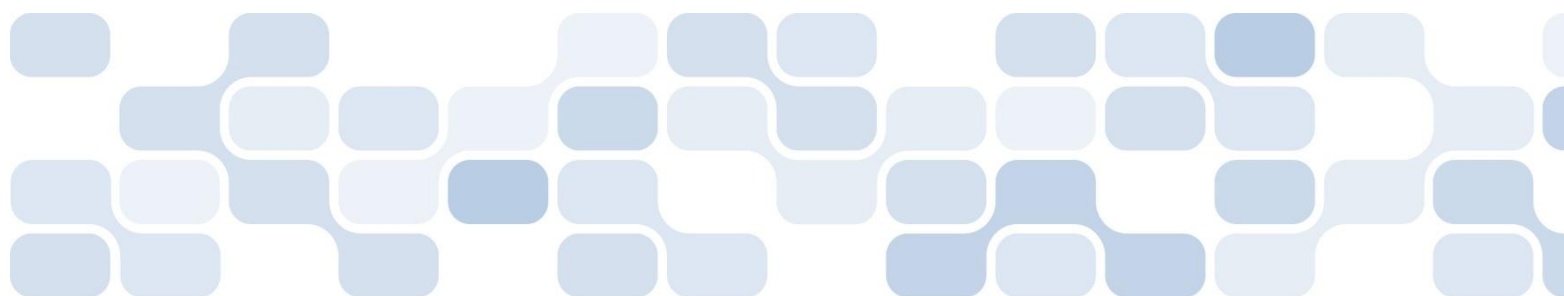




Visual Studio Do-It-Yourself シリーズ

第 1 回 Windows アプリケーション開発の概要

Microsoft®



■はじめに – Visual Studio Do-It-Yourself について

本 Visual Studio Do-It-Yourself シリーズでは、9 回にわたって Visual Studio による Windows アプリケーション開発の基礎を学習していきます。本書の目的は、初めて Visual Studio を学ぶプログラマーが、スタンドアロンで実行可能な Windows アプリケーションを開発する際のファーストステップを提供することです。コンパクトな実例を紹介し、Visual Studio の統合開発環境を体験しながら、Windows アプリケーション開発の理解を深めてください。

第 1 回は、アプリケーションを開発する前準備として、Visual Studio の概要を紹介するとともに、統合開発環境を使った Windows アプリケーション開発の手順についてご紹介します。はじめて Visual Studio を使われる方を対象にしていますので、これまで Visual Studio をお使いになったことのある方は、次回以降にスキップできます。次回以降は、Visual Studio の統合開発環境を使った Windows アプリケーションを開発について、ユーザー インターフェイスの設計方法や、プログラム コードの記述方法を具体的に学習していきます。

●実行環境について

本書では、Windows アプリケーションを開発する環境として、Visual Studio 2010 Express Edition 以上を前提にします。無償で利用できる Visual Studio 2010 Express Edition はこちら (<http://www.microsoft.com/japan/msdn/vstudio/express/>) からダウンロードできます。Visual Studio 2010 にはさまざまなエディションがあり、それぞれのエディションで使える機能やメニュー階層に差異があるため、本書で解説する手順とは異なる場合があります。

Visual Studio 2010 では、Visual Studio 2005 以前の Windows フォームを使うアプリケーションの開発もできますが、ここでは WPF (Windows Presentation Foundation) という新しい技術を使います。WPF を使う場合でも、Windows フォームと同じく、コントロールをウィンドウにドラッグ アンド ドロップして配置したり、イベント ハンドラーを記述することで Windows アプリケーションを作成します。また、XAML (eXtensible Application Markup Language) と呼ばれる記述方法により、WPF をさらに活用できることがあります。

●本書がカバーする内容

本書では、Visual Studio でアプリケーション開発するためのビジュアル操作と簡単なコーディングを扱います。プログラミング言語には C# を使いますが、C# 言語の詳細については触れません。あらかじめご了承ください。

■WPF とは

WPF (Windows Presentation Foundation) とは、ユーザー インターフェイスを作成するための新しい技術で、高速かつなめらかな描画、柔軟性の高いレイアウト、スタイルとロジックの分離といった、Windows フォームにはない新しい特徴を備えています。WPF により、いままでにはないユーザー エクスペリエンスを実現します。

