
</tr>
<tr>
<td><strong>OnClear</strong> (inherited from <strong>CollectionBase</strong>)</td>
<td>Performs additional custom processes when clearing the contents of the <strong>CollectionBase</strong> instance.</td>
</tr>
<tr>
<td><strong>OnClearComplete</strong> (inherited from <strong>CollectionBase</strong>)</td>
<td>Performs additional custom processes after clearing the contents of the <strong>CollectionBase</strong> instance.</td>
</tr>
<tr>
<td><strong>OnInsert</strong> (inherited from <strong>CollectionBase</strong>)</td>
<td>Performs additional custom processes before inserting a new element into the <strong>CollectionBase</strong> instance.</td>
</tr>
<tr>
<td><strong>OnInsertComplete</strong> (inherited from <strong>CollectionBase</strong>)</td>
<td>Performs additional custom processes after inserting a new element into the <strong>CollectionBase</strong> instance.</td>
</tr>
<tr>
<td><strong>OnRemove</strong> (inherited from <strong>CollectionBase</strong>)</td>
<td>Performs additional custom processes when removing an element from the <strong>CollectionBase</strong> instance.</td>
</tr>
<tr>
<td><strong>OnRemoveComplete</strong> (inherited from <strong>CollectionBase</strong>)</td>
<td>Performs additional custom processes after removing an element from the <strong>CollectionBase</strong> instance.</td>
</tr>
<tr>
<td><strong>OnSet</strong> (inherited from <strong>CollectionBase</strong>)</td>
<td>Performs additional custom processes before setting a</td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>---------------------------------------------------------------------------</td>
</tr>
<tr>
<td><code>OnSetComplete</code> (inherited from <code>CollectionBase</code>)</td>
<td>Performs additional custom processes after setting a value in the <code>CollectionBase</code> instance.</td>
</tr>
<tr>
<td><code>OnValidate</code> (inherited from <code>CollectionBase</code>)</td>
<td>Performs additional custom processes when validating a value.</td>
</tr>
</tbody>
</table>

**Explicit Interface Implementations**

<table>
<thead>
<tr>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>ICollection.CopyTo</code> (inherited from <code>CollectionBase</code>)</td>
<td></td>
</tr>
<tr>
<td><code>IList.Add</code> (inherited from <code>CollectionBase</code>)</td>
<td></td>
</tr>
<tr>
<td><code>IList.Contains</code> (inherited from <code>CollectionBase</code>)</td>
<td></td>
</tr>
<tr>
<td><code>IList.IndexOf</code> (inherited from <code>CollectionBase</code>)</td>
<td></td>
</tr>
<tr>
<td><code>IList.Insert</code> (inherited from <code>CollectionBase</code>)</td>
<td></td>
</tr>
<tr>
<td><code>IList.Remove</code> (inherited from <code>CollectionBase</code>)</td>
<td></td>
</tr>
</tbody>
</table>

**See Also**


---

Copyright (C) 2003 Paul Welter
PhoneticRuleCollection Constructor

Initializes a new instance of 'PhoneticRuleCollection'.

Overload List

Initializes a new instance of 'PhoneticRuleCollection'.

public PhoneticRuleCollection();

Initializes a new instance of 'PhoneticRuleCollection' with an array of 'PhoneticRule' objects.

public PhoneticRuleCollection(PhoneticRule[]);

Initializes a new instance of 'PhoneticRuleCollection' based on an already existing instance.

public PhoneticRuleCollection(PhoneticRuleCollection);

See Also


Copyright (C) 2003 Paul Welter
PhoneticRuleCollection Constructor ()

Initializes a new instance of 'PhoneticRuleCollection'.

[Visual Basic] Overloads Public Sub New()

[C#]
public PhoneticRuleCollection();

See Also


Copyright (C) 2003 Paul Welter
PhoneticRuleCollection Constructor (PhoneticRuleCollection)

Initializes a new instance of 'PhoneticRuleCollection' based on an already existing instance.

[Visual Basic] Overloads Public Sub New(_
    ByVal value As PhoneticRuleCollection _
)

[C#] public PhoneticRuleCollection(_
    PhoneticRuleCollection value _
);

Parameters

value
A 'PhoneticRuleCollection' from which the contents is copied

See Also


Copyright (C) 2003 Paul Welter
PhoneticRuleCollection Constructor (PhoneticRule[])  

Initializes a new instance of 'PhoneticRuleCollection' with an array of 'PhoneticRule' objects.

**Parameters**

*value*  
An array of 'PhoneticRule' objects with which to initialize the collection

**See Also**


Copyright (C) 2003 Paul Welter
PhoneticRuleCollection Properties

The properties of the **PhoneticRuleCollection** class are listed below. For a complete list of **PhoneticRuleCollection** class members, see the **PhoneticRuleCollection Members** topic.

**Public Instance Properties**

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Count (inherited from CollectionBase)</td>
<td>Gets the number of elements contained in the <strong>CollectionBase</strong> instance.</td>
</tr>
<tr>
<td>Item</td>
<td>Represents the 'PhoneticRule' item at the specified index position.</td>
</tr>
</tbody>
</table>

**Protected Instance Properties**

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>InnerList (inherited from CollectionBase)</td>
<td>Gets an <strong>ArrayList</strong> containing the list of elements in the <strong>CollectionBase</strong> instance.</td>
</tr>
<tr>
<td>List (inherited from CollectionBase)</td>
<td>Gets an <strong>IList</strong> containing the list of elements in the <strong>CollectionBase</strong> instance.</td>
</tr>
</tbody>
</table>

**See Also**


---

*Copyright (C) 2003 Paul Welter*
PhoneticRuleCollection.Item Property

Represents the 'PhoneticRule' item at the specified index position.

[Visual Basic] Public Default Property Item(
    ByVal index As Integer
) As PhoneticRule

[C#]
public PhoneticRule this[
    int index
] {get; set;}

Parameters

index
The zero-based index of the entry to locate in the collection.

Property Value
The entry at the specified index of the collection.

See Also

Copyright (C) 2003 Paul Welter
The methods of the **PhoneticRuleCollection** class are listed below. For a complete list of **PhoneticRuleCollection** class members, see the **PhoneticRuleCollection Members** topic.

### Public Instance Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Add</strong></td>
<td>Adds a 'PhoneticRule' item with the specified value to the 'PhoneticRuleCollection'</td>
</tr>
<tr>
<td><strong>AddRange</strong></td>
<td>Overloaded. Copies the elements of an array at the end of this instance of 'PhoneticRuleCollection'.</td>
</tr>
<tr>
<td><strong>Clear</strong></td>
<td>(inherited from CollectionBase) Removes all objects from the <strong>CollectionBase</strong> instance.</td>
</tr>
<tr>
<td><strong>Contains</strong></td>
<td>Gets a value indicating whether the 'PhoneticRuleCollection' contains the specified value.</td>
</tr>
<tr>
<td><strong>CopyTo</strong></td>
<td>Copies the 'PhoneticRuleCollection' values to a one-dimensional System.Array instance starting at the specified array index.</td>
</tr>
<tr>
<td><strong>Equals</strong></td>
<td>(inherited from <strong>Object</strong>) Determines whether the specified <strong>Object</strong> is equal to the current <strong>Object</strong>.</td>
</tr>
<tr>
<td><strong>GetEnumerator</strong></td>
<td>Returns an enumerator that can be used to iterate through the 'PhoneticRuleCollection'.</td>
</tr>
<tr>
<td><strong>GetHashCode</strong></td>
<td>(inherited from <strong>Object</strong>) Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table.</td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>--------</td>
<td>-------------</td>
</tr>
<tr>
<td><strong>Get</strong>&lt;sup&gt;1&lt;/sup&gt; (inherited from <strong>Object</strong>)</td>
<td>Gets the <em>Type</em> of the current instance.</td>
</tr>
<tr>
<td><strong>IndexOf</strong></td>
<td>Returns the index of a 'PhoneticRule' object in the collection.</td>
</tr>
<tr>
<td><strong>Insert</strong></td>
<td>Inserts an existing 'PhoneticRule' into the collection at the specified index.</td>
</tr>
<tr>
<td><strong>Remove</strong></td>
<td>Removes a specific item from the 'PhoneticRuleCollection'.</td>
</tr>
<tr>
<td><strong>RemoveAt</strong> (inherited from <strong>CollectionBase</strong>)</td>
<td>Removes the element at the specified index of the <strong>CollectionBase</strong> instance.</td>
</tr>
<tr>
<td><strong>ToString</strong> (inherited from <strong>Object</strong>)</td>
<td>Returns a <em>String</em> that represents the current <strong>Object</strong>.</td>
</tr>
</tbody>
</table>

### Protected Instance Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Finalize</strong> (inherited from <strong>Object</strong>)</td>
<td>Allows an <strong>Object</strong> to attempt to free resources and perform other cleanup operations before the <strong>Object</strong> is reclaimed by garbage collection.</td>
</tr>
<tr>
<td><strong>MemberwiseClone</strong> (inherited from <strong>Object</strong>)</td>
<td>Creates a shallow copy of the current <strong>Object</strong>.</td>
</tr>
<tr>
<td><strong>OnClear</strong> (inherited from <strong>CollectionBase</strong>)</td>
<td>Performs additional custom processes when clearing the contents of the <strong>CollectionBase</strong> instance.</td>
</tr>
<tr>
<td><strong>OnClearComplete</strong> (inherited from <strong>CollectionBase</strong>)</td>
<td>Performs additional custom processes after clearing the contents of the <strong>CollectionBase</strong> instance.</td>
</tr>
<tr>
<td><strong>OnInsert</strong> (inherited from <strong>CollectionBase</strong>)</td>
<td>Performs additional custom processes before inserting a</td>
</tr>
<tr>
<td>Event Name</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td><strong>OnInsertComplete</strong></td>
<td>Performs additional custom processes after inserting a new element into the <strong>CollectionBase</strong> instance.</td>
</tr>
<tr>
<td><strong>OnRemove</strong></td>
<td>Performs additional custom processes when removing an element from the <strong>CollectionBase</strong> instance.</td>
</tr>
<tr>
<td><strong>OnRemoveComplete</strong></td>
<td>Performs additional custom processes after removing an element from the <strong>CollectionBase</strong> instance.</td>
</tr>
<tr>
<td><strong>OnSet</strong></td>
<td>Performs additional custom processes before setting a value in the <strong>CollectionBase</strong> instance.</td>
</tr>
<tr>
<td><strong>OnSetComplete</strong></td>
<td>Performs additional custom processes after setting a value in the <strong>CollectionBase</strong> instance.</td>
</tr>
<tr>
<td><strong>OnValidate</strong></td>
<td>Performs additional custom processes when validating a value.</td>
</tr>
</tbody>
</table>

**Explicit Interface Implementations**

<table>
<thead>
<tr>
<th>Interface Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>ICollection.CopyTo</strong></td>
<td></td>
</tr>
<tr>
<td><strong>IList.Add</strong></td>
<td></td>
</tr>
<tr>
<td><strong>IList.Contains</strong></td>
<td></td>
</tr>
<tr>
<td><strong>IList.IndexOf</strong></td>
<td></td>
</tr>
<tr>
<td><strong>IList.</strong></td>
<td></td>
</tr>
</tbody>
</table>
IList.Insert (inherited from CollectionBase)

IList.Remove (inherited from CollectionBase)

See Also


Copyright (C) 2003 Paul Welter
PhoneticRuleCollection.Add Method

Adds a 'PhoneticRule' item with the specified value to the 'PhoneticRuleCollection'

[Visual Basic]Public Function Add( _
    ByVal value As PhoneticRule _
) As Integer

[C#]
public int Add(
    PhoneticRule value
);

Parameters

value
    The 'PhoneticRule' to add.

Return Value

The index at which the new element was inserted.

See Also


Copyright (C) 2003 Paul Welter
PhoneticRuleCollection.AddRange Method

Copies the elements of an array at the end of this instance of 'PhoneticRuleCollection'.

Overload List

Copies the elements of an array at the end of this instance of 'PhoneticRuleCollection'.

```csharp
public void AddRange(PhoneticRule[]);
```

Adds the contents of another 'PhoneticRuleCollection' at the end of this instance.

```csharp
public void AddRange(PhoneticRuleCollection);
```

See Also


---

Copyright (C) 2003 Paul Welter
Copies the elements of an array at the end of this instance of 'PhoneticRuleCollection'.

Parameters

value
An array of 'PhoneticRule' objects to add to the collection.

See Also


Copyright (C) 2003 Paul Welter
PhoneticRuleCollection.AddRange Method (PhoneticRuleCollection)

Adds the contents of another 'PhoneticRuleCollection' at the end of this instance.

[Visual Basic]
Overloads Public Sub AddRange(
    ByVal value As PhoneticRuleCollection
)

[C#]
public void AddRange(
    PhoneticRuleCollection value
);

Parameters

value
A 'PhoneticRuleCollection' containing the objects to add to the collection.

See Also


Copyright (C) 2003 Paul Welter
PhoneticRuleCollection.Contains Method

Gets a value indicating whether the 'PhoneticRuleCollection' contains the specified value.

[Visual Basic] Public Function Contains( _
   ByVal value As PhoneticRule _
) As Boolean

[C#] public bool Contains( _
   PhoneticRule value _
);

Parameters

value
   The item to locate.

Return Value

True if the item exists in the collection; false otherwise.

See Also


Copyright (C) 2003 Paul Welter
PhoneticRuleCollection.CopyTo Method

Copies the 'PhoneticRuleCollection' values to a one-dimensional System.Array instance starting at the specified array index.

**[Visual Basic]**

```vbnet
Public Sub CopyTo(_
    ByVal array As PhoneticRule(), _
    ByVal index As Integer)_
```

**[C#]**

```csharp
public void CopyTo(
    PhoneticRule[] array,
    int index
);
```

**Parameters**

array

The one-dimensional System.Array that represents the copy destination.

index

The index in the array where copying begins.

**See Also**


Copyright (C) 2003 Paul Welter
PhoneticRuleCollection.GetEnumerator Method

Returns an enumerator that can be used to iterate through the 'PhoneticRuleCollection'.

[Visual Basic] Public Function GetEnumerator() As New PhoneticRuleEnumerator(GetEnumerator();

[C#] new public PhoneticRuleEnumerator GetEnumerator();

See Also


Copyright (C) 2003 Paul Welter
Returns the index of a 'PhoneticRule' object in the collection.

**Visual Basic**

```vbnet
Public Function IndexOf( ByVal value As PhoneticRule ) As Integer
```

**C#**

```csharp
public int IndexOf( PhoneticRule value );
```

### Parameters

- **value**
  The 'PhoneticRule' object whose index will be retrieved.

### Return Value

If found, the index of the value; otherwise, -1.

### See Also

- [PhoneticRuleCollection Class](#)

---

Copyright (C) 2003 Paul Welter
PhoneticRuleCollection.Insert Method

Inserts an existing 'PhoneticRule' into the collection at the specified index.

[Visual Basic] Public Sub Insert( _
    ByVal index As Integer, _
    ByVal value As PhoneticRule _
)

[C#]
public void Insert(
    int index,
    PhoneticRule value
);

Parameters

index
The zero-based index where the new item should be inserted.

value
The item to insert.

See Also


Copyright (C) 2003 Paul Welter
PhoneticRuleCollection.Remove Method

Removes a specific item from the 'PhoneticRuleCollection'.

[Visual Basic] Public Sub Remove( _
    ByVal value As PhoneticRule _
)

[C#]
public void Remove(
    PhoneticRule value
);

Parameters

value
The item to remove from the 'PhoneticRuleCollection'.

See Also


Copyright (C) 2003 Paul Welter
PhoneticRuleEnumerator Class

A strongly typed enumerator for 'PhoneticRuleCollection'

For a list of all members of this type, see PhoneticRuleEnumerator Members.

**System.Object**  PhoneticRuleEnumerator

[Visual Basic]
Public Class PhoneticRuleEnumerator
Implements IEnumerator

[C#]
public class PhoneticRuleEnumerator : IEnumerator

Requirements

**Namespace:** NetSpell.SpellChecker.Dictionary.Phonetic
**Assembly:** NetSpell.SpellChecker (in NetSpell.SpellChecker.dll)

See Also


Copyright (C) 2003 Paul Welter
## PhoneticRuleEnumerator Members

### PhoneticRuleEnumerator overview

### Public Instance Constructors

<table>
<thead>
<tr>
<th>Constructor</th>
<th>Method</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>PhoneticRuleEnumerator</strong> Constructor</td>
<td>Enumerator constructor</td>
</tr>
</tbody>
</table>

### Public Instance Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Current</strong></td>
<td>Gets the current element from the collection (strongly typed)</td>
</tr>
</tbody>
</table>

### Public Instance Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Equals</strong> (inherited from <strong>Object</strong>)</td>
<td>Determines whether the specified <strong>Object</strong> is equal to the current <strong>Object</strong>.</td>
</tr>
<tr>
<td><strong>GetHashCode</strong> (inherited from <strong>Object</strong>)</td>
<td>Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table.</td>
</tr>
<tr>
<td><strong>GetType</strong> (inherited from <strong>Object</strong>)</td>
<td>Gets the <strong>Type</strong> of the current instance.</td>
</tr>
<tr>
<td><strong>MoveNext</strong></td>
<td>Advances the enumerator to the next element of the collection</td>
</tr>
<tr>
<td><strong>Reset</strong></td>
<td>Sets the enumerator to the first element in the collection</td>
</tr>
<tr>
<td><strong>ToString</strong> (inherited from <strong>Object</strong>)</td>
<td>Returns a <strong>String</strong> that represents the current <strong>Object</strong>.</td>
</tr>
</tbody>
</table>

### Protected Instance Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Finalize</strong> (inherited from <strong>Object</strong>)</td>
<td>Allows an <strong>Object</strong> to attempt to free resources and perform other cleanup operations before the <strong>Object</strong> is reclaimed by</td>
</tr>
</tbody>
</table>
garbage collection.

| **MemberwiseClone** (inherited from **Object**) | Creates a shallow copy of the current **Object**. |

Private Instance Properties

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>System.Collections.IEnumerator.Current</strong></td>
<td></td>
</tr>
</tbody>
</table>

Explicit Interface Implementations

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>IEnumerator.MoveNext</strong></td>
<td>Advances the enumerator to the next element of the collection</td>
</tr>
<tr>
<td><strong>IEnumerator.Reset</strong></td>
<td>Sets the enumerator to the first element in the collection</td>
</tr>
</tbody>
</table>

See Also

- PhoneticRuleEnumerator Class |

Copyright (C) 2003 Paul Welter
PhoneticRuleEnumerator Constructor

Enumerator constructor

[Visual Basic] Public Sub New(  
    ByVal mappings As PhoneticRuleCollection  )

[C#]
public PhoneticRuleEnumerator(  
    PhoneticRuleCollection mappings  );

See Also


Copyright (C) 2003 Paul Welter
PhoneticRuleEnumerator Properties

The properties of the **PhoneticRuleEnumerator** class are listed below. For a complete list of **PhoneticRuleEnumerator** class members, see the [PhoneticRuleEnumerator Members](#) topic.

<table>
<thead>
<tr>
<th>Public Instance Properties</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Current</strong></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Private Instance Properties</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="#" alt="System.Collections.IEnumerator.Current" /></td>
</tr>
</tbody>
</table>

**See Also**


---

[Copyright (C) 2003 Paul Welter](#)
PhoneticRuleEnumerator.Current Property

Gets the current element from the collection (strongly typed)

[Visual Basic] Public ReadOnly Property Current

[C#] public PhoneticRule Current {get;}

See Also


Copyright (C) 2003 Paul Welter

[Visual Basic] NotOverridable Private Readonly

[C#] private object System.Collections.IEnumerator.Current

See Also

PhoneticRuleEnumerator Class |

Copyright (C) 2003 Paul Welter
The methods of the **PhoneticRuleEnumerator** class are listed below. For a complete list of **PhoneticRuleEnumerator** class members, see the **PhoneticRuleEnumerator Members** topic.

### Public Instance Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Equals</strong> (inherited from <strong>Object</strong>)</td>
<td>Determines whether the specified <strong>Object</strong> is equal to the current <strong>Object</strong>.</td>
</tr>
<tr>
<td><strong>GetHashCode</strong> (inherited from <strong>Object</strong>)</td>
<td>Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table.</td>
</tr>
<tr>
<td><strong>GetType</strong> (inherited from <strong>Object</strong>)</td>
<td>Gets the <strong>Type</strong> of the current instance.</td>
</tr>
<tr>
<td><strong>MoveNext</strong></td>
<td>Advances the enumerator to the next element of the collection</td>
</tr>
<tr>
<td><strong>Reset</strong></td>
<td>Sets the enumerator to the first element in the collection</td>
</tr>
<tr>
<td><strong>ToString</strong> (inherited from <strong>Object</strong>)</td>
<td>Returns a <strong>String</strong> that represents the current <strong>Object</strong>.</td>
</tr>
</tbody>
</table>

### Protected Instance Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Finalize</strong> (inherited from <strong>Object</strong>)</td>
<td>Allows an <strong>Object</strong> to attempt to free resources and perform other cleanup operations before the <strong>Object</strong> is reclaimed by garbage collection.</td>
</tr>
<tr>
<td><strong>MemberwiseClone</strong> (inherited from <strong>Object</strong>)</td>
<td>Creates a shallow copy of the current <strong>Object</strong>.</td>
</tr>
</tbody>
</table>

### Explicit Interface Implementations

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>IEnumerator.MoveNext</strong></td>
<td>Advances the enumerator to the</td>
</tr>
<tr>
<td>IEnumerator.Reset</td>
<td>Sets the enumerator to the first element in the collection</td>
</tr>
<tr>
<td>-------------------</td>
<td>----------------------------------------------------------</td>
</tr>
</tbody>
</table>

See Also


---

Copyright (C) 2003 Paul Welter
PhoneticRuleEnumerator.MoveNext Method

Advances the enumerator to the next element of the collection

[Visual Basic] Public Function MoveNext() As Boolean

[C#] public bool MoveNext();

See Also


Copyright (C) 2003 Paul Welter
**PhoneticRuleEnumerator.Reset Method**

Sets the enumerator to the first element in the collection

[Visual Basic] `Public Sub Reset()`

[C#] `public void Reset();`

See Also

- [PhoneticRuleEnumerator Class](#)

Copyright (C) 2003 Paul Welter
PhoneticRuleEnumerator.IEnumerator.MoveNext Method

Advances the enumerator to the next element of the collection

[Visual Basic] Function MoveNext() As Boolean Implements _
IEumerator.MoveNext

[C#] bool IEnumerator.MoveNext();

Implements
IEenumerator.MoveNext

See Also

Copyright (C) 2003 Paul Welter
PhoneticRuleEnumerator.IEnumerator.Reset Method

Sets the enumerator to the first element in the collection

[Visual Basic] Sub Reset() Implements _
IEnumerator.Reset

[C#]
void IEnumerator.Reset();

Implements
IEnumerator.Reset

See Also

Copyright (C) 2003 Paul Welter
PhoneticUtility Class

This class holds helper methods for phonetic encoding
For a list of all members of this type, see PhoneticUtility Members.

System.Object  PhoneticUtility

[Visual Basic]
NotInheritable Public Class PhoneticUtility

[C#]
public sealed class PhoneticUtility

Requirements

Assembly: NetSpell.SpellChecker (in NetSpell.SpellChecker.dll)

See Also


Copyright (C) 2003 Paul Welter
### PhoneticUtility Members

#### PhoneticUtility overview

#### Public Static (Shared) Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>EncodeRule</strong></td>
<td>Converts the rule text in to a PhoneticRule class</td>
</tr>
</tbody>
</table>

#### Public Instance Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Equals</strong></td>
<td>Determines whether the specified <strong>Object</strong> is equal to the current <strong>Object</strong>.</td>
</tr>
<tr>
<td><strong>GetHashCode</strong></td>
<td>Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table.</td>
</tr>
<tr>
<td><strong>GetType</strong></td>
<td>Gets the <strong>Type</strong> of the current instance.</td>
</tr>
<tr>
<td><strong>ToString</strong></td>
<td>Returns a <strong>String</strong> that represents the current <strong>Object</strong>.</td>
</tr>
</tbody>
</table>

#### Protected Instance Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Finalize</strong></td>
<td>Allows an <strong>Object</strong> to attempt to free resources and perform other cleanup operations before the <strong>Object</strong> is reclaimed by garbage collection.</td>
</tr>
<tr>
<td><strong>MemberwiseClone</strong></td>
<td>Creates a shallow copy of the current <strong>Object</strong>.</td>
</tr>
</tbody>
</table>

#### See Also

Copyright (C) 2003 Paul Welter
The methods of the **PhoneticUtility** class are listed below. For a complete list of **PhoneticUtility** class members, see the **PhoneticUtility Members** topic.

### Public Static (Shared) Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>EncodeRule</strong></td>
<td>Converts the rule text in to a PhoneticRule class</td>
</tr>
</tbody>
</table>

### Public Instance Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Equals</strong></td>
<td>Determines whether the specified <strong>Object</strong> is equal to the current <strong>Object</strong>.</td>
</tr>
<tr>
<td><strong>GetHashCode</strong></td>
<td>Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table.</td>
</tr>
<tr>
<td><strong>GetType</strong></td>
<td>Gets the <strong>Type</strong> of the current instance.</td>
</tr>
<tr>
<td><strong>ToString</strong></td>
<td>Returns a <strong>String</strong> that represents the current <strong>Object</strong>.</td>
</tr>
</tbody>
</table>

### Protected Instance Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Finalize</strong></td>
<td>Allows an <strong>Object</strong> to attempt to free resources and perform other cleanup operations before the <strong>Object</strong> is reclaimed by garbage collection.</td>
</tr>
<tr>
<td><strong>MemberwiseClone</strong></td>
<td>Creates a shallow copy of the current <strong>Object</strong>.</td>
</tr>
</tbody>
</table>

### See Also

- **PhoneticUtility Class** | **NetSpell.SpellChecker.Dictionary.Phrasal**
Namespace

Copyright (C) 2003 Paul Welter
**PhoneticUtility.EncodeRule Method**

Converts the rule text in to a PhoneticRule class

```Visual Basic
Public Shared Sub EncodeRule(
    ByVal ruleText As String, _
    ByRef rule As PhoneticRule)
```

```C#
public static void EncodeRule(
    string ruleText, _
    ref PhoneticRule rule)
```

**Parameters**

*ruleText*

The text to convert

*rule*

The object that will hold the conversion data

**See Also**


---

Copyright (C) 2003 Paul Welter
# NetSpell.SpellChecker.Forms Namespace

## Classes

<table>
<thead>
<tr>
<th>Class</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SuggestionForm</td>
<td>The SpellingForm is used to display suggestions when there is a misspelled word</td>
</tr>
</tbody>
</table>

*Copyright (C) 2003 Paul Welter*
The SpellingForm is used to display suggestions when there is a misspelled word.

For a list of all members of this type, see SuggestionForm Members.

### SuggestionForm Class

[Visual Basic]
```vbnet
Public Class SuggestionForm
    Inherits Form
```

[C#]
```csharp
public class SuggestionForm : Form
```

### Requirements

**Namespace:** NetSpell.SpellChecker.Forms  
**Assembly:** NetSpell.SpellChecker (in NetSpell.SpellChecker.dll)

### See Also

SuggestionForm Members | NetSpell.SpellChecker.Forms Namespace

---

Copyright (C) 2003 Paul Welter
# SuggestionForm Members

## SuggestionForm overview

## Public Instance Constructors

<table>
<thead>
<tr>
<th>SuggestionForm Constructor</th>
<th>Default Constructor</th>
</tr>
</thead>
</table>

## Public Instance Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>AcceptButton</strong></td>
<td>(inherited from Form) Gets or sets the button on the form that is clicked when the user presses the ENTER key.</td>
</tr>
<tr>
<td><strong>AccessibilityObject</strong></td>
<td>(inherited from Control) Gets the AccessibleObject assigned to the control.</td>
</tr>
<tr>
<td><strong>AccessibleDefaultActionDescription</strong></td>
<td>(inherited from Control) Gets or sets the default action description of the control for use by accessibility client applications.</td>
</tr>
<tr>
<td><strong>AccessibleDescription</strong></td>
<td>(inherited from Control) Gets or sets the description of the control used by accessibility client applications.</td>
</tr>
<tr>
<td><strong>AccessibleName</strong></td>
<td>(inherited from Control) Gets or sets the name of the control used by accessibility client applications.</td>
</tr>
<tr>
<td><strong>AccessibleRole</strong></td>
<td>(inherited from Control) Gets or sets the accessible role of the control.</td>
</tr>
<tr>
<td><strong>ActiveControl</strong></td>
<td>(inherited from ContainerControl) Gets or sets the active control on the container control.</td>
</tr>
<tr>
<td><strong>ActiveMdiChild</strong></td>
<td>(inherited from Form) Gets the currently active multiple document interface (MDI) child window.</td>
</tr>
<tr>
<td>Property</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td><strong>AllowDrop</strong></td>
<td>(inherited from <strong>Control</strong>) Gets or sets a value indicating whether the control can accept data that the user drags onto it.</td>
</tr>
<tr>
<td><strong>AllowTransparency</strong></td>
<td>(inherited from <strong>Form</strong>) Gets or sets which edges of the control are anchored to the edges of its container.</td>
</tr>
<tr>
<td><strong>Anchor</strong></td>
<td>(inherited from <strong>Control</strong>) Gets or sets which edges of the control are anchored to the edges of its container.</td>
</tr>
<tr>
<td><strong>AutoSize</strong></td>
<td>(inherited from <strong>Form</strong>) Gets or sets a value indicating whether the form adjusts its size to fit the height of the font used on the form and scales its controls.</td>
</tr>
<tr>
<td><strong>AutoSizeBaseSize</strong></td>
<td>(inherited from <strong>Form</strong>) Gets or sets the base size used for autoscaling of the form.</td>
</tr>
<tr>
<td><strong>AutoScroll</strong></td>
<td>(inherited from <strong>Form</strong>) Gets or sets a value indicating whether the form enables autoscrolling.</td>
</tr>
<tr>
<td><strong>AutoScrollMargin</strong></td>
<td>(inherited from <strong>ScrollableControl</strong>) Gets or sets the size of the auto-scroll margin.</td>
</tr>
<tr>
<td><strong>AutoScrollMinSize</strong></td>
<td>(inherited from <strong>ScrollableControl</strong>) Gets or sets the minimum size of the auto-scroll.</td>
</tr>
<tr>
<td><strong>AutoScrollPosition</strong></td>
<td>(inherited from <strong>ScrollableControl</strong>) Gets or sets the location of the auto-scroll position.</td>
</tr>
<tr>
<td><strong>BackColor</strong></td>
<td>(inherited from <strong>Form</strong>) Gets or sets the background color displayed in the control.</td>
</tr>
<tr>
<td><strong>BackgroundImage</strong></td>
<td>(inherited from <strong>Control</strong>) Gets or sets the background image displayed in the control.</td>
</tr>
<tr>
<td><strong>BindingContext</strong></td>
<td>(inherited from <strong>ContainerControl</strong>) Gets the distance between</td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>--------</td>
<td>-------------</td>
</tr>
<tr>
<td><strong>Bounds</strong> (inherited from <strong>Control</strong>)</td>
<td>Gets or sets the size and location of the control including its nonclient elements.</td>
</tr>
<tr>
<td><strong>CancelButton</strong> (inherited from <strong>Form</strong>)</td>
<td>Gets or sets the button control that is clicked when the user presses the ESC key.</td>
</tr>
<tr>
<td><strong>CanFocus</strong> (inherited from <strong>Control</strong>)</td>
<td>Gets a value indicating whether the control can receive focus.</td>
</tr>
<tr>
<td><strong>CanSelect</strong> (inherited from <strong>Control</strong>)</td>
<td>Gets a value indicating whether the control can be selected.</td>
</tr>
<tr>
<td><strong>Capture</strong> (inherited from <strong>Control</strong>)</td>
<td>Gets or sets a value indicating whether the control has captured the mouse.</td>
</tr>
<tr>
<td><strong>CausesValidation</strong> (inherited from <strong>Control</strong>)</td>
<td>Gets or sets a value indicating whether the control causes validation to be performed on any controls that require validation when it receives focus.</td>
</tr>
<tr>
<td><strong>ClientRectangle</strong> (inherited from <strong>Control</strong>)</td>
<td>Gets the rectangle that represents the client area of the control.</td>
</tr>
<tr>
<td><strong>ClientSize</strong> (inherited from <strong>Form</strong>)</td>
<td>Gets or sets the size of the client area of the form.</td>
</tr>
<tr>
<td><strong>CompanyName</strong> (inherited from <strong>Control</strong>)</td>
<td>Gets the name of the company or creator of the application containing the</td>
</tr>
<tr>
<td>Property</td>
<td>Description</td>
</tr>
<tr>
<td>----------</td>
<td>-------------</td>
</tr>
<tr>
<td><code>Container</code> (inherited from Component)</td>
<td>Gets the <code>IContainer</code> that contains the <code>Component</code>.</td>
</tr>
<tr>
<td><code>ContainsFocus</code> (inherited from Control)</td>
<td>Gets a value indicating whether the control, or one of its child controls, currently has the input focus.</td>
</tr>
<tr>
<td><code>ContextMenu</code> (inherited from Control)</td>
<td>Gets or sets the shortcut menu associated with the control.</td>
</tr>
<tr>
<td><code>ControlBox</code> (inherited from Form)</td>
<td>Gets or sets a value indicating whether a control box is displayed in the caption bar of the form.</td>
</tr>
<tr>
<td><code>Controls</code> (inherited from Control)</td>
<td>Gets the collection of controls contained within the control.</td>
</tr>
<tr>
<td><code>Created</code> (inherited from Control)</td>
<td>Gets a value indicating whether the control has been created.</td>
</tr>
<tr>
<td><code>Cursor</code> (inherited from Control)</td>
<td>Gets or sets the cursor that is displayed when the mouse pointer is over the control.</td>
</tr>
<tr>
<td><code>DataBindings</code> (inherited from Control)</td>
<td>Gets the data bindings for the control.</td>
</tr>
<tr>
<td><code>DesktopBounds</code> (inherited from Form)</td>
<td>Gets or sets the size and location of the form on the Windows desktop.</td>
</tr>
<tr>
<td><code>DesktopLocation</code> (inherited from Form)</td>
<td>Gets or sets the location of the form on the Windows desktop.</td>
</tr>
<tr>
<td><code>DialogResult</code> (inherited from Form)</td>
<td>Gets or sets the dialog result for the form.</td>
</tr>
<tr>
<td>Property</td>
<td>Description</td>
</tr>
<tr>
<td>------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>DisplayRectangle</td>
<td>(inherited from ScrollableControl) Gets a value indicating whether the control is in the process of being disposed of.</td>
</tr>
<tr>
<td>Disposing</td>
<td>(inherited from Control) Gets a value indicating whether the control is in the process of being disposed of.</td>
</tr>
<tr>
<td>Dock</td>
<td>(inherited from Control) Gets or sets which edge of the parent container a control is docked to.</td>
</tr>
<tr>
<td>DockPadding</td>
<td>(inherited from ScrollableControl) Gets the dock padding settings for all edges of the control.</td>
</tr>
<tr>
<td>Enabled</td>
<td>(inherited from Control) Gets or sets a value indicating whether the control can respond to user interaction.</td>
</tr>
<tr>
<td>Focused</td>
<td>(inherited from Control) Gets a value indicating whether the control has input focus.</td>
</tr>
<tr>
<td>Font</td>
<td>(inherited from Control) Gets or sets the font of the text displayed by the control.</td>
</tr>
<tr>
<td>ForeColor</td>
<td>(inherited from Control) Gets or sets the foreground color of the control.</td>
</tr>
<tr>
<td>FormBorderStyle</td>
<td>(inherited from Form) Gets or sets the border style of the form.</td>
</tr>
<tr>
<td>Handle</td>
<td>(inherited from Control) Gets the window handle that the control is bound to.</td>
</tr>
<tr>
<td>HasChildren</td>
<td>(inherited from Control) Gets a value indicating whether the control contains one or more child controls.</td>
</tr>
<tr>
<td>Height</td>
<td>(inherited from Control) Gets or sets the height of the control.</td>
</tr>
</tbody>
</table>
| HelpButton             | (inherited from Form) Gets or sets a value indicating whether a Help...
<table>
<thead>
<tr>
<th>Property (inherited from Control)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Icon</strong> (inherited from Form)</td>
<td>Gets or sets the icon for the form.</td>
</tr>
<tr>
<td><strong>ImeMode</strong> (inherited from Control)</td>
<td>Gets or sets the Input Method Editor (IME) mode of the control.</td>
</tr>
<tr>
<td><strong>InvokeRequired</strong> (inherited from Control)</td>
<td>Gets a value indicating whether the caller must call an invoke method when making method calls to the control because the caller is on a different thread than the one the control was created on.</td>
</tr>
<tr>
<td><strong>IsAccessible</strong> (inherited from Control)</td>
<td>Gets or sets a value indicating whether the control is visible to accessibility applications.</td>
</tr>
<tr>
<td><strong>IsDisposed</strong> (inherited from Control)</td>
<td>Gets a value indicating whether the control has been disposed of.</td>
</tr>
<tr>
<td><strong>IsHandleCreated</strong> (inherited from Control)</td>
<td>Gets a value indicating whether the control has a handle associated with it.</td>
</tr>
<tr>
<td><strong>IsMdiChild</strong> (inherited from Form)</td>
<td>Gets a value indicating whether the form is a multiple document interface (MDI) child form.</td>
</tr>
<tr>
<td><strong>IsMdiContainer</strong> (inherited from Form)</td>
<td>Gets or sets a value indicating whether the form is a container for multiple document interface (MDI) child forms.</td>
</tr>
<tr>
<td>Property</td>
<td>Description</td>
</tr>
<tr>
<td>------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td><strong>IsRestrictedWindow</strong></td>
<td>Gets or sets a value indicating whether the form will receive key events</td>
</tr>
<tr>
<td></td>
<td>before the event is passed to the control that has focus.</td>
</tr>
<tr>
<td><strong>KeyPreview</strong></td>
<td>Gets or sets a value indicating whether the form will receive key events</td>
</tr>
<tr>
<td></td>
<td>before the event is passed to the control that has focus.</td>
</tr>
<tr>
<td><strong>Left</strong></td>
<td>Gets or sets the x-coordinate of a control's left edge in pixels.</td>
</tr>
<tr>
<td><strong>Location</strong></td>
<td>Gets or sets the coordinates of the upper-left corner of the control relative</td>
</tr>
<tr>
<td></td>
<td>to the upper-left corner of its container.</td>
</tr>
<tr>
<td><strong>MaximizeBox</strong></td>
<td>Gets or sets a value indicating whether the maximize button is displayed</td>
</tr>
<tr>
<td></td>
<td>in the caption bar of the form.</td>
</tr>
<tr>
<td><strong>MaximumSize</strong></td>
<td>Gets the maximum size the form can be resized to.</td>
</tr>
<tr>
<td><strong>MdiChildren</strong></td>
<td>Gets an array of forms that represent the multiple document interface (MDI)</td>
</tr>
<tr>
<td></td>
<td>child forms that are parented to this form.</td>
</tr>
<tr>
<td><strong>MdiParent</strong></td>
<td>Gets or sets the current multiple document interface (MDI) parent form of this</td>
</tr>
<tr>
<td></td>
<td>form.</td>
</tr>
<tr>
<td><strong>Menu</strong></td>
<td>Gets or sets the <strong>MainMenu</strong> that is displayed in the form.</td>
</tr>
<tr>
<td><strong>MergedMenu</strong></td>
<td>Gets the merged menu for the form.</td>
</tr>
<tr>
<td><strong>MinimizeBox</strong></td>
<td>Gets or sets a value</td>
</tr>
<tr>
<td>Property</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td><strong>Form</strong></td>
<td>indicating whether the minimize button is displayed in the caption bar of the form.</td>
</tr>
<tr>
<td><strong>MinimumSize</strong> (inherited from Form)</td>
<td>Gets or sets the minimum size the form can be resized to.</td>
</tr>
<tr>
<td><strong>Modal</strong> (inherited from Form)</td>
<td>Gets a value indicating whether this form is displayed modally.</td>
</tr>
<tr>
<td><strong>Name</strong> (inherited from Control)</td>
<td>Gets or sets the name of the control.</td>
</tr>
<tr>
<td><strong>Opacity</strong> (inherited from Form)</td>
<td>Gets or sets the opacity level of the form.</td>
</tr>
<tr>
<td><strong>OwnedForms</strong> (inherited from Form)</td>
<td>Gets an array of Form objects that represent all forms that are owned by this form.</td>
</tr>
<tr>
<td><strong>Owner</strong> (inherited from Form)</td>
<td>Gets or sets the form that owns this form.</td>
</tr>
<tr>
<td><strong>Parent</strong> (inherited from Control)</td>
<td>Gets or sets the parent container of the control.</td>
</tr>
<tr>
<td><strong>ParentForm</strong> (inherited from ContainerControl)</td>
<td>Gets the form that the container control is assigned to.</td>
</tr>
<tr>
<td><strong>ProductName</strong> (inherited from Control)</td>
<td>Gets the product name of the assembly containing the control.</td>
</tr>
<tr>
<td><strong>ProductVersion</strong> (inherited from Control)</td>
<td>Gets the version of the assembly containing the control.</td>
</tr>
<tr>
<td><strong>RecreatingHandle</strong> (inherited from Control)</td>
<td>Gets a value indicating whether the control is currently re-creating its</td>
</tr>
<tr>
<td>Property</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------</td>
<td>--------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td><strong>Region</strong></td>
<td>Gets or sets the window region associated with the control.</td>
</tr>
<tr>
<td><strong>Right</strong></td>
<td>Gets the distance between the right edge of the control and the left edge of its container.</td>
</tr>
<tr>
<td><strong>RightToLeft</strong></td>
<td>Gets or sets a value indicating whether control's elements are aligned to support locales using right-to-left fonts.</td>
</tr>
<tr>
<td><strong>ShowInTaskbar</strong></td>
<td>Gets or sets a value indicating whether the form is displayed in the Windows taskbar.</td>
</tr>
<tr>
<td><strong>Site</strong></td>
<td>Gets or sets the site of the control.</td>
</tr>
<tr>
<td><strong>Size</strong></td>
<td>Gets or sets the size of the form.</td>
</tr>
<tr>
<td><strong>SizeGripStyle</strong></td>
<td>Gets or sets the style of the size grip to display in the lower-right corner of the form.</td>
</tr>
<tr>
<td><strong>StartPosition</strong></td>
<td>Gets or sets the starting position of the form at run time.</td>
</tr>
<tr>
<td><strong>TabIndex</strong></td>
<td>Gets or sets a value indicating whether the user can give the focus to this control using the TAB key.</td>
</tr>
<tr>
<td><strong>TabStop</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Tag</strong> (inherited from <strong>Control</strong>)</td>
<td>Gets or sets the object that contains data about the control.</td>
</tr>
<tr>
<td><strong>Text</strong> (inherited from <strong>Control</strong>)</td>
<td>Gets or sets the text associated with this control.</td>
</tr>
<tr>
<td><strong>Top</strong> (inherited from <strong>Control</strong>)</td>
<td>Gets or sets the y-coordinate of the control's top edge in pixels.</td>
</tr>
<tr>
<td><strong>TopLevel</strong> (inherited from <strong>Form</strong>)</td>
<td>Gets or sets a value indicating whether to display the form as a top-level window.</td>
</tr>
<tr>
<td><strong>TopLevelControl</strong> (inherited from <strong>Control</strong>)</td>
<td>Gets the parent control that is not parented by another Windows Forms control. Typically, this is the outermost <strong>Form</strong> that the control is contained in.</td>
</tr>
<tr>
<td><strong>TopMost</strong> (inherited from <strong>Form</strong>)</td>
<td>Gets or sets a value indicating whether the form should be displayed as the top-most form of your application.</td>
</tr>
<tr>
<td><strong>TransparencyKey</strong> (inherited from <strong>Form</strong>)</td>
<td>Gets or sets the color that will represent transparent areas of the form.</td>
</tr>
<tr>
<td><strong>Visible</strong> (inherited from <strong>Control</strong>)</td>
<td>Gets or sets a value indicating whether the control is displayed.</td>
</tr>
<tr>
<td><strong>Width</strong> (inherited from <strong>Control</strong>)</td>
<td>Gets or sets the width of the control.</td>
</tr>
<tr>
<td><strong>WindowState</strong> (inherited from <strong>Form</strong>)</td>
<td>Gets or sets the form's window state.</td>
</tr>
<tr>
<td><strong>WindowTarget</strong> (inherited from</td>
<td></td>
</tr>
<tr>
<td>Public Instance Methods</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------</td>
<td>-------------</td>
</tr>
<tr>
<td>✤ <strong>Activate</strong> (inherited from <strong>Form</strong>)</td>
<td>Activates the form and gives it focus.</td>
</tr>
<tr>
<td>✤ <strong>AddOwnedForm</strong> (inherited from <strong>Form</strong>)</td>
<td>Adds an owned form to this form.</td>
</tr>
<tr>
<td>✤ <strong>BeginInvoke</strong> (inherited from <strong>Control</strong>)</td>
<td>Overloaded. Executes the specified delegate asynchronously with the specified arguments, on the thread that the control's underlying handle was created on.</td>
</tr>
<tr>
<td>✤ <strong>BringToFront</strong> (inherited from <strong>Control</strong>)</td>
<td>Brings the control to the front of the z-order.</td>
</tr>
<tr>
<td>✤ <strong>Close</strong> (inherited from <strong>Form</strong>)</td>
<td>Closes the form.</td>
</tr>
<tr>
<td>✤ <strong>Contains</strong> (inherited from <strong>Control</strong>)</td>
<td>Retrieves a value indicating whether the specified control is a child of the control.</td>
</tr>
<tr>
<td>✤ <strong>CreateControl</strong> (inherited from <strong>Control</strong>)</td>
<td>Forces the creation of the control, including the creation of the handle and any child controls.</td>
</tr>
<tr>
<td>✤ <strong>CreateGraphics</strong> (inherited from <strong>Control</strong>)</td>
<td>Creates the <strong>Graphics</strong> object for the control.</td>
</tr>
<tr>
<td>✤ <strong>CreateObjRef</strong> (inherited from <strong>MarshalByRefObject</strong>)</td>
<td>Creates an object that contains all the relevant information required to generate a proxy used to communicate with a remote object.</td>
</tr>
<tr>
<td>✤ <strong>Dispose</strong> (inherited from <strong>Component</strong>)</td>
<td>Overloaded. Releases all resources used by the <strong>Component</strong>.</td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>DoDragDrop (inherited from Control)</td>
<td>Begins a drag-and-drop operation.</td>
</tr>
<tr>
<td>EndInvoke (inherited from Control)</td>
<td>Retrieves the return value of the asynchronous operation represented by the IAsyncResult object passed.</td>
</tr>
<tr>
<td>Equals (inherited from Object)</td>
<td>Determines whether the specified Object is equal to the current Object.</td>
</tr>
<tr>
<td>FindForm (inherited from Control)</td>
<td>Retrieves the form that the control is on.</td>
</tr>
<tr>
<td>Focus (inherited from Control)</td>
<td>Sets input focus to the control.</td>
</tr>
<tr>
<td>GetChildAtPoint (inherited from Control)</td>
<td>Retrieves the child control that is located at the specified coordinates.</td>
</tr>
<tr>
<td>GetContainerControl (inherited from Control)</td>
<td>Returns the next ContainerControl up the control's chain of parent controls.</td>
</tr>
<tr>
<td>GetHashCode (inherited from Object)</td>
<td>Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table.</td>
</tr>
<tr>
<td>GetLifetimeService (inherited from MarshalByRefObject)</td>
<td>Retrieves the current lifetime service object that controls the lifetime policy for this instance.</td>
</tr>
<tr>
<td>GetNextControl (inherited from Control)</td>
<td>Retrieves the next control forward or back in the tab order of child controls.</td>
</tr>
<tr>
<td>GetType (inherited from Object)</td>
<td>Gets the Type of the current instance.</td>
</tr>
<tr>
<td>Hide (inherited from Control)</td>
<td>Conceals the control from the user.</td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>--------</td>
<td>-------------</td>
</tr>
<tr>
<td><strong>InitializeLifetimeService</strong> (inherited from MarshalByRefObject)</td>
<td>Obtains a lifetime service object to control the lifetime policy for this instance.</td>
</tr>
<tr>
<td><strong>Invalidate</strong> (inherited from Control)</td>
<td>Overloaded. Invalidates the specified region of the control (adds it to the control's update region, which is the area that will be repainted at the next paint operation), and causes a paint message to be sent to the control.</td>
</tr>
<tr>
<td><strong>Invoke</strong> (inherited from Control)</td>
<td>Overloaded. Executes the specified delegate, on the thread that owns the control's underlying window handle, with the specified list of arguments.</td>
</tr>
<tr>
<td><strong>LayoutMdi</strong> (inherited from Form)</td>
<td>Arranges the multiple document interface (MDI) child forms within the MDI parent form.</td>
</tr>
<tr>
<td><strong>PerformLayout</strong> (inherited from Control)</td>
<td>Overloaded. Forces the control to apply layout logic to all its child controls.</td>
</tr>
<tr>
<td><strong>PointToClient</strong> (inherited from Control)</td>
<td>Computes the location of the specified screen point into client coordinates.</td>
</tr>
<tr>
<td><strong>PointToScreen</strong> (inherited from Control)</td>
<td>Computes the location of the specified client point into screen coordinates.</td>
</tr>
<tr>
<td><strong>PreProcessMessage</strong> (inherited from Control)</td>
<td>Preprocesses input messages within the message loop before they are dispatched.</td>
</tr>
<tr>
<td><strong>RectangleToClient</strong> (inherited from Control)</td>
<td>Computes the size and location of the specified screen rectangle in client coordinates.</td>
</tr>
<tr>
<td><strong>Property</strong></td>
<td>Description</td>
</tr>
<tr>
<td>------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>RectangleToScreen (inherited from Control)</td>
<td>Computes the size and location of the specified client rectangle in screen coordinates.</td>
</tr>
<tr>
<td>Refresh (inherited from Control)</td>
<td>Forces the control to invalidate its client area and immediately redraw itself and any child controls.</td>
</tr>
<tr>
<td>RemoveOwnedForm (inherited from Form)</td>
<td>Removes an owned form from this form.</td>
</tr>
<tr>
<td>ResetBackColor (inherited from Control)</td>
<td>Resets the BackColor property to its default value.</td>
</tr>
<tr>
<td>ResetBindings (inherited from Control)</td>
<td>Resets the DataBindings property to its default value.</td>
</tr>
<tr>
<td>ResetCursor (inherited from Control)</td>
<td>Resets the Cursor property to its default value.</td>
</tr>
<tr>
<td>ResetFont (inherited from Control)</td>
<td>Resets the Font property to its default value.</td>
</tr>
<tr>
<td>ResetForeColor (inherited from Control)</td>
<td>Resets the ForeColor property to its default value.</td>
</tr>
<tr>
<td>ResetImeMode (inherited from Control)</td>
<td>Resets the ImeMode property to its default value.</td>
</tr>
<tr>
<td>ResetRightToLeft (inherited from Control)</td>
<td>Resets the RightToLeft property to its default value.</td>
</tr>
<tr>
<td>ResetText (inherited from Control)</td>
<td>Resets the Text property to its default value.</td>
</tr>
<tr>
<td>ResumeLayout (inherited from Control)</td>
<td>Overloaded. Resumes normal layout logic.</td>
</tr>
<tr>
<td>Scale (inherited from Control)</td>
<td>Overloaded. Scales the control and any child controls to the specified ratio.</td>
</tr>
<tr>
<td>ScrollControlIntoView (inherited from Control)</td>
<td></td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>------------------------------------------------------------------------------</td>
</tr>
<tr>
<td><strong>Select</strong> (inherited from Control)</td>
<td>Overloaded. Activates the control.</td>
</tr>
<tr>
<td><strong>SelectNextControl</strong> (inherited from Control)</td>
<td>Activates the next control.</td>
</tr>
<tr>
<td><strong>SendToBack</strong> (inherited from Control)</td>
<td>Sends the control to the back of the z-order.</td>
</tr>
<tr>
<td><strong>SetAutoScrollMargin</strong> (inherited from ScrollableControl)</td>
<td>Sets the size of the auto-scroll margins.</td>
</tr>
<tr>
<td><strong>SetBounds</strong> (inherited from Control)</td>
<td>Overloaded. Sets the bounds of the control to the specified location and size.</td>
</tr>
<tr>
<td><strong>SetDesktopBounds</strong> (inherited from Form)</td>
<td>Sets the bounds of the form in desktop coordinates.</td>
</tr>
<tr>
<td><strong>SetDesktopLocation</strong> (inherited from Form)</td>
<td>Sets the location of the form in desktop coordinates.</td>
</tr>
<tr>
<td><strong>Show</strong> (inherited from Control)</td>
<td>Displays the control to the user.</td>
</tr>
<tr>
<td><strong>ShowDialog</strong> (inherited from Form)</td>
<td>Overloaded. Shows the form as a modal dialog box with no owner window.</td>
</tr>
<tr>
<td><strong>SuspendLayout</strong> (inherited from Control)</td>
<td>Temporarily suspends the layout logic for the control.</td>
</tr>
<tr>
<td><strong>ToString</strong> (inherited from Form)</td>
<td></td>
</tr>
<tr>
<td><strong>Update</strong> (inherited from Control)</td>
<td>Causes the control to redraw the invalidated regions within its client area.</td>
</tr>
<tr>
<td><strong>Validate</strong> (inherited from ContainerControl)</td>
<td>Validates the last invalidated control and its ancestors up through, but not including, the current control.</td>
</tr>
</tbody>
</table>

**Public Instance Events**
<table>
<thead>
<tr>
<th>Event</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Activated</strong> (inherited from Form)</td>
<td>Occurs when the form is activated in code or by the user.</td>
</tr>
<tr>
<td><strong>BackColorChanged</strong> (inherited from Control)</td>
<td>Occurs when the value of the BackColor property changes.</td>
</tr>
<tr>
<td><strong>BackgroundImageChanged</strong> (inherited from Control)</td>
<td>Occurs when the value of the BackgroundImage property changes.</td>
</tr>
<tr>
<td><strong>BindingContextChanged</strong> (inherited from Control)</td>
<td>Occurs when the value of the BindingContext property changes.</td>
</tr>
<tr>
<td><strong>CausesValidationChanged</strong> (inherited from Control)</td>
<td>Occurs when the value of the CausesValidation property changes.</td>
</tr>
<tr>
<td><strong>ChangeUICues</strong> (inherited from Control)</td>
<td>Occurs when the focus or keyboard user interface (UI) cues change.</td>
</tr>
<tr>
<td><strong>Click</strong> (inherited from Control)</td>
<td>Occurs when the control is clicked.</td>
</tr>
<tr>
<td><strong>Closed</strong> (inherited from Form)</td>
<td>Occurs when the form is closed.</td>
</tr>
<tr>
<td><strong>Closing</strong> (inherited from Form)</td>
<td>Occurs when the form is closing.</td>
</tr>
<tr>
<td><strong>ContextMenuChanged</strong> (inherited from Control)</td>
<td>Occurs when the value of the ContextMenu property changes.</td>
</tr>
<tr>
<td><strong>ControlAdded</strong> (inherited from Control)</td>
<td>Occurs when a new control is added to the ControlCollection.</td>
</tr>
<tr>
<td><strong>ControlRemoved</strong> (inherited from Control)</td>
<td>Occurs when a control is removed from the ControlCollection.</td>
</tr>
<tr>
<td><strong>CursorChanged</strong> (inherited from Control)</td>
<td>Occurs when the value of the Cursor property changes.</td>
</tr>
<tr>
<td><strong>Deactivate</strong> (inherited from Form)</td>
<td>Occurs when the form loses focus and is not the active form.</td>
</tr>
<tr>
<td>Event</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td><code>Disposed</code> (inherited from Component)</td>
<td>Adds an event handler to listen to the <code>Disposed</code> event on the component.</td>
</tr>
<tr>
<td><code>DockChanged</code> (inherited from Control)</td>
<td>Occurs when the value of the <code>Dock</code> property changes.</td>
</tr>
<tr>
<td><code>DoubleClick</code> (inherited from Control)</td>
<td>Occurs when the control is double-clicked.</td>
</tr>
<tr>
<td><code>DragDrop</code> (inherited from Control)</td>
<td>Occurs when a drag-and-drop operation is completed.</td>
</tr>
<tr>
<td><code>DragEnter</code> (inherited from Control)</td>
<td>Occurs when an object is dragged into the control's bounds.</td>
</tr>
<tr>
<td><code>DragLeave</code> (inherited from Control)</td>
<td>Occurs when an object is dragged out of the control's bounds.</td>
</tr>
<tr>
<td><code>DragOver</code> (inherited from Control)</td>
<td>Occurs when an object is dragged over the control's bounds.</td>
</tr>
<tr>
<td><code>EnabledChanged</code> (inherited from Control)</td>
<td>Occurs when the <code>Enabled</code> property value has changed.</td>
</tr>
<tr>
<td><code>Enter</code> (inherited from Control)</td>
<td>Occurs when the control is entered.</td>
</tr>
<tr>
<td><code>FontChanged</code> (inherited from Control)</td>
<td>Occurs when the <code>Font</code> property value changes.</td>
</tr>
<tr>
<td><code>ForeColorChanged</code> (inherited from Control)</td>
<td>Occurs when the <code>ForeColor</code> property value changes.</td>
</tr>
<tr>
<td><code>GiveFeedback</code> (inherited from Control)</td>
<td>Occurs during a drag operation.</td>
</tr>
<tr>
<td><code>GotFocus</code> (inherited from Control)</td>
<td>Occurs when the control receives focus.</td>
</tr>
<tr>
<td><code>HandleCreated</code> (inherited from Control)</td>
<td>Occurs when a handle is created for the control.</td>
</tr>
<tr>
<td>Event Name</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>HandleDestroyed (inherited from Control)</td>
<td>Occurs when the control's handle is in the process of being destroyed.</td>
</tr>
<tr>
<td>HelpRequested (inherited from Control)</td>
<td>Occurs when the user requests help for a control.</td>
</tr>
<tr>
<td>ImeModeChanged (inherited from Control)</td>
<td>Occurs when the ImeMode property has changed.</td>
</tr>
<tr>
<td>InputLanguageChanged (inherited from Form)</td>
<td>Occurs after the input language of the form has changed.</td>
</tr>
<tr>
<td>InputLanguageChanging (inherited from Form)</td>
<td>Occurs when the user attempts to change the input language for the form.</td>
</tr>
<tr>
<td>Invalidated (inherited from Control)</td>
<td>Occurs when a control's display requires redrawing.</td>
</tr>
<tr>
<td>KeyDown (inherited from Control)</td>
<td>Occurs when a key is pressed while the control has focus.</td>
</tr>
<tr>
<td>KeyPress (inherited from Control)</td>
<td>Occurs when a key is pressed while the control has focus.</td>
</tr>
<tr>
<td>KeyUp (inherited from Control)</td>
<td>Occurs when a key is released while the control has focus.</td>
</tr>
<tr>
<td>Layout (inherited from Control)</td>
<td>Occurs when a control should reposition its child controls.</td>
</tr>
<tr>
<td>Leave (inherited from Control)</td>
<td>Occurs when the input focus leaves the control.</td>
</tr>
<tr>
<td>Load (inherited from Form)</td>
<td>Occurs before a form is displayed for the first time.</td>
</tr>
<tr>
<td>LocationChanged (inherited from Control)</td>
<td>Occurs when the Location property value has changed.</td>
</tr>
<tr>
<td>LostFocus (inherited from Control)</td>
<td>Occurs when the control loses focus.</td>
</tr>
<tr>
<td>MaximizedBoundsChanged (inherited from Form)</td>
<td>Occurs when the value of the MaximizedBounds property has</td>
</tr>
<tr>
<td>Event Name</td>
<td>Inherited From</td>
</tr>
<tr>
<td>----------------------------</td>
<td>-------------------------</td>
</tr>
<tr>
<td>MaximumSizeChanged</td>
<td>Form</td>
</tr>
<tr>
<td>MdiChildActivate</td>
<td>Form</td>
</tr>
<tr>
<td>MenuComplete</td>
<td>Form</td>
</tr>
<tr>
<td>MenuStart</td>
<td>Form</td>
</tr>
<tr>
<td>MinimumSizeChanged</td>
<td>Form</td>
</tr>
<tr>
<td>MouseDown</td>
<td>Control</td>
</tr>
<tr>
<td>MouseEnter</td>
<td>Control</td>
</tr>
<tr>
<td>MouseHover</td>
<td>Control</td>
</tr>
<tr>
<td>MouseLeave</td>
<td>Control</td>
</tr>
<tr>
<td>MouseMove</td>
<td>Control</td>
</tr>
<tr>
<td>MouseUp</td>
<td>Control</td>
</tr>
<tr>
<td>MouseWheel</td>
<td>Control</td>
</tr>
<tr>
<td>Event Name</td>
<td>Inherited From</td>
</tr>
<tr>
<td>----------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>Move</td>
<td>Control</td>
</tr>
<tr>
<td>Paint</td>
<td>Control</td>
</tr>
<tr>
<td>ParentChanged</td>
<td>Control</td>
</tr>
<tr>
<td>QueryAccessibilityHelp</td>
<td>Control</td>
</tr>
<tr>
<td>QueryContinueDrag</td>
<td>Control</td>
</tr>
<tr>
<td>Resize</td>
<td>Control</td>
</tr>
<tr>
<td>RightToLeftChanged</td>
<td>Control</td>
</tr>
<tr>
<td>SizeChanged</td>
<td>Control</td>
</tr>
<tr>
<td>StyleChanged</td>
<td>Control</td>
</tr>
<tr>
<td>SystemColorsChanged</td>
<td>Control</td>
</tr>
<tr>
<td>TabIndexChanged</td>
<td>Form</td>
</tr>
<tr>
<td>TabStopChanged</td>
<td>Control</td>
</tr>
<tr>
<td>TextChanged</td>
<td>Control</td>
</tr>
<tr>
<td>Validated</td>
<td>Control</td>
</tr>
<tr>
<td>Event</td>
<td>Description</td>
</tr>
<tr>
<td>-------</td>
<td>-------------</td>
</tr>
<tr>
<td>Validating (inherited from Control)</td>
<td>Occurs when the control is validating.</td>
</tr>
<tr>
<td>VisibleChanged (inherited from Control)</td>
<td>Occurs when the Visible property value changes.</td>
</tr>
</tbody>
</table>

Protected Instance Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CreateParams (inherited from Form)</td>
<td></td>
</tr>
<tr>
<td>DefaultImeMode (inherited from Form)</td>
<td>Gets the default Input Method Editor (IME) mode supported by the control.</td>
</tr>
<tr>
<td>DefaultSize (inherited from Form)</td>
<td></td>
</tr>
<tr>
<td>DesignMode (inherited from Component)</td>
<td>Gets a value that indicates whether the Component is currently in design mode.</td>
</tr>
<tr>
<td>Events (inherited from Component)</td>
<td>Gets the list of event handlers that are attached to this Component.</td>
</tr>
<tr>
<td>FontHeight (inherited from Control)</td>
<td>Gets or sets the height of the font of the control.</td>
</tr>
<tr>
<td>HScroll (inherited from ScrollableControl)</td>
<td>Gets or sets a value indicating whether the horizontal scroll bar is visible.</td>
</tr>
<tr>
<td>MaximizedBounds (inherited from Form)</td>
<td>Gets and sets the size of the form when it is maximized.</td>
</tr>
<tr>
<td>RenderRightToLeft (inherited from Control)</td>
<td></td>
</tr>
<tr>
<td>ResizeRedraw (inherited from Control)</td>
<td>Gets or sets a value indicating whether the control redraws itself when resized.</td>
</tr>
<tr>
<td>ShowFocusCues (inherited from Control)</td>
<td>Gets a value indicating whether the control should display focus</td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>--------</td>
<td>-------------</td>
</tr>
<tr>
<td><strong>ShowKeyboardCues</strong> (inherited from Control)</td>
<td>Gets a value indicating whether the control should display keyboard shortcuts.</td>
</tr>
<tr>
<td><strong>VScroll</strong> (inherited from ScrollableControl)</td>
<td>Gets or sets a value indicating whether the vertical scroll bar is visible.</td>
</tr>
</tbody>
</table>

**Protected Instance Methods**

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>AccessibilityNotifyClients</strong> (inherited from Control)</td>
<td>Notifies the accessibility client applications of the specified AccessibleEvents for the specified child control.</td>
</tr>
<tr>
<td><strong>ActivateMdiChild</strong> (inherited from Form)</td>
<td></td>
</tr>
<tr>
<td><strong>AdjustFormScrollbars</strong> (inherited from Form)</td>
<td></td>
</tr>
<tr>
<td><strong>ApplyAutoScaling</strong> (inherited from Form)</td>
<td></td>
</tr>
<tr>
<td><strong>CenterToParent</strong> (inherited from Form)</td>
<td></td>
</tr>
<tr>
<td><strong>CenterToScreen</strong> (inherited from Form)</td>
<td></td>
</tr>
<tr>
<td><strong>CreateAccessibilityInstance</strong> (inherited from Control)</td>
<td>Creates a new accessibility object for the control.</td>
</tr>
<tr>
<td><strong>CreateControlsInstance</strong> (inherited from Form)</td>
<td></td>
</tr>
<tr>
<td><strong>CreateHandle</strong> (inherited from Form)</td>
<td></td>
</tr>
<tr>
<td><strong>DefWndProc</strong> (inherited from Form)</td>
<td></td>
</tr>
<tr>
<td><strong>DestroyHandle</strong> (inherited from Control)</td>
<td>Destroys the handle associated with the control.</td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>--------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Dispose</td>
<td>Overloaded. Clean up any resources being used.</td>
</tr>
<tr>
<td>Finalize</td>
<td>Releases unmanaged resource and performs other cleanup operations before the Component is reclaimed by garbage collection.</td>
</tr>
<tr>
<td>GetScrollState</td>
<td>Returns an object that represents a service provided by the Component or by its Container.</td>
</tr>
<tr>
<td>.GetService</td>
<td>Returns an object that represents a service provided by the Component or by its Container.</td>
</tr>
<tr>
<td>GetStyle</td>
<td>Retrieves the value of the specified control style bit for the control.</td>
</tr>
<tr>
<td>GetTopLevel</td>
<td>Determines if the control is a top-level control.</td>
</tr>
<tr>
<td>InitLayout</td>
<td>Called after the control has been added to another container.</td>
</tr>
<tr>
<td>InvokeGotFocus</td>
<td>Raises the GotFocus event for the specified control.</td>
</tr>
<tr>
<td>InvokeLostFocus</td>
<td>Raises the LostFocus event for the specified control.</td>
</tr>
<tr>
<td>InvokeOnClick</td>
<td>Raises the Click event for the specified control.</td>
</tr>
<tr>
<td>InvokePaint</td>
<td>Raises the Paint event for the specified control.</td>
</tr>
<tr>
<td>InvokePaintBackground</td>
<td>Raises the PaintBackground event for the specified control.</td>
</tr>
<tr>
<td>IsInputChar</td>
<td>Determines if a character is an input character that the control recognizes.</td>
</tr>
<tr>
<td>Property</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>![IsInputKey](inherited from Control)</td>
<td>Determines whether the specified key is a regular input key or a special key that requires preprocessing.</td>
</tr>
<tr>
<td>![MemberwiseClone](inherited from Object)</td>
<td>Creates a shallow copy of the current Object.</td>
</tr>
<tr>
<td>![NotifyInvalidate](inherited from Control)</td>
<td></td>
</tr>
<tr>
<td>![OnActivated](inherited from Form)</td>
<td>Raises the Activated event.</td>
</tr>
<tr>
<td>![OnBackColorChanged](inherited from Control)</td>
<td>Raises the BackColorChanged event.</td>
</tr>
<tr>
<td>![OnBackgroundImageChanged](inherited from Control)</td>
<td>Raises the BackgroundImageChanged event.</td>
</tr>
<tr>
<td>![OnBindingContextChanged](inherited from Control)</td>
<td>Raises the BindingContextChanged event.</td>
</tr>
<tr>
<td>![OnCausesValidationChanged](inherited from Control)</td>
<td>Raises the CausesValidationChanged event.</td>
</tr>
<tr>
<td>![OnChangeUICues](inherited from Control)</td>
<td>Raises the ChangeUICues event.</td>
</tr>
<tr>
<td>![OnClick](inherited from Control)</td>
<td>Raises the Click event.</td>
</tr>
<tr>
<td>![OnClosed](inherited from Form)</td>
<td>Raises the Closed event.</td>
</tr>
<tr>
<td>![OnClosing](inherited from Form)</td>
<td>Raises the Closing event.</td>
</tr>
<tr>
<td>![OnContextMenuChanged](inherited from Control)</td>
<td>Raises the ContextMenuChanged event.</td>
</tr>
<tr>
<td>![OnControlAdded](inherited from Control)</td>
<td>Raises the ControlAdded event.</td>
</tr>
<tr>
<td>![OnControlRemoved](inherited from ContainerControl)</td>
<td></td>
</tr>
<tr>
<td>![OnCreateControl](inherited from Form)</td>
<td></td>
</tr>
<tr>
<td>Event Name</td>
<td>Inherited From</td>
</tr>
<tr>
<td>----------------------------</td>
<td>-------------------------</td>
</tr>
<tr>
<td>OnCursorChanged</td>
<td>Control</td>
</tr>
<tr>
<td>OnDeactivate</td>
<td>Form</td>
</tr>
<tr>
<td>OnDockChanged</td>
<td>Control</td>
</tr>
<tr>
<td>OnDoubleClick</td>
<td>Control</td>
</tr>
<tr>
<td>OnDragDrop</td>
<td>Control</td>
</tr>
<tr>
<td>OnDragEnter</td>
<td>Control</td>
</tr>
<tr>
<td>OnDragLeave</td>
<td>Control</td>
</tr>
<tr>
<td>OnDragOver</td>
<td>Control</td>
</tr>
<tr>
<td>OnEnabledChanged</td>
<td>Control</td>
</tr>
<tr>
<td>OnEnter</td>
<td>Control</td>
</tr>
<tr>
<td>OnFontChanged</td>
<td>Form</td>
</tr>
<tr>
<td>OnForeColorChanged</td>
<td>Control</td>
</tr>
<tr>
<td>OnGiveFeedback</td>
<td>Control</td>
</tr>
<tr>
<td>OnGotFocus</td>
<td>Control</td>
</tr>
<tr>
<td>OnHandleCreated</td>
<td>Form</td>
</tr>
<tr>
<td>OnHandleDestroyed</td>
<td>Form</td>
</tr>
<tr>
<td>Event Name</td>
<td>Inherited From</td>
</tr>
<tr>
<td>----------------------------------------</td>
<td>------------------------</td>
</tr>
<tr>
<td>OnHelpRequested</td>
<td>(inherited from Control)</td>
</tr>
<tr>
<td>OnImeModeChanged</td>
<td>(inherited from Control)</td>
</tr>
<tr>
<td>OnInputLanguageChanged</td>
<td>(inherited from Form)</td>
</tr>
<tr>
<td>OnInputLanguageChanging</td>
<td>(inherited from Form)</td>
</tr>
<tr>
<td>OnInvalidated</td>
<td>(inherited from Control)</td>
</tr>
<tr>
<td>OnKeyDown</td>
<td>(inherited from Control)</td>
</tr>
<tr>
<td>OnKeyPress</td>
<td>(inherited from Control)</td>
</tr>
<tr>
<td>OnKeyUp</td>
<td>(inherited from Control)</td>
</tr>
<tr>
<td>OnLayout</td>
<td>(inherited from ScrollableControl)</td>
</tr>
<tr>
<td>OnLeave</td>
<td>(inherited from Control)</td>
</tr>
<tr>
<td>OnLoad</td>
<td>(inherited from Form)</td>
</tr>
<tr>
<td>OnLocationChanged</td>
<td>(inherited from Control)</td>
</tr>
<tr>
<td>OnLostFocus</td>
<td>(inherited from Control)</td>
</tr>
<tr>
<td>OnMaximizedBoundsChanged</td>
<td>(inherited from Form)</td>
</tr>
<tr>
<td>OnMaximumSizeChanged</td>
<td>(inherited from Form)</td>
</tr>
<tr>
<td>OnMdiChildActivate</td>
<td>(inherited from Form)</td>
</tr>
<tr>
<td>OnMenuComplete</td>
<td>(inherited from Control)</td>
</tr>
<tr>
<td>Event</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td><code>OnMenuStart</code></td>
<td>(inherited from <code>Form</code>) Raises the MenuStart event.</td>
</tr>
<tr>
<td><code>OnMinimumSizeChanged</code></td>
<td>(inherited from <code>Form</code>) Raises the MinimumSizeChanged event.</td>
</tr>
<tr>
<td><code>OnMouseDown</code></td>
<td>(inherited from <code>Control</code>) Raises the MouseDown event.</td>
</tr>
<tr>
<td><code>OnMouseEnter</code></td>
<td>(inherited from <code>Control</code>) Raises the MouseEnter event.</td>
</tr>
<tr>
<td><code>OnMouseHover</code></td>
<td>(inherited from <code>Control</code>) Raises the MouseHover event.</td>
</tr>
<tr>
<td><code>OnMouseLeave</code></td>
<td>(inherited from <code>Control</code>) Raises the MouseLeave event.</td>
</tr>
<tr>
<td><code>OnMouseMove</code></td>
<td>(inherited from <code>Control</code>) Raises the MouseMove event.</td>
</tr>
<tr>
<td><code>OnMouseUp</code></td>
<td>(inherited from <code>Control</code>) Raises the MouseUp event.</td>
</tr>
<tr>
<td><code>OnMouseWheel</code></td>
<td>(inherited from <code>ScrollableControl</code>)</td>
</tr>
<tr>
<td><code>OnMove</code></td>
<td>(inherited from <code>Control</code>) Raises the Move event.</td>
</tr>
<tr>
<td><code>OnNotifyMessage</code></td>
<td>(inherited from <code>Control</code>) Notifies the control of Windows messages.</td>
</tr>
<tr>
<td><code>OnPaint</code></td>
<td>(inherited from <code>Form</code>) Paints the background of the control.</td>
</tr>
<tr>
<td><code>OnPaintBackground</code></td>
<td>(inherited from <code>Control</code>)</td>
</tr>
<tr>
<td></td>
<td>Paints the background of the control.</td>
</tr>
<tr>
<td><code>OnParentBackColorChanged</code></td>
<td>(inherited from <code>Control</code>) Raises the BackColorChanged event when the BackColor property value of the control's container changes.</td>
</tr>
<tr>
<td><code>OnParentBackgroundImageChanged</code></td>
<td>(inherited from <code>Control</code>) Raises the BackgroundImageChanged event.</td>
</tr>
<tr>
<td>Event Name</td>
<td>Description</td>
</tr>
<tr>
<td>------------------------------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>OnParentBindingContextChanged (inherited from Control)</td>
<td>Raises the BindingContextChanged event when the BindingContext property value of the control's container changes.</td>
</tr>
<tr>
<td>OnParentChanged (inherited from Control)</td>
<td>Raises the ParentChanged event.</td>
</tr>
<tr>
<td>OnParentEnabledChanged (inherited from Control)</td>
<td>Raises the EnabledChanged event when the Enabled property value of the control's container changes.</td>
</tr>
<tr>
<td>OnParentFontChanged (inherited from Control)</td>
<td>Raises the FontChanged event when the Font property value of the control's container changes.</td>
</tr>
<tr>
<td>OnParentForeColorChanged (inherited from Control)</td>
<td>Raises the ForeColorChanged event when the ForeColor property value of the control's container changes.</td>
</tr>
<tr>
<td>OnParentRightToLeftChanged (inherited from Control)</td>
<td>Raises the RightToLeftChanged event when the RightToLeft property value of the control's container changes.</td>
</tr>
<tr>
<td>OnParentVisibleChanged (inherited from Control)</td>
<td>Raises the VisibleChanged event when the Visible property value of the control's container changes.</td>
</tr>
<tr>
<td>OnQueryContinueDrag (inherited from Control)</td>
<td>Raises the QueryContinueDrag event.</td>
</tr>
<tr>
<td>OnResize (inherited from Form)</td>
<td></td>
</tr>
<tr>
<td>OnRightToLeftChanged (inherited from Control)</td>
<td>Raises the RightToLeftChanged event when the RightToLeft property value of the control's container changes.</td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------------------------------------</td>
<td>------------------------------------------------------------------</td>
</tr>
<tr>
<td><strong>OnSizeChanged</strong> (inherited from Control)</td>
<td>Raises the <strong>SizeChanged</strong> event.</td>
</tr>
<tr>
<td><strong>OnStyleChanged</strong> (inherited from Form)</td>
<td></td>
</tr>
<tr>
<td><strong>OnSystemColorsChanged</strong> (inherited from Control)</td>
<td>Raises the <strong>SystemColorsChanged</strong> event.</td>
</tr>
<tr>
<td><strong>OnTabIndexChanged</strong> (inherited from Control)</td>
<td>Raises the <strong>TabIndexChanged</strong> event.</td>
</tr>
<tr>
<td><strong>OnTabStopChanged</strong> (inherited from Control)</td>
<td>Raises the <strong>TabStopChanged</strong> event.</td>
</tr>
<tr>
<td><strong>OnTextChanged</strong> (inherited from Form)</td>
<td></td>
</tr>
<tr>
<td><strong>OnValidated</strong> (inherited from Control)</td>
<td>Raises the <strong>Validated</strong> event.</td>
</tr>
<tr>
<td><strong>OnValidating</strong> (inherited from Control)</td>
<td>Raises the <strong>Validating</strong> event.</td>
</tr>
<tr>
<td><strong>OnVisibleChanged</strong> (inherited from Form)</td>
<td></td>
</tr>
<tr>
<td><strong>ProcessCmdKey</strong> (inherited from Form)</td>
<td></td>
</tr>
<tr>
<td><strong>ProcessDialogChar</strong> (inherited from ContainerControl)</td>
<td>Processes a key message and generates the appropriate control events.</td>
</tr>
<tr>
<td><strong>ProcessDialogKey</strong> (inherited from Form)</td>
<td></td>
</tr>
<tr>
<td><strong>ProcessKeyEventArg</strong> (inherited from Control)</td>
<td></td>
</tr>
<tr>
<td><strong>ProcessKeyPreview</strong> (inherited from Form)</td>
<td></td>
</tr>
<tr>
<td><strong>ProcessMnemonic</strong> (inherited from Form)</td>
<td></td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td><strong>ProcessTabKey</strong> (inherited from Form)</td>
<td></td>
</tr>
<tr>
<td><strong>RaiseDragEvent</strong> (inherited from Control)</td>
<td></td>
</tr>
<tr>
<td><strong>RaiseKeyEvent</strong> (inherited from Control)</td>
<td></td>
</tr>
<tr>
<td><strong>RaiseMouseEvent</strong> (inherited from Control)</td>
<td></td>
</tr>
<tr>
<td><strong>RaisePaintEvent</strong> (inherited from Control)</td>
<td></td>
</tr>
<tr>
<td><strong>RecreateHandle</strong> (inherited from Control)</td>
<td>Forces the re-creation of the handle for the control.</td>
</tr>
<tr>
<td><strong>ResetMouseEventArgs</strong> (inherited from Control)</td>
<td></td>
</tr>
<tr>
<td><strong>RtlTranslateAlignment</strong> (inherited from Control)</td>
<td>Overloaded. Converts the specified <a href="#">HorizontalAlignment</a> to the appropriate <a href="#">HorizontalAlignment</a> to support right-to-left text.</td>
</tr>
<tr>
<td><strong>RtlTranslateContent</strong> (inherited from Control)</td>
<td>Converts the specified <a href="#">ContentAlignment</a> to the appropriate <a href="#">ContentAlignment</a> to support right-to-left text.</td>
</tr>
<tr>
<td><strong>RtlTranslateHorizontal</strong> (inherited from Control)</td>
<td>Converts the specified <a href="#">HorizontalAlignment</a> to the appropriate <a href="#">HorizontalAlignment</a> to support right-to-left text.</td>
</tr>
<tr>
<td><strong>RtlTranslateLeftRight</strong> (inherited from Control)</td>
<td>Converts the specified <a href="#">LeftRightAlignment</a> to the appropriate <a href="#">LeftRightAlignment</a> to support right-to-left text.</td>
</tr>
<tr>
<td><strong>ScaleCore</strong> (inherited from Form)</td>
<td></td>
</tr>
<tr>
<td>Method Name</td>
<td>Class</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>------------------------------------------</td>
</tr>
<tr>
<td>Select</td>
<td>Form</td>
</tr>
<tr>
<td>SetBoundsCore</td>
<td>Form</td>
</tr>
<tr>
<td>SetClientRectSizeCore</td>
<td>Form</td>
</tr>
<tr>
<td>SetDisplayRectLocation</td>
<td>ScrollableControl</td>
</tr>
<tr>
<td>SetScrollState</td>
<td>ScrollableControl</td>
</tr>
<tr>
<td>SetStyle</td>
<td>Control</td>
</tr>
<tr>
<td>SetTopLevel</td>
<td>Control</td>
</tr>
<tr>
<td>SetVisibleCore</td>
<td>Form</td>
</tr>
<tr>
<td>UpdateBounds</td>
<td>Control</td>
</tr>
<tr>
<td>UpdateDefaultButton</td>
<td>Form</td>
</tr>
<tr>
<td>UpdateStyles</td>
<td>Control</td>
</tr>
<tr>
<td>UpdateZOrder</td>
<td>Control</td>
</tr>
<tr>
<td>WndProc</td>
<td>Form</td>
</tr>
</tbody>
</table>

**Protected Internal Instance Methods**

<table>
<thead>
<tr>
<th>Method Name</th>
<th>Class</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ProcessKeyMessage</td>
<td>Control</td>
<td>Processes a keyboard message.</td>
</tr>
</tbody>
</table>

**Explicit Interface Implementations**

<table>
<thead>
<tr>
<th>Method Name</th>
<th>Class</th>
<th>Description</th>
</tr>
</thead>
</table>
See Also

SuggestionForm Class | NetSpell.SpellChecker.Forms Namespace

Copyright (C) 2003 Paul Welter
SuggestionForm Constructor

Default Constructor

[Visual Basic] Public Sub New(_
ByVal spell As Spelling _
)

[C#]
public SuggestionForm(
    Spelling spell
);

See Also

SuggestionForm Class | NetSpell.SpellChecker.Forms Namespace

Copyright (C) 2003 Paul Welter
### SuggestionForm Methods

The methods of the **SuggestionForm** class are listed below. For a complete list of **SuggestionForm** class members, see the **SuggestionForm Members** topic.

**Public Instance Methods**

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Activate</strong> (inherited from <strong>Form</strong>)</td>
<td>Activates the form and gives it focus.</td>
</tr>
<tr>
<td><strong>AddOwnedForm</strong> (inherited from <strong>Form</strong>)</td>
<td>Adds an owned form to this form.</td>
</tr>
<tr>
<td><strong>BeginInvoke</strong> (inherited from <strong>Control</strong>)</td>
<td>Overloaded. Executes the specified delegate asynchronously with the specified arguments, on the thread that the control's underlying handle was created on.</td>
</tr>
<tr>
<td><strong>BringToFront</strong> (inherited from <strong>Control</strong>)</td>
<td>Brings the control to the front of the z-order.</td>
</tr>
<tr>
<td><strong>Close</strong> (inherited from <strong>Form</strong>)</td>
<td>Closes the form.</td>
</tr>
<tr>
<td><strong>Contains</strong> (inherited from <strong>Control</strong>)</td>
<td>Retrieves a value indicating whether the specified control is a child of the control.</td>
</tr>
<tr>
<td><strong>CreateControl</strong> (inherited from <strong>Control</strong>)</td>
<td>Forces the creation of the control, including the creation of the handle and any child controls.</td>
</tr>
<tr>
<td><strong>CreateGraphics</strong> (inherited from <strong>Control</strong>)</td>
<td>Creates the <strong>Graphics</strong> object for the control.</td>
</tr>
<tr>
<td><strong>CreateObjRef</strong> (inherited from <strong>MarshalByRefObject</strong>)</td>
<td>Creates an object that contains all the relevant information required to generate a proxy used to communicate with a</td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>-------------</td>
</tr>
<tr>
<td><strong>Dispose</strong> (inherited from Component)</td>
<td>Overloaded. Releases all resources used by the Component.</td>
</tr>
<tr>
<td><strong>DoDragDrop</strong> (inherited from Control)</td>
<td>Begins a drag-and-drop operation.</td>
</tr>
<tr>
<td><strong>EndInvoke</strong> (inherited from Control)</td>
<td>Retrieves the return value of the asynchronous operation represented by the IAsyncResult object passed.</td>
</tr>
<tr>
<td><strong>Equals</strong> (inherited from Object)</td>
<td>Determines whether the specified Object is equal to the current Object.</td>
</tr>
<tr>
<td><strong>FindForm</strong> (inherited from Control)</td>
<td>Retrieves the form that the control is on.</td>
</tr>
<tr>
<td><strong>Focus</strong> (inherited from Control)</td>
<td>Sets input focus to the control.</td>
</tr>
<tr>
<td><strong>GetChildAtPoint</strong> (inherited from Control)</td>
<td>Retrieves the child control that is located at the specified coordinates.</td>
</tr>
<tr>
<td><strong>GetContainerControl</strong> (inherited from Control)</td>
<td>Returns the next ContainerControl up the control's chain of parent controls.</td>
</tr>
<tr>
<td><strong>GetHashCode</strong> (inherited from Object)</td>
<td>Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table.</td>
</tr>
<tr>
<td><strong>GetLifetimeService</strong> (inherited from MarshalByRefObject)</td>
<td>Retrieves the current lifetime service object that controls the lifetime policy for this instance.</td>
</tr>
<tr>
<td><strong>GetNextControl</strong> (inherited from Control)</td>
<td>Retrieves the next control forward or back in the tab order of child controls.</td>
</tr>
<tr>
<td>Method (inherited from Object)</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>-------------</td>
</tr>
<tr>
<td>GetType</td>
<td>Gets the Type of the current instance.</td>
</tr>
<tr>
<td>Hide (inherited from Control)</td>
<td>Conceals the control from the user.</td>
</tr>
<tr>
<td>InitializeLifetimeService (inherited from MarshalByRefObject)</td>
<td>Obtains a lifetime service object to control the lifetime policy for this instance.</td>
</tr>
<tr>
<td>Invalidate (inherited from Control)</td>
<td>Overloaded. Invalidates the specified region of the control (adds it to the control's update region, which is the area that will be repainted at the next paint operation), and causes a paint message to be sent to the control.</td>
</tr>
<tr>
<td>Invoke (inherited from Control)</td>
<td>Overloaded. Executes the specified delegate, on the thread that owns the control's underlying window handle, with the specified list of arguments.</td>
</tr>
<tr>
<td>LayoutMdi (inherited from Form)</td>
<td>Arranges the multiple document interface (MDI) child forms within the MDI parent form.</td>
</tr>
<tr>
<td>PerformLayout (inherited from Control)</td>
<td>Overloaded. Forces the control to apply layout logic to all its child controls.</td>
</tr>
<tr>
<td>PointToClient (inherited from Control)</td>
<td>Computes the location of the specified screen point into client coordinates.</td>
</tr>
<tr>
<td>PointToScreen (inherited from Control)</td>
<td>Computes the location of the specified client point into screen coordinates.</td>
</tr>
<tr>
<td>PreProcessMessage (inherited from Control)</td>
<td>Preprocesses input messages within the message loop before</td>
</tr>
</tbody>
</table>
they are dispatched.

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>RectangleToClient</strong> (inherited from Control)</td>
<td>Computes the size and location of the specified screen rectangle in client coordinates.</td>
</tr>
<tr>
<td><strong>RectangleToScreen</strong> (inherited from Control)</td>
<td>Computes the size and location of the specified client rectangle in screen coordinates.</td>
</tr>
<tr>
<td><strong>Refresh</strong> (inherited from Control)</td>
<td>Forces the control to invalidate its client area and immediately redraw itself and any child controls.</td>
</tr>
<tr>
<td><strong>RemoveOwnedForm</strong> (inherited from Form)</td>
<td>Removes an owned form from this form.</td>
</tr>
<tr>
<td><strong>ResetBackColor</strong> (inherited from Control)</td>
<td>Resets the <strong>BackColor</strong> property to its default value.</td>
</tr>
<tr>
<td><strong>ResetBindings</strong> (inherited from Control)</td>
<td>Resets the <strong>DataBindings</strong> property to its default value.</td>
</tr>
<tr>
<td><strong>ResetCursor</strong> (inherited from Control)</td>
<td>Resets the <strong>Cursor</strong> property to its default value.</td>
</tr>
<tr>
<td><strong>ResetFont</strong> (inherited from Control)</td>
<td>Resets the <strong>Font</strong> property to its default value.</td>
</tr>
<tr>
<td><strong>ResetForeColor</strong> (inherited from Control)</td>
<td>Resets the <strong>ForeColor</strong> property to its default value.</td>
</tr>
<tr>
<td><strong>ResetImeMode</strong> (inherited from Control)</td>
<td>Resets the <strong>ImeMode</strong> property to its default value.</td>
</tr>
<tr>
<td><strong>ResetRightToLeft</strong> (inherited from Control)</td>
<td>Resets the <strong>RightToLeft</strong> property to its default value.</td>
</tr>
<tr>
<td><strong>ResetText</strong> (inherited from Control)</td>
<td>Resets the <strong>Text</strong> property to its default value.</td>
</tr>
<tr>
<td><strong>ResumeLayout</strong> (inherited from Control)</td>
<td>Overloaded. Resumes normal layout logic.</td>
</tr>
<tr>
<td><strong>Scale</strong> (inherited from Control)</td>
<td>Overloaded. Scales the control</td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>--------</td>
<td>-------------</td>
</tr>
<tr>
<td><code>ScrollControlIntoView</code> (inherited from <code>ScrollableControl</code></td>
<td>and any child controls to the specified ratio.</td>
</tr>
<tr>
<td><code>Select</code> (inherited from <code>Control</code></td>
<td>Overloaded. Activates the control.</td>
</tr>
<tr>
<td><code>SelectNextControl</code> (inherited from <code>Control</code></td>
<td>Activates the next control.</td>
</tr>
<tr>
<td><code>SendToBack</code> (inherited from <code>Control</code></td>
<td>Sends the control to the back of the z-order.</td>
</tr>
<tr>
<td><code>SetAutoScrollMargin</code> (inherited from <code>ScrollableControl</code></td>
<td>Sets the size of the auto-scroll margins.</td>
</tr>
<tr>
<td><code>SetBounds</code> (inherited from <code>Control</code></td>
<td>Overloaded. Sets the bounds of the control to the specified location and size.</td>
</tr>
<tr>
<td><code>SetDesktopBounds</code> (inherited from <code>Form</code></td>
<td>Sets the bounds of the form in desktop coordinates.</td>
</tr>
<tr>
<td><code>SetDesktopLocation</code> (inherited from <code>Form</code></td>
<td>Sets the location of the form in desktop coordinates.</td>
</tr>
<tr>
<td><code>Show</code> (inherited from <code>Control</code></td>
<td>Displays the control to the user.</td>
</tr>
<tr>
<td><code>ShowDialog</code> (inherited from <code>Form</code></td>
<td>Overloaded. Shows the form as a modal dialog box with no owner window.</td>
</tr>
<tr>
<td><code>SuspendLayout</code> (inherited from <code>Control</code></td>
<td>Temporarily suspends the layout logic for the control.</td>
</tr>
<tr>
<td><code>ToString</code> (inherited from <code>Form</code></td>
<td></td>
</tr>
<tr>
<td><code>Update</code> (inherited from <code>Control</code></td>
<td>Causes the control to redraw the invalidated regions within its client area.</td>
</tr>
<tr>
<td><code>Validate</code> (inherited from <code>ContainerControl</code></td>
<td>Validates the last invalidated control and its ancestors up</td>
</tr>
</tbody>
</table>
through, but not including, the current control.

## Protected Instance Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image" alt="AccessibilityNotifyClients" /> (inherited from Control)</td>
<td>Notifies the accessibility client applications of the specified AccessibleEvents for the specified child control.</td>
</tr>
<tr>
<td><img src="image" alt="ActivateMdiChild" /> (inherited from Form)</td>
<td></td>
</tr>
<tr>
<td><img src="image" alt="AdjustFormScrollbars" /> (inherited from Form)</td>
<td></td>
</tr>
<tr>
<td><img src="image" alt="ApplyAutoScaling" /> (inherited from Form)</td>
<td></td>
</tr>
<tr>
<td><img src="image" alt="CenterToParent" /> (inherited from Form)</td>
<td></td>
</tr>
<tr>
<td><img src="image" alt="CenterToScreen" /> (inherited from Form)</td>
<td></td>
</tr>
<tr>
<td><img src="image" alt="CreateAccessibilityInstance" /> (inherited from Control)</td>
<td>Creates a new accessibility object for the control.</td>
</tr>
<tr>
<td><img src="image" alt="CreateControlsInstance" /> (inherited from Form)</td>
<td></td>
</tr>
<tr>
<td><img src="image" alt="CreateHandle" /> (inherited from Form)</td>
<td></td>
</tr>
<tr>
<td><img src="image" alt="DefWndProc" /> (inherited from Form)</td>
<td></td>
</tr>
<tr>
<td><img src="image" alt="DestroyHandle" /> (inherited from Control)</td>
<td>Destroys the handle associated with the control.</td>
</tr>
<tr>
<td><img src="image" alt="Dispose" /></td>
<td>Overloaded. Clean up any resources being used.</td>
</tr>
<tr>
<td><img src="image" alt="Finalize" /> (inherited from Component)</td>
<td>Releases unmanaged resources and performs other cleanup operations before the</td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Component is reclaimed by garbage collection.</td>
<td></td>
</tr>
<tr>
<td><strong>GetScrollState</strong> (inherited from ScrollableControl)</td>
<td></td>
</tr>
<tr>
<td><strong>GetService</strong> (inherited from Component)</td>
<td>Returns an object that represents a service provided by the Component or by its Container.</td>
</tr>
<tr>
<td><strong>GetStyle</strong> (inherited from Control)</td>
<td>Retrieves the value of the specified control style bit for the control.</td>
</tr>
<tr>
<td><strong>GetTopLevel</strong> (inherited from Control)</td>
<td>Determines if the control is a top-level control.</td>
</tr>
<tr>
<td><strong>InitLayout</strong> (inherited from Control)</td>
<td>Called after the control has been added to another container.</td>
</tr>
<tr>
<td><strong>InvokeGotFocus</strong> (inherited from Control)</td>
<td>Raises the GotFocus event for the specified control.</td>
</tr>
<tr>
<td><strong>InvokeLostFocus</strong> (inherited from Control)</td>
<td>Raises the LostFocus event for the specified control.</td>
</tr>
<tr>
<td><strong>InvokeOnClick</strong> (inherited from Control)</td>
<td>Raises the Click event for the specified control.</td>
</tr>
<tr>
<td><strong>InvokePaint</strong> (inherited from Control)</td>
<td>Raises the Paint event for the specified control.</td>
</tr>
<tr>
<td><strong>InvokePaintBackground</strong> (inherited from Control)</td>
<td>Raises the PaintBackground event for the specified control.</td>
</tr>
<tr>
<td><strong>IsInputChar</strong> (inherited from Control)</td>
<td>Determines if a character is an input character that the control recognizes.</td>
</tr>
<tr>
<td><strong>IsInputKey</strong> (inherited from Control)</td>
<td>Determines whether the specified key is a regular input key or a special key that requires preprocessing.</td>
</tr>
<tr>
<td>Method (inherited from)</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------------------------------------------------</td>
<td>-------------------------------------------------------</td>
</tr>
<tr>
<td><strong>MemberwiseClone</strong> (inherited from Object)</td>
<td>Creates a shallow copy of the current Object.</td>
</tr>
<tr>
<td><strong>NotifyInvalidate</strong> (inherited from Control)</td>
<td></td>
</tr>
<tr>
<td><strong>OnActivated</strong> (inherited from Form)</td>
<td>Raises the Activated event.</td>
</tr>
<tr>
<td><strong>OnBackColorChanged</strong> (inherited from Control)</td>
<td>Raises the BackColorChanged event.</td>
</tr>
<tr>
<td><strong>OnBackgroundImageChanged</strong> (inherited from Control)</td>
<td>Raises the BackgroundImageChanged event.</td>
</tr>
<tr>
<td><strong>OnBindingContextChanged</strong> (inherited from Control)</td>
<td>Raises the BindingContextChanged event.</td>
</tr>
<tr>
<td><strong>OnCausesValidationChanged</strong> (inherited from Control)</td>
<td>Raises the CausesValidationChanged event.</td>
</tr>
<tr>
<td><strong>OnChangeUICues</strong> (inherited from Control)</td>
<td>Raises the ChangeUICues event.</td>
</tr>
<tr>
<td><strong>OnClick</strong> (inherited from Control)</td>
<td>Raises the Click event.</td>
</tr>
<tr>
<td><strong>OnClosed</strong> (inherited from Form)</td>
<td>Raises the Closed event.</td>
</tr>
<tr>
<td><strong>OnClosing</strong> (inherited from Form)</td>
<td>Raises the Closing event.</td>
</tr>
<tr>
<td><strong>OnContextMenuChanged</strong> (inherited from Control)</td>
<td>Raises the ContextMenuChanged event.</td>
</tr>
<tr>
<td><strong>OnControlAdded</strong> (inherited from Control)</td>
<td>Raises the ControlAdded event.</td>
</tr>
<tr>
<td><strong>OnControlRemoved</strong> (inherited from ContainerControl)</td>
<td></td>
</tr>
<tr>
<td><strong>OnCreateControl</strong> (inherited from Form)</td>
<td></td>
</tr>
<tr>
<td><strong>OnCursorChanged</strong> (inherited from Control)</td>
<td>Raises the CursorChanged event.</td>
</tr>
<tr>
<td><strong>OnDeactivate</strong> (inherited from Form)</td>
<td>Raises the Deactivate event.</td>
</tr>
<tr>
<td>Event</td>
<td>Description</td>
</tr>
<tr>
<td>-------</td>
<td>-------------</td>
</tr>
<tr>
<td>OnDockChanged</td>
<td>(inherited from Control) Raises the DockChanged event.</td>
</tr>
<tr>
<td>OnDoubleClick</td>
<td>(inherited from Control) Raises the DoubleClick event.</td>
</tr>
<tr>
<td>OnDragDrop</td>
<td>(inherited from Control) Raises the DragDrop event.</td>
</tr>
<tr>
<td>OnDragEnter</td>
<td>(inherited from Control) Raises the DragEnter event.</td>
</tr>
<tr>
<td>OnDragLeave</td>
<td>(inherited from Control) Raises the DragLeave event.</td>
</tr>
<tr>
<td>OnDragOver</td>
<td>(inherited from Control) Raises the DragOver event.</td>
</tr>
<tr>
<td>OnEnabledChanged</td>
<td>(inherited from Control) Raises the EnabledChanged event.</td>
</tr>
<tr>
<td>OnEnter</td>
<td>(inherited from Control) Raises the Enter event.</td>
</tr>
<tr>
<td>OnFontChanged</td>
<td>(inherited from Form) Raises the ForeColorChanged event.</td>
</tr>
<tr>
<td>OnForeColorChanged</td>
<td>(inherited from Control) Raises the ForeColorChanged event.</td>
</tr>
<tr>
<td>OnGiveFeedback</td>
<td>(inherited from Control) Raises the GiveFeedback event.</td>
</tr>
<tr>
<td>OnGotFocus</td>
<td>(inherited from Control) Raises the GotFocus event.</td>
</tr>
<tr>
<td>OnHandleCreated</td>
<td>(inherited from Form)</td>
</tr>
<tr>
<td>OnHandleDestroyed</td>
<td>(inherited from Form)</td>
</tr>
<tr>
<td>OnHelpRequested</td>
<td>(inherited from Control) Raises the HelpRequested event.</td>
</tr>
<tr>
<td>OnImeModeChanged</td>
<td>(inherited from Control) Raises the ImeModeChanged event.</td>
</tr>
<tr>
<td>Event Name</td>
<td>Description</td>
</tr>
<tr>
<td>------------------------------------------------</td>
<td>-------------------------------------------------------</td>
</tr>
<tr>
<td>OnInputLanguageChanged</td>
<td>Raises the <code>InputLanguageChanged</code> event.</td>
</tr>
<tr>
<td>OnInputLanguageChanging</td>
<td>Raises the <code>InputLanguageChanging</code> event.</td>
</tr>
<tr>
<td>OnInvalidated</td>
<td>Raises the <code>Invalidated</code> event.</td>
</tr>
<tr>
<td>OnKeyDown</td>
<td>Raises the <code>KeyDown</code> event.</td>
</tr>
<tr>
<td>OnKeyPress</td>
<td>Raises the <code>KeyPress</code> event.</td>
</tr>
<tr>
<td>OnKeyUp</td>
<td>Raises the <code>KeyUp</code> event.</td>
</tr>
<tr>
<td>OnLayout</td>
<td>Raises the <code>KeyPress</code> event.</td>
</tr>
<tr>
<td>OnLeave</td>
<td>Raises the <code>Leave</code> event.</td>
</tr>
<tr>
<td>OnLoad</td>
<td>Raises the <code>Load</code> event.</td>
</tr>
<tr>
<td>OnLocationChanged</td>
<td>Raises the <code>LocationChanged</code> event.</td>
</tr>
<tr>
<td>OnLostFocus</td>
<td>Raises the <code>LostFocus</code> event.</td>
</tr>
<tr>
<td>OnMaximizedBoundsChanged</td>
<td>Raises the <code>MaximizedBoundsChanged</code> event.</td>
</tr>
<tr>
<td>OnMaximumSizeChanged</td>
<td>Raises the <code>MaximumSizeChanged</code> event.</td>
</tr>
<tr>
<td>OnMdiChildActivate</td>
<td>Raises the <code>MdiChildActivate</code> event.</td>
</tr>
<tr>
<td>OnMenuComplete</td>
<td>Raises the <code>MenuComplete</code> event.</td>
</tr>
<tr>
<td>OnMenuStart</td>
<td>Raises the <code>MenuStart</code> event.</td>
</tr>
<tr>
<td>OnMinimumSizeChanged</td>
<td>Raises the <code>MinimumSizeChanged</code> event.</td>
</tr>
<tr>
<td>Event (inherited from)</td>
<td>Description</td>
</tr>
<tr>
<td>------------------------</td>
<td>-------------</td>
</tr>
<tr>
<td><strong>OnMouseDown</strong> (inherited from Control)</td>
<td>Raises the <strong>MouseDown</strong> event.</td>
</tr>
<tr>
<td><strong>OnMouseEnter</strong> (inherited from Control)</td>
<td>Raises the <strong>MouseEnter</strong> event.</td>
</tr>
<tr>
<td><strong>OnMouseHover</strong> (inherited from Control)</td>
<td>Raises the <strong>MouseHover</strong> event.</td>
</tr>
<tr>
<td><strong>OnMouseLeave</strong> (inherited from Control)</td>
<td>Raises the <strong>MouseLeave</strong> event.</td>
</tr>
<tr>
<td><strong>OnMouseMove</strong> (inherited from Control)</td>
<td>Raises the <strong>MouseMove</strong> event.</td>
</tr>
<tr>
<td><strong>OnMouseUp</strong> (inherited from Control)</td>
<td>Raises the <strong>MouseUp</strong> event.</td>
</tr>
<tr>
<td><strong>OnMouseWheel</strong> (inherited from ScrollableControl)</td>
<td>Raises the <strong>Move</strong> event.</td>
</tr>
<tr>
<td><strong>OnNotifyMessage</strong> (inherited from Control)</td>
<td>Notifies the control of Windows messages.</td>
</tr>
<tr>
<td><strong>OnPaint</strong> (inherited from Form)</td>
<td></td>
</tr>
<tr>
<td><strong>OnPaintBackground</strong> (inherited from Control)</td>
<td>Paints the background of the control.</td>
</tr>
<tr>
<td><strong>OnParentBackColorChanged</strong> (inherited from Control)</td>
<td>Raises the <strong>BackColorChanged</strong> event when the BackColor property value of the control's container changes.</td>
</tr>
<tr>
<td><strong>OnParentBackgroundImageChanged</strong> (inherited from Control)</td>
<td>Raises the <strong>BackgroundImageChanged</strong> event when the BackgroundImage property value of the control's container changes.</td>
</tr>
<tr>
<td><strong>OnParentBindingContextChanged</strong></td>
<td>Raises the</td>
</tr>
<tr>
<td>Event Name</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------------------</td>
<td>---------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td><strong>(inherited from Control)</strong></td>
<td><strong>BindingContextChanged</strong> event when the BindingContext property value of the control's container changes.</td>
</tr>
<tr>
<td><strong>OnParentChanged</strong> (inherited from Control)</td>
<td>Raises the ParentChanged event.</td>
</tr>
<tr>
<td><strong>OnParentEnabledChanged</strong> (inherited from Control)</td>
<td>Raises the EnabledChanged event when the Enabled property value of the control's container changes.</td>
</tr>
<tr>
<td><strong>OnParentFontChanged</strong> (inherited from Control)</td>
<td>Raises the FontChanged event when the Font property value of the control's container changes.</td>
</tr>
<tr>
<td><strong>OnParentForeColorChanged</strong> (inherited from Control)</td>
<td>Raises the ForeColorChanged event when the ForeColor property value of the control's container changes.</td>
</tr>
<tr>
<td><strong>OnParentRightToLeftChanged</strong> (inherited from Control)</td>
<td>Raises the RightToLeftChanged event when the RightToLeft property value of the control's container changes.</td>
</tr>
<tr>
<td><strong>OnParentVisibleChanged</strong> (inherited from Control)</td>
<td>Raises the VisibleChanged event when the Visible property value of the control's container changes.</td>
</tr>
<tr>
<td><strong>OnQueryContinueDrag</strong> (inherited from Control)</td>
<td>Raises the QueryContinueDrag event.</td>
</tr>
<tr>
<td><strong>OnResize</strong> (inherited from Form)</td>
<td></td>
</tr>
<tr>
<td><strong>OnRightToLeftChanged</strong> (inherited from Control)</td>
<td>Raises the RightToLeftChanged event.</td>
</tr>
<tr>
<td><strong>OnSizeChanged</strong> (inherited from Control)</td>
<td>Raises the SizeChanged event.</td>
</tr>
<tr>
<td><strong>OnStyleChanged</strong> (inherited from Control)</td>
<td></td>
</tr>
<tr>
<td>Method Name</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td><strong>OnSystemColorsChanged</strong></td>
<td>(inherited from <strong>Control</strong>) Raises the <strong>SystemColorsChanged</strong> event.</td>
</tr>
<tr>
<td><strong>OnTabIndexChanged</strong></td>
<td>(inherited from <strong>Control</strong>) Raises the <strong>TabIndexChanged</strong> event.</td>
</tr>
<tr>
<td><strong>OnTabStopChanged</strong></td>
<td>(inherited from <strong>Control</strong>) Raises the <strong>TabStopChanged</strong> event.</td>
</tr>
<tr>
<td><strong>OnTextChanged</strong></td>
<td>(inherited from <strong>Form</strong>) Raises the <strong>TextChanged</strong> event.</td>
</tr>
<tr>
<td><strong>OnValidated</strong></td>
<td>(inherited from <strong>Control</strong>) Raises the <strong>Validated</strong> event.</td>
</tr>
<tr>
<td><strong>OnValidating</strong></td>
<td>(inherited from <strong>Control</strong>) Raises the <strong>Validating</strong> event.</td>
</tr>
<tr>
<td><strong>OnVisibleChanged</strong></td>
<td>(inherited from <strong>Form</strong>) Raises the <strong>VisibleChanged</strong> event.</td>
</tr>
<tr>
<td><strong>ProcessCmdKey</strong></td>
<td>(inherited from <strong>Form</strong>) Processes a key message and generates the appropriate control events.</td>
</tr>
<tr>
<td><strong>ProcessDialogChar</strong></td>
<td>(inherited from <strong>ContainerControl</strong>)</td>
</tr>
<tr>
<td><strong>ProcessDialogKey</strong></td>
<td>(inherited from <strong>Form</strong>)</td>
</tr>
<tr>
<td><strong>ProcessKeyEventArgs</strong></td>
<td>(inherited from <strong>Control</strong>) Raises the <strong>ProcessKeyEventArgs</strong></td>
</tr>
<tr>
<td><strong>ProcessKeyPreview</strong></td>
<td>(inherited from <strong>Form</strong>)</td>
</tr>
<tr>
<td><strong>ProcessMnemonic</strong></td>
<td>(inherited from <strong>ContainerControl</strong>)</td>
</tr>
<tr>
<td><strong>ProcessTabKey</strong></td>
<td>(inherited from <strong>Form</strong>)</td>
</tr>
<tr>
<td><strong>RaiseDragEvent</strong></td>
<td>(inherited from <strong>Form</strong>)</td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>--------</td>
<td>-------------</td>
</tr>
<tr>
<td><strong>RaiseKeyEvent</strong> (inherited from Control)</td>
<td></td>
</tr>
<tr>
<td><strong>RaiseMouseEvent</strong> (inherited from Control)</td>
<td></td>
</tr>
<tr>
<td><strong>RaisePaintEvent</strong> (inherited from Control)</td>
<td></td>
</tr>
<tr>
<td><strong>RecreateHandle</strong> (inherited from Control)</td>
<td>Forces the re-creation of the handle for the control.</td>
</tr>
<tr>
<td><strong>ResetMouseEventArgs</strong> (inherited from Control)</td>
<td></td>
</tr>
<tr>
<td><strong>RtlTranslateAlignment</strong> (inherited from Control)</td>
<td>Overloaded. Converts the specified ( \text{HorizontalAlignment} ) to the appropriate ( \text{HorizontalAlignment} ) to support right-to-left text.</td>
</tr>
<tr>
<td><strong>RtlTranslateContentAlignment</strong> (inherited from Control)</td>
<td>Converts the specified ( \text{ContentAlignment} ) to the appropriate ( \text{ContentAlignment} ) to support right-to-left text.</td>
</tr>
<tr>
<td><strong>RtlTranslateHorizontal</strong> (inherited from Control)</td>
<td>Converts the specified ( \text{HorizontalAlignment} ) to the appropriate ( \text{HorizontalAlignment} ) to support right-to-left text.</td>
</tr>
<tr>
<td><strong>RtlTranslateLeftRight</strong> (inherited from Control)</td>
<td>Converts the specified ( \text{LeftRightAlignment} ) to the appropriate ( \text{LeftRightAlignment} ) to support right-to-left text.</td>
</tr>
<tr>
<td><strong>ScaleCore</strong> (inherited from Form)</td>
<td></td>
</tr>
<tr>
<td><strong>Select</strong> (inherited from Form)</td>
<td>Overloaded.</td>
</tr>
<tr>
<td><strong>SetBoundsCore</strong> (inherited from Form)</td>
<td></td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>--------</td>
<td>-------------</td>
</tr>
<tr>
<td><strong>SetClientSizeCore</strong> (inherited from Form)</td>
<td></td>
</tr>
<tr>
<td><strong>SetDisplayRectLocation</strong> (inherited from ScrollableControl)</td>
<td></td>
</tr>
<tr>
<td><strong>SetScrollState</strong> (inherited from ScrollableControl)</td>
<td></td>
</tr>
<tr>
<td><strong>SetStyle</strong> (inherited from Control)</td>
<td>Sets the specified style bit to the specified value.</td>
</tr>
<tr>
<td><strong>SetTopLevel</strong> (inherited from Control)</td>
<td>Sets the control as the top-level control.</td>
</tr>
<tr>
<td><strong>SetVisibleCore</strong> (inherited from Form)</td>
<td></td>
</tr>
<tr>
<td><strong>UpdateBounds</strong> (inherited from Control)</td>
<td>Overloaded. Updates the bounds of the control with the current size and location.</td>
</tr>
<tr>
<td><strong>UpdateDefaultButton</strong> (inherited from Form)</td>
<td></td>
</tr>
<tr>
<td><strong>UpdateStyles</strong> (inherited from Control)</td>
<td>Forces the assigned styles to be reapplied to the control.</td>
</tr>
<tr>
<td><strong>UpdateZOrder</strong> (inherited from Control)</td>
<td>Updates the control in its parent's z-order.</td>
</tr>
<tr>
<td><strong>WndProc</strong> (inherited from Form)</td>
<td></td>
</tr>
</tbody>
</table>

**Protected Internal Instance Methods**

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>ProcessKeyMessage</strong> (inherited from Control)</td>
<td>Processes a keyboard message.</td>
</tr>
</tbody>
</table>

**Explicit Interface Implementations**

<table>
<thead>
<tr>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>IContainerControl.ActivateControl</strong> (inherited from ContainerControl)</td>
<td></td>
</tr>
</tbody>
</table>
See Also

SuggestionForm Class | NetSpell.SpellChecker.Forms Namespace

Copyright (C) 2003 Paul Welter
**SuggestionForm.Dispose Method**

Clean up any resources being used.

**Overload List**

Inherited from [Component](#).

```csharp
public void Dispose();
```

Clean up any resources being used.

```csharp
protected override void Dispose(bool);
```

**See Also**

[SuggestionForm Class](#) | [NetSpell.SpellChecker.Forms Namespace](#)
SuggestionForm.Dispose Method (Boolean)

Clean up any resources being used.

```csharp
protected override void Dispose(bool disposing);
```

See Also

SuggestionForm Class | NetSpell.SpellChecker.Forms Namespace | SuggestionForm.Dispose Overload List

Copyright (C) 2003 Paul Welter