SpellChecker for WPF/Silverlight

2018.02.20 更新

グレープシティ株式会社

目次

製品の概要

SpellChecker for WPF/Silverlight は、XML ドキュメントの生成に使用されるフォーム、Web ページ、コメント内のスペルミスを検出するために Visual Studio のアドインとしてアプリケーションのスペルチェックを実行する **IntelliSpell** と同じエンジンを使用します。

> **メモ**: 説明内に含まれるクラスおよびメンバーに対するリファレンスへのリンクは、原則としてWPF版のリファレンスページを参照します。Silverlight版については、目次から同名のメンバーを参照してください。

主な特長

SpellChecker for WPF/Sivlerlight を使用すると、機能豊富でカスタマイズされたアプリケーション を作成できます。次の主要な機能を利用して、**SpellChecker** for WPF/Silverlight を最大限に活用してください。

- **最も効率のよいスペルチェッカー**
  標準的なシステムにおいて、**SpellChecker** は1秒あたり 400,000 ワードをスペルチェックできます。スペル辞書は圧縮でき、保守も簡単です。

- **使用が簡単**
  コントロールのスペルチェックに必要なことは、**CheckControl** メソッドの呼び出しだけです。

- **柔軟なスペルチェック**
  SpellChecker はさまざまなスペルチェックサービスを提供します。

  - 文字列やコントロールに対してスペルチェックを実行し、スペルミスの単語に対しては正しいスペル候補を提示します。

  - インタフェースベースのアーキテクチャを使用しているため、テキストを含む任意のコントロールに対してスペルチェックを実行したり、カスタムのスペルダイアログ、辞書、テキストパーサーを実装することができます。

  - **SpellChecker** の**SpellOptions** プロパティを使用して、スペルチェックのオプションを微調整できます。

- **使いやすいクライアント側スペルチェッカー**
  SpellChecker は、Web ページでスペルチェックを実装するためのクライアント側ソリューションを提供します。最小限の作業でスペルチェックを実装して配布できます。コントロールのスペルチェックに必要なことは、**CheckControl** メソッドの呼び出しだけです。

- **多言語サポート**
  SpellChecker には、英語、スペイン語、フランス語、イタリア語、ドイツ語、ポルトガル語、オランダ語、ロシア語、デンマーク語、スウェーデン語、ギリシャ語などの 20 以上の言語のスペル辞書が付属します。各言語のスペル辞書は、オンラインデモ からダウンロードできます。

SpellChecker for WPF の使い方

SpellChecker for WPF/Sliverlight は、プレーンテキストやコントロールのスペルチェックに使用できるスペルチェックエンジンです。

概念と主要なプロパティ

以下のトピックでは、**Spellchecker for WPF/Sliverlight** の概念と主要なプロパティについて説明します。

スペル辞書

SpellChecker for WPF/Silverlight は、3種類の辞書を使用してテキストをチェックします。それぞれの辞書は、**MainDictionary**、**UserDictionary**、**CustomDictionary** の3つのプロパティの1つとして公開されます。以下のトピックでは、この3つのプロパティについて1つずつ詳しく説明します。

MainDictionary

メイン辞書は、読み取り専用の圧縮された単語リストで、一般に数十万の単語が格納されます。通常、メイン辞書は、次の例のように、アプリケーションの起動時に **LoadAsync** メソッドを使用してロードされます。

C#

```
public Page()
  {
    InitializeComponent();
    // 他の初期化...
    // メイン辞書をロードします
    SpellDictionary md = c1SpellChecker1.MainDictionary;
    md.LoadCompleted += md_LoadCompleted;
    md.LoadProgressChanged += md_LoadProgressChanged;
    md.LoadAsync("C1Spell_en-US.dct");
  }
```

LoadAsync メソッドは、1つのパラメータとしてスペル辞書の URL を受け取ります。通常、スペル辞書はアプリケーションの一部として配布されます。上の例は、**SpellChecker** に付属するアメリカ英語のスペル辞書 **C1Spell_en-US.dct** をロードします。この辞書には、約 120,000 の単語が収録され、サイズは 250 キロバイトです。

LoadAsync メソッドは、メイン辞書を非同期にロードするため、辞書をサーバーからダウンロードしている間にアプリケーションを開始できます。ダウンロードが完了すると、C1SpellChecker コンポーネントは、**C1SpellChecker** コンポーネントが利用可能かどうかを判断するために使用される **LoadCompleted** メソッドを起動します。

MainDictionary.State プロパティを使用すると、メイン辞書のロードが成功したかどうかをいつでも確認できます。

Stream オブジェクトからメイン辞書をロードし、必要に応じてロードプロセスをカスタマイズすることもできます。

UserDictionary

ユーザー辞書は、読み取り/書き込み可能な単語リストで、名前や技術用語など(「ComponentOne」や「Silverlight」など)のユーザー定義の用語を格納します。ユーザーは、スペルチェック中にスペルダイアログボックスの[追加]ボタンをクリックして、カスタム辞書に単語を追加できます。コードを使用して単語を追加および削除することもできます。

通常、ユーザー辞書は、サーバーではなく、アプリケーション内の分離ストレージに保存されます。次のよう

に、**LoadFromIsolatedStorage** メソッドと **SaveToIsolatedStorage** メソッドを使用して、カスタム辞書をロードおよび保存できます。

C#

```
public Page()
 {
   InitializeComponent();
   // 他の初期化...
   // メイン辞書をロードします
   SpellDictionary md = c1SpellChecker1.MainDictionary;
   md.LoadCompleted += md_LoadCompleted;
   md.LoadProgressChanged += md_LoadProgressChanged;
   md.LoadAsync("C1Spell_en-US.dct");
   // ユーザー辞書をロードします
   UserDictionary ud = c1SpellChecker1.UserDictionary;
   ud.LoadFromIsolatedStorage("Custom.dct");
   // アプリケーションの終了時にユーザー辞書を保存します
   App.Current.Exit += App_Exit;
 }
 void App_Exit(object sender, EventArgs e)
 {
   // 変更されたユーザー辞書を圧縮された分離ストレージに保存します
   UserDictionary ud = c1SpellChecker1.UserDictionary;
   ud.SaveToIsolatedStorage("Custom.dct");
 }
```

このコードは、ユーザー辞書を分離ストレージからロードし、アプリケーションの終了時にすべての変更を保存するためのイベントハンドラをアタッチします。通常、ユーザー辞書は小さく、圧縮された形式で保存されるため、ストレージ領域をそれほど使用しません。

ユーザー辞書を **Stream** オブジェクトにロードしたり保存して、必要に応じてユーザー辞書の永続化メカニズムをカスタマイズすることもできます。

CustomDictionary

カスタム辞書は C1Window の派生クラスで、**ISpellDictionary** インタフェースを実装します。これを使用して、アプリケーションに合わせたカスタムロジックに基づいて、辞書を作成できます。たとえば、Web 上の単語を探してキャッシュに保存する辞書や、専用のデータベースから単語を探す辞書を作成できます。

カスタム辞書はオプションです。

スペルチェックのモード

SpellChecker for WPF/Sliverlight は、バッチモード、モーダルスペルチェックモード、入力時スペルチェックモードの3つのスペルチェックモードをサポートします。以下のセクションでは、スペルチェックの各モードについて説明します

バッチモード

CheckText、**CheckWord**、**GetSuggestions** の各メソッドは、文字列をチェックし、エラーのリストを取得し、正しいスペルの候補を提供します。これらのメソッドを使用すると、コントロール内のテキスト以外のテキストをスペルチェックできます。たとえば、データベースに保存されているテキストをスペルチェックできます。

次のコードはこのモードの例です。

C#

```
    // テキストをチェックします
  var someText = "this text comtains two errrors.";
  var errors = c1SpellChecker1.CheckText(someText);
  Debug.WriteLine("CheckText(\"{0}\") =", someText);
  foreach (var error in errors)
  {
    Debug.WriteLine("\t{0}, {1}-{2}",
      error.Text, error.Start, error.Length);
    foreach (string suggestion in
       c1SpellChecker1.GetSuggestions(error.Text, 1000))
    {
      Debug.WriteLine("\t\t{0}?", suggestion);
    }
  }
```

モーダルスペルチェック

CheckControlAsync メソッドは、**ISpellCheckableEditor** インタフェースを実装しているコントロールのスペルチェックを実行します。Microsoft **TextBox** コントロールと **C1RichTextBox** コントロールは、このインタフェースを組み込みで実装しています。他のコントロールには、独自のクラスを作成してスペルチェック可能なラッパーを提供することができます。

次のコードはこのモードの例です。

C#

```
    // モーダルスペルチェックを表示します
  private void Button2_Click(object sender, RoutedEventArgs e)
  {
    // イベントハンドラを登録します
    c1SpellChecker1.CheckControlCompleted +=
      c1SpellChecker1_CheckControlCompleted;
    // テキストボックスをスペルチェックします
    c1SpellChecker1.CheckControlAsync(textBox1);
    // イベントハンドラの登録を解除します
    c1SpellChecker1.CheckControlCompleted +=
      c1SpellChecker1_CheckControlCompleted;
  }
  void c1SpellChecker1_CheckControlCompleted(object sender,
       CheckControlCompletedEventArgs e)
  {
    Debug.WriteLine("CheckControlCompleted: {0} errors found",
      e.ErrorCount);
    if (e.Cancelled)
      WriteLine("\t(cancelled...)");
  }
```

このコードは、スペルチェックダイアログボックスを表示し、スペルミスをそれぞれ強調表示し、正しいスペル候補を表示して、ユーザーがエラーを1つずつ修正したり、無視できるようにします。プロセスが完了したら、**CheckControlCompleted** イベントが起動し、ユーザーへのフィードバックを提供します。

入力時スペルチェック

このモードは、ユーザーの入力中にコントロールを監視し、Microsoft Word のスペルチェッカーと同様に、エラーに赤い波線で下線を引きます。スペルミスの単語を右クリックすると、正しいスペルの候補を含むコンテキストメニューが表示されます。

このモードは、**SpellChecker for WPF/Silverlight** にはまだ実装されていません。

スペルチェックのオプション

Options プロパティを使用すると、スペルチェックプロセスを微調整できます。このプロパティは、次のプロパティを含む **SpellOptions** オブジェクトを返します。

- **DialogLanguage**：組み込みのスペルダイアログボックスで使用される言語をカスタマイズします。
- **Ignore**：大文字、大文字小文字の混在した単語、数字、HTML タグ、URL などを含む単語を無視するかどうかを指定します。
- **ActiveSpellingEnabled**：入力時スペルチェックが有効にされているかどうかを取得または設定します。現時点では、このプロパティは **SpellChecker for WPF/Silverlight** でサポートされていません。
- **ShowSuggestionsInContextMenu**：入力中スペルチェックが有効な場合に、C1SpellChecker コンポーネントがスペルミスの単語のコンテキストメニューを提供するかどうかを取得または設定します。現時点では、このプロパティは **SpellChecker for WPF/Silverlight** でサポートされていません。
- **MaxSuggestionsInContextMenu**：スペルミスの単語に関連付けられたコンテキストメニューに表示する正しいスペル候補の最大数を取得または設定します。現時点では、このプロパティは **SpellChecker for WPF/Silverlight** でサポートされていません。
- **UnderlineColor**：入力時モードでスペルミスの単語に引く下線の色を取得または設定します。現時点では、このプロパティは **SpellChecker for WPF/Silverlight** でサポートされていません。

スペルチェックダイアログボックス

CheckControlAsync メソッドを呼び出すと、**SpellChecker for WPF/Sliverlight** は、コントロール内で見つかったエラーをそれぞれ強調表示し、ユーザーがエラーを修正または無視するためのダイアログボックスを表示します。

SpellChecker for WPF/Silverlight には、デフォルトで使用されるスペルダイアログボックスが組み込まれています。組み込みのダイアログボックスをそのまま使用することも、それをカスタマイズすることも、独自のダイアログボックスに置き換えることもできます。

組み込みのスペルダイアログボックスの使用

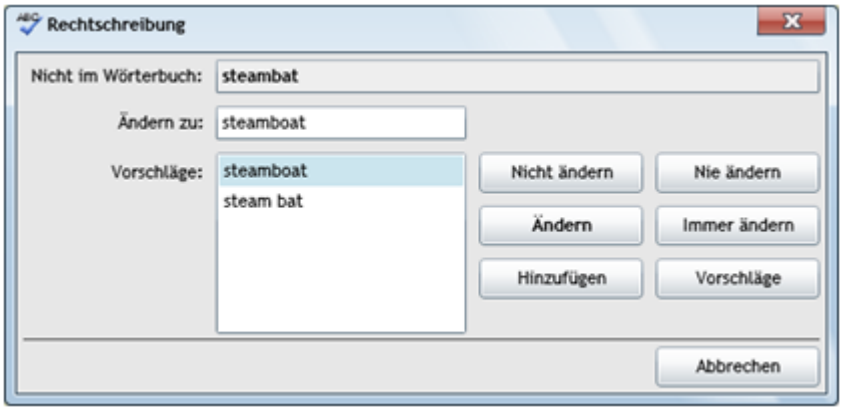
組み込みのスペルダイアログボックスは簡単に使用できます。必要なことは、**CheckControlAsync** メソッドを呼び出すだけです。

組み込みのダイアログボックスには、7つの言語をサポートするローカライズ機能が組み込まれています。ローカライズは、**SpellOptions.DialogLanguage** プロパティによって制御されます。デフォルト設定の **Automatic** は、ダイアログボックスを **CultureInfo.CurrentCulture** プロパティに対応する言語で表示します。これは次のように上書きできます。

C#

```
SpellOptions options = c1SpellChecker1.Options;
options.DialogLanguage = DialogLanguage.German;
```

これはスペルダイアログボックスのドイツ語バージョンです。



ダイアログボックスの言語は、アプリケーションの言語に一致し、現在の辞書の言語とは独立していることに注意してください。たとえば、英語のアプリケーションを記述し、これを使用してドイツ語のテキストをチェックすることができます。

組み込みのスペルダイアログボックスのカスタマイズ

組み込みのダイアログボックスは、C1Window から派生します。C1Window をインスタンス化し、プロパティを変更し、カスタマイズされたダイアログボックスを **CheckControlAsync** メソッドに渡すと、小規模なカスタマイズを簡単に実行できます。

たとえば、次のコードは、組み込みのダイアログボックスのキャプションを修正します。

C#

```
var dlg = new C1SpellDialog();
dlg.Header = "カスタマイズされたキャプション";
c1SpellChecker1.CheckControlAsync(textBox1, false, dlg);
```

これはダイアログボックスのカスタマイズバージョンです。

カスタムスペルダイアログボックスの使用

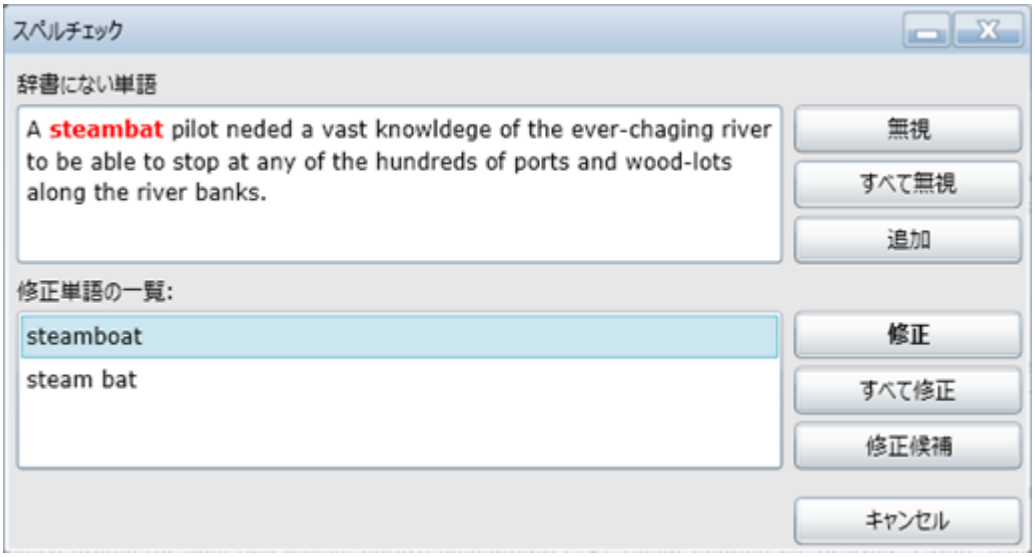
カスタムスペルダイアログボックスを使用するには、次の2つのタスクを実行する必要があります。

1. C1Window から派生するクラスを作成し、**ISpellDialog** インタフェースを実装します。
2. **ISpellDialog** パラメータを受け取るオーバーロードバージョンの**CheckControlAsync** メソッドを使用します。

最初のタスクを簡単に行えるように、独自のダイアログボックスを作成するためのベースに使用できる2つのダイアログボック

スが用意されています。このうちの1つは組み込みのダイアログと同じで、もう1つは Microsoft Word のスペルダイアログボックスに似ています。ソースコードは、"Samples" フォルダに製品と一緒にインストールされている "SpellCheckerSample" アプリケーションにあります。

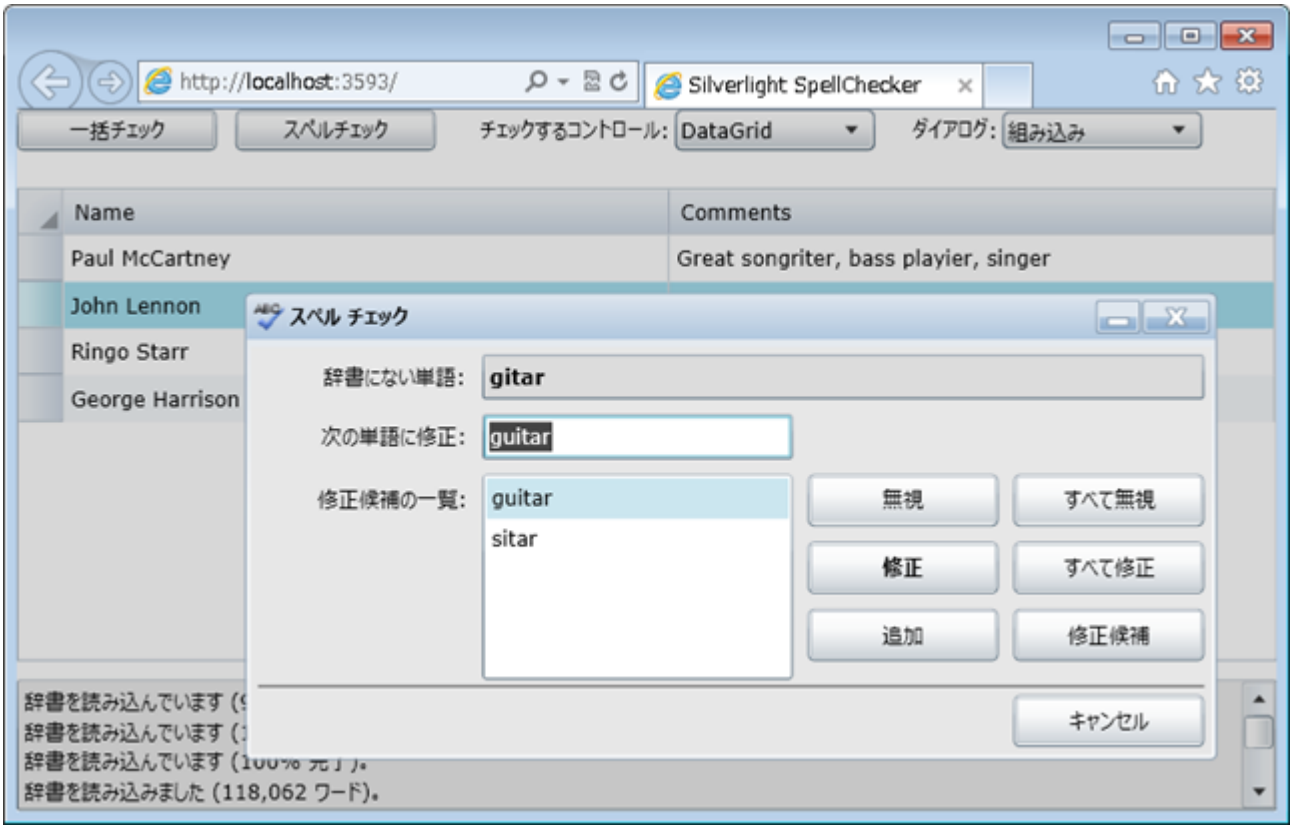
2番目のカスタムダイアログボックスは、次のように表示されます。



さまざまな種類のコントロールのスペルチェック

SpellChecker for WPF/Silverlight を使用すると、テキストを含む任意のコントロールのスペルチェックを実行できます。**C1RichTextBox** と Microsoft **TextBox** コントロールは、スペルチェックが組み込みでサポートされています。他のコントロールでスペルチェックを実行するには、**ISpellCheckableEditor** インタフェースを実装するラッパークラスを用意する必要があります。

たとえば、Silverlightの[SpellCheckerSample]は DataGrid コントロールのスペルチェックを実行する方法を示しています。グリッドに代わって ISpellCheckableEditor インタフェースを実装する **DataGridSpellWrapper** というクラスを作成します。このラッパークラスは、グリッドセルを列挙し、**C1SpellChecker** コンポーネントに各セルのテキストを公開します。エラーが見つかると、セルが選択され、通常どおりスペルダイアログボックスにエラーと正しいスペルの候補が表示されます。次の画像は、SpellCheckerSample サンプルで **DataGrid** をスペルチェックしているところです。



スペル辞書の形式

メイン辞書は、.dct 拡張子を持つ zip ファイルです。この zip ファイルには複数の単語リストを格納できます。各リストは、各行に有効な単語が1つずつ含まれる UTF-8 エンコードのテキストファイルです。これらのリストの拡張子はすべて ".words" である必要があります。

小文字で始まる単語(「apple」、「banana」、「cherry」など)は、通常の単語であると見なされ、小文字でも大文字でも使用できます。大文字で始まる単語(「Albert」、「Bernoulli」、「Cantor」など)は、固有名詞であると見なされ、小文字で使用されるとスペルミスとしてマークされます。

.dct ファイルには1つの "rules" エントリも格納できます。これは、辞書の言語でテキストをスペルチェックする条件を適用するルールを指定します。たとえば、**SpellChecker for WPF/Silverlight** に付属するフランス語辞書には、次のルールが含まれています。

- **IgnorePrefix**:l' d' j' da' m' s' n' qu'
- **IgnoreSuffix**:'s

これらは、スペルチェッカーに一般的なプリフィックスとサフィックスを無視するように指示します。これらは単語をチェックする前に削除されます。次に例を示します。

- l'amour(「amour」をチェック:正しい)
- l'amuor(「amuor」をチェック:**正しくない**)
- Maxim's(「Maxim」をチェック:正しい)
- Naxim's(「Naxim」をチェック:**正しくない**)

  条件に含まれていないプリフィックスやサフィックスは、スペルミスとしてマークされます。
- h'amour(「h'amour」をチェック:**正しくない**)

- x'amuor(「x'amuor」をチェック: **正しくない**)

辞書ファイルを作成および編集するには、メモ帳や WinZip などの一般的なアプリケーションを使用したり、**SpellChecker for WPF/Silverlight** に付属する C1DictionaryEditor アプリケーションを使用することができます。