以下の画面は、WPF アプリケーションの実行例です。



はじめてプログラミングをされる方はもちろん、これまで Windows フォームでアプリケーションを開発されていた方は、よりスムーズに WPF アプリケーションを開発していただけるでしょう。今回ご説明するように、ウィンドウにコントロール (部品) を配置し、プロパティやイベント ハンドラーを割り当てていくというビジュアル開発のスタイルは、今までと変わりません。

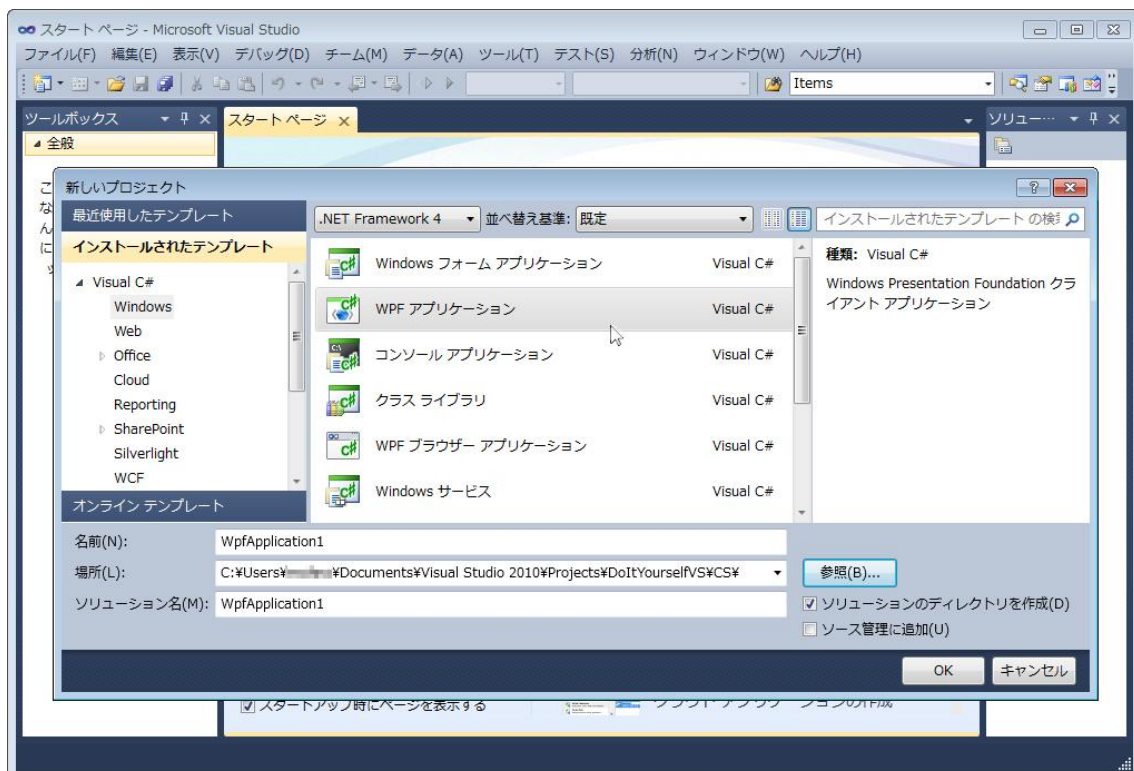
■ Visual Studio による Windows アプリケーションの開発

Visual Studio は、ボタンや入力ボックスなどのユーザー インターフェイスを作成するための「部品」をウィンドウに配置し、C#や Visual Basic などのプログラミング言語を使って動作（ロジック）を記述することで、アプリケーションを開発するためのツールです。

アプリケーションを作成するためには、まず「プロジェクト」を作成します。プロジェクトとは、ユーザー インターフェイスやロジック、その他のアプリケーションを作成するためのあらゆる情報をひとまとまりに管理できる単位です。あるソフトウェアのために複数のアプリケーション（実行可能ファイル）が必要な場合もあります。このため、Visual Studio では複数のプロジェクトをまとめて「ソリューション」として管理できます。

●プロジェクトの作成

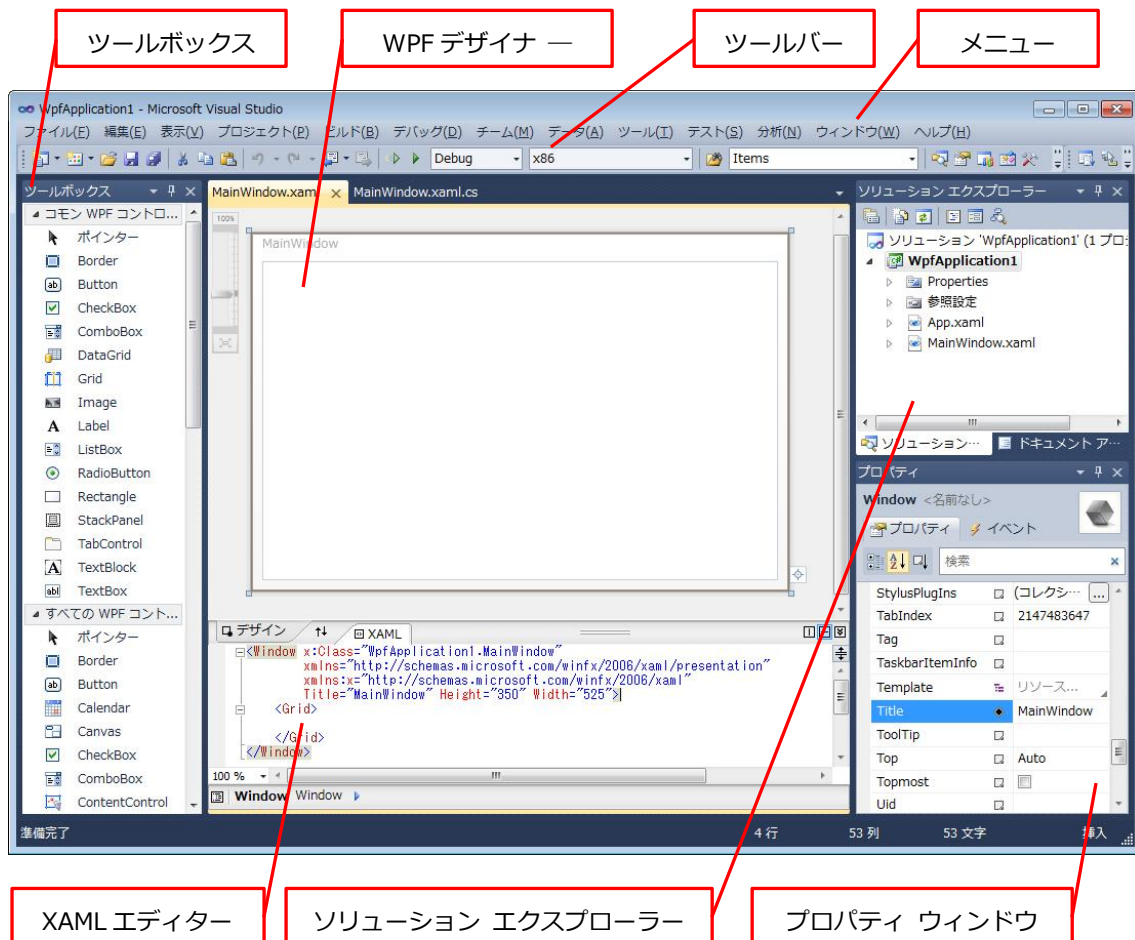
Windowsのスタートメニューを使って Visual Studio を起動してください。[ファイル]–[新規作成]–[プロジェクト]（Express Edition では [ファイル]–[新しいプロジェクト]）を選べると、次の図のようなウィンドウが表示されます。



「インストールされたテンプレート」で使いたい言語カテゴリー（ここでは Visual C#）から「Windows」を選び、「WPF アプリケーション」を選んで、[OK] ボタンをクリックします。この作業で、メイン ウィンドウをひとつ持つアプリケーションのためのプロジェクト（およびソリューション）が作成されます。

●統合開発環境

プロジェクトを作成した直後の Visual Studio の様子と、各部分の名称を示します。ここに表示されていないウィンドウ、あるいは以下の図に表示されていないのに手元で表示されていないウィンドウがあるときは、[表示] メニューを使って表示できます。



ウィンドウの右上に×印がついているものは、ここをクリックすることで閉じることができます。閉じたウィンドウは、[表示] メニューを使って再び表示できます。×印の左隣にある虫ピンのようなアイコンをクリックすると、ウィンドウを使っていないときに自動的に隠す（タイトルバーだけの状態にすること）ことができます。この虫ピンが縦になっているときは「ピンで止めたように」その位置に固定され、虫ピンが横になっているときは「ピンが外れている」ことを意味しています。

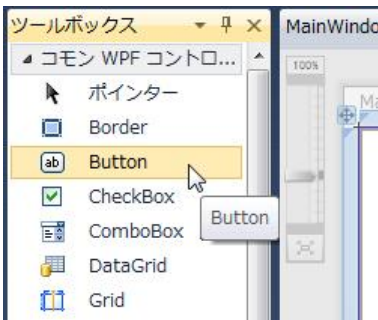
また、ウィンドウのタイトルバーをマウスでドラッグすることで、ウィンドウの配置を変更できます。上下左右にドッキングしたり、タブ形式で同じ場所を複数のウィンドウで切り替えたりできます。ウィンドウのレイアウトを初期設定に戻したいときは、[ウィンドウ]メニューで [ウィンドウ レイアウトのリセット] を選んでください。

とくに、Express Edition では表示されるウィンドウや配置が上記と異なっていますので、上記の説明を参考にして、今後の解説で必要なウィンドウを表示、配置してください。

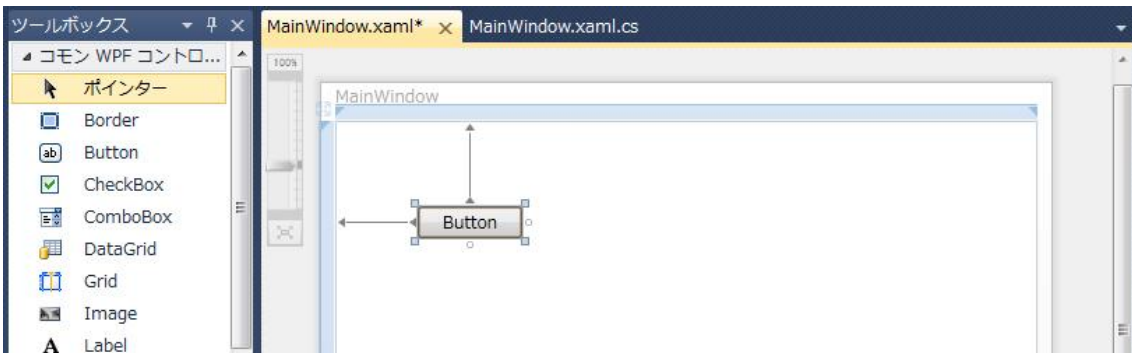
●コントロールを配置する

コントロールとは、ユーザー インターフェイスを作成するための部品のことです。コントロールは、左側にある「ツールボックス」に用途別に分類されています。ここから使いたい部品を選んで、中央のウィンドウに配置します。

まず、ツールボックスから「Button」コントロールを探してクリックしてください。すると、選んだコントロールが次の図のようにハイライトされます。



次にウィンドウの適当な位置をクリックしてください。すると、クリックした場所に Button コントロールが配置されます。



コントロールを配置すると、ツールボックスのハイライト表示が最上部の「ポインター」に戻ります。この状態のときにウィンドウ上でクリックすることは、コントロールを選ぶことを意味します。選んでいるコントロールの四隅または四辺をドラッグすれば、配置したコントロールの大きさを変更できます。コントロール自身をドラッグすれば、配置場所を移動できます。

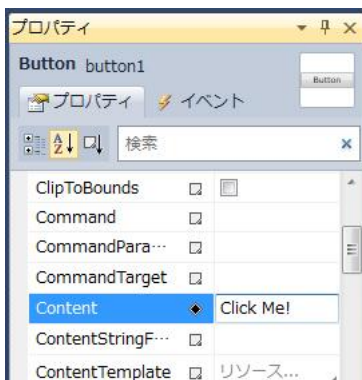
コントロールを配置するときウィンドウ上をドラッグすれば、その時にコントロールの大きさを設定できます。また、ツールボックスからウィンドウ上にコントロールをドラッグ アンド ドロップすることもできます。ツールボックスで、コントロールをダブルクリックすると、ウィンドウの中央にそのコントロールが配置されます。

ここでは、さらに「TextBox」コントロールを選んでウィンドウに配置しておいてください。

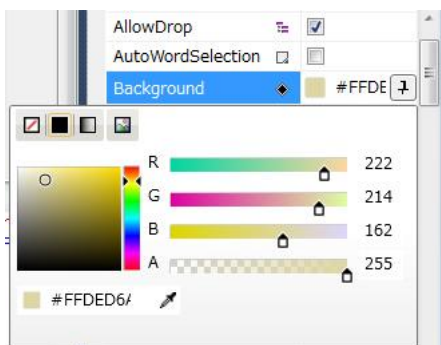
●プロパティを設定する

右下に「プロパティ」というタイトルのウィンドウがあります。コントロールを選ぶと、そのコントロールのプロパティの一覧が表示されます。プロパティは、コントロールをどんな色で表示するか、コントロールに表示する文字列を何にするかといったことを決める属性のことです。

ウィンドウ上に配置した Button コントロールを選んでプロパティ ウィンドウを見ると、自動的に button1 という名前が付けられていることがわかります。プロパティ ウィンドウの左側の列から「Content」というプロパティを探してください。これはボタン上に表示する内容をあらわすものです。デフォルトでは「Button」という文字列が割り当てられているので、これを「Click Me!」に変更します。するとウィンドウ上のボタンに表示されている文字列も「Click Me!」に変わります。



次にウィンドウ上の TextBox コントロールを選んでください。プロパティ ウィンドウでは textBox1 という名前が付けられています。Background プロパティを探して右側の下向き▼ボタンをクリックすると、色を選択できるポップアップが表示されます。

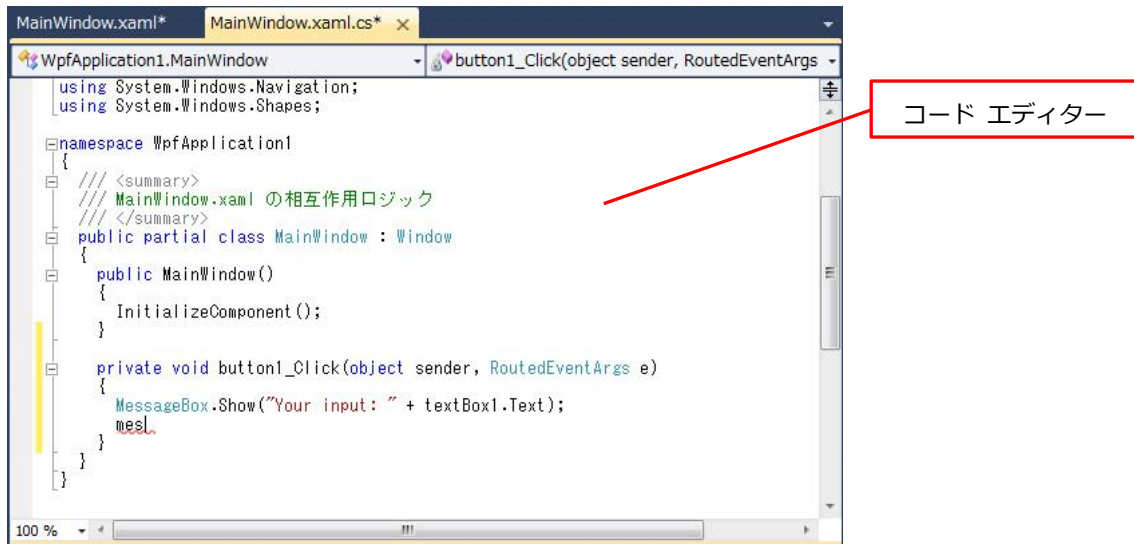


このように、プロパティによってコントロールをどう見せるか、どう使うかを、あらかじめ選べるようになっていきます。

ウィンドウのコントロールの置かれていない空白領域を選ぶと、Grid というレイアウト用コントロールの、MainWindow というタイトルバーをクリックすると、Window 自身を選択でき、プロパティ ウィンドウで、これらのプロパティを変更できます。

●イベント ハンドラーを割り当てる

ウィンドウに置かれている Button コントロールをダブルクリックすると、プログラム コードを編集するコード エディターが開きます。ここに、プログラミング言語（ここではC#）を使って、プログラム コードを書きます。



ここでは、次のようなプログラムを入力します（太字部分）。


```
private void button1_Click(object sender, RoutedEventArgs e)
{
    MessageBox.Show("Your input: " + textBox1.Text);
}
```


textBox1 という名前の入力ボックスに入力した文字列は、コントロールの Text プロパティで取り出せます。このプログラムは、"Your input: "という文字列と、このプロパティの値をつなげて、MessageBox クラスの Show という機能を使って画面に表示することを示しています。

コード エディターには豊富な入力支援機能が備わっています。たとえば、「MessageBox」と入力するために M、e、s、…と入力すると、それに合わせて候補がリストアップされ、そこから選ぶことができます。



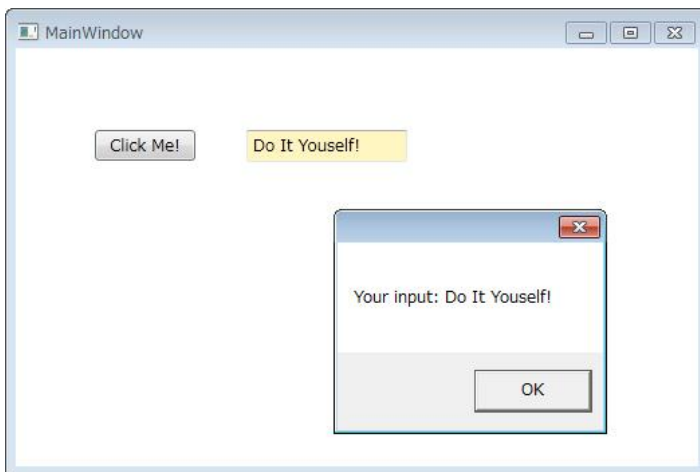
●アプリケーションを実行する

ウィンドウにコントロールを配置してプロパティを設定したり、イベント ハンドラーを割り当てたりすることでアプリケーションを作り上げます。アプリケーションが完成したら、ツールバーの [すべてを保存 ()] ボタンを押して、プロジェクト全体を保存しておきましょう。これは、[ファイル]-[すべてを保存] メニューを選ぶことと同じです。

実際にアプリケーションを実行させるためには、[デバッグ開始 ()] ボタンを押します。これは、[デバッグ]-[デバッグ開始] メニューと同じです。このボタンを押すと、必要に応じて「プロジェクトのビルド」が行われて、アプリケーションが開始します。

もし、「ビルド エラーが発生しました…」というエラー メッセージが表示されたら、これまでの作業を見直して間違いがないか確認し、修正してください。

正しく作業できている場合、このアプリケーションの実行画面は、次のようになります。これは、入力ボックスに「Do It Yourself!」と入力して、[Click Me!] ボタンをクリックした様子です。

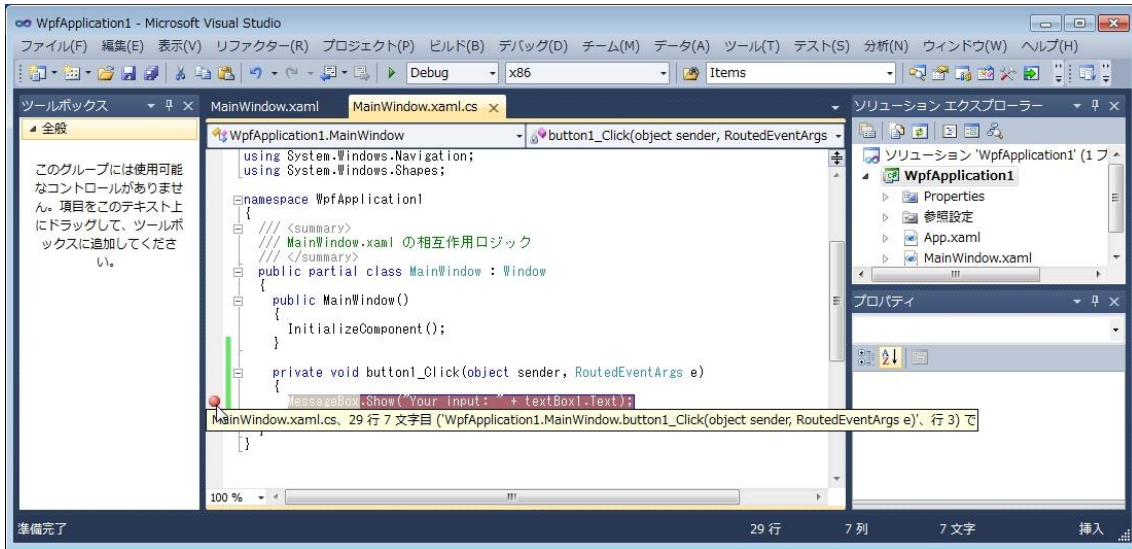


メッセージ ボックスの [OK] を押して閉じ、MainWindow の右上の [×] を押すと、アプリケーションが終了します。

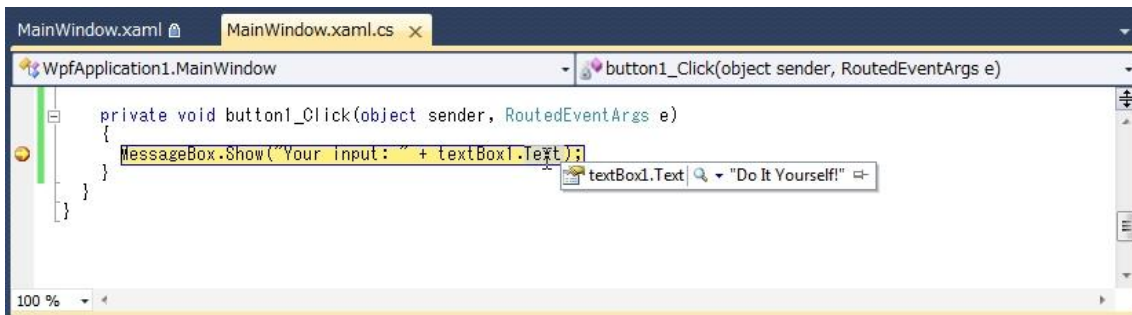
●アプリケーションをデバッグする

プログラムの問題のことをバグ (bug) と言い、バグを取り除くことをデバッグ (debug) と言います。Visual Studio には、効率的なデバッグ作業のための機能が備わっています。

コードエディタで、プログラム コードの左端をクリックすると赤い丸印とともに、そのコード行が赤くハイライト表示されます。これは「ブレークポイント」(停止位置) を設定している様子です。



前述のようにアプリケーションをデバッグ実行すると、このブレークポイントにたどり着いたときにアプリケーションの実行を停止できます。このとき、たとえばマウスカーソルを「textBox1.Text」の上に移動させると、その値をポップアップして表示します。このようにして、プログラムが期待通り動作しているかどうかを確認することができます。



アプリケーションの実行を再開するときは、ふたたびデバッグ実行ボタンをクリックしてください。デバッグ実行中にプログラムを修正することはできませんが、ブレークポイントで停止しているときには修正できます。デバッグ実行を再開するときには、修正した内容が反映されます。

■まとめ

今回は、Visual Studio の基本的な機能を使って、非常に簡単なアプリケーションを作成しました。Visual Studio には数多くの機能が搭載されていますが、最初からそのすべてを覚える必要はありません。実際にアプリケーションを作りながら、容易に機能を学べるようになっています。

次回以降は、より実践的な機能を中心に学習していきます。なお、次回以降では、今回ご紹介したプロジェクトの作成やデバッグについては、とくに手順を説明しません。あらかじめご了承ください。