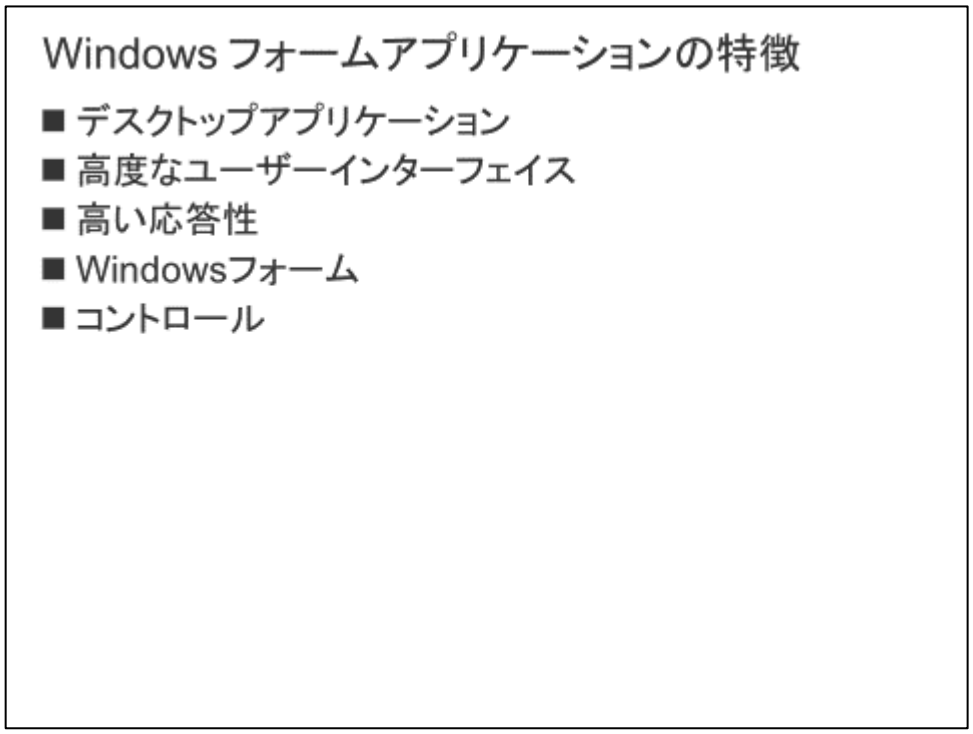
■ Windowsフォームアプリケーション■

■ Windows フォームアプリケーションの特徴



■　デスクトップアプリケーション

Windows アプリケーションは、デスクトップアプリケーションとして動作します。そのため、ユーザーのコンピューターのリソース(CPU、メモリ、etc)を使用して動作します。

■　高度なユーザインタフェース

Windows アプリケーションのユーザインタフェースは、HTML 要素で表現される Web アプリケーションのユーザインタフェースと比較して、高度なインタフェースやわかり易いインタフェースを作成しやすくなっています。

■　高い応答性

Windows アプリケーションは、意図的にネットワークリソースにアクセスしなければ、スタンドアロン環境で動作します。そのため、頻繁に Web サーバーとの通信が発生する Web アプリケーションと比較して、ユーザーの操作にすばやくレスポンスを返すことが出来ます。

■　Windowsフォーム

.NET Framework 上での Windows アプリケーションは、個々のウィンドウを「Windows フォーム」として扱います。Visual Studio を利用すると、Windows フォームのためのコードの大部分を自動生成させることが出来ます。

■　コントロール

Windows フォームのプログラミングでは、ユーザインタフェースの作成に「コントロール」を用います。Windows フォーム用のコントロールには、シンプルなコントロールから、高度な機能を持ったコントロールまで、様々なコントロールが用意されています。

■ Windowsフォームアプリケーションの作成手順■

■ プロジェクトの作成
■ ユーザーインターフェイスの作成
■ プロパティの設定
■ イベントハンドラの実装

■　スタートページ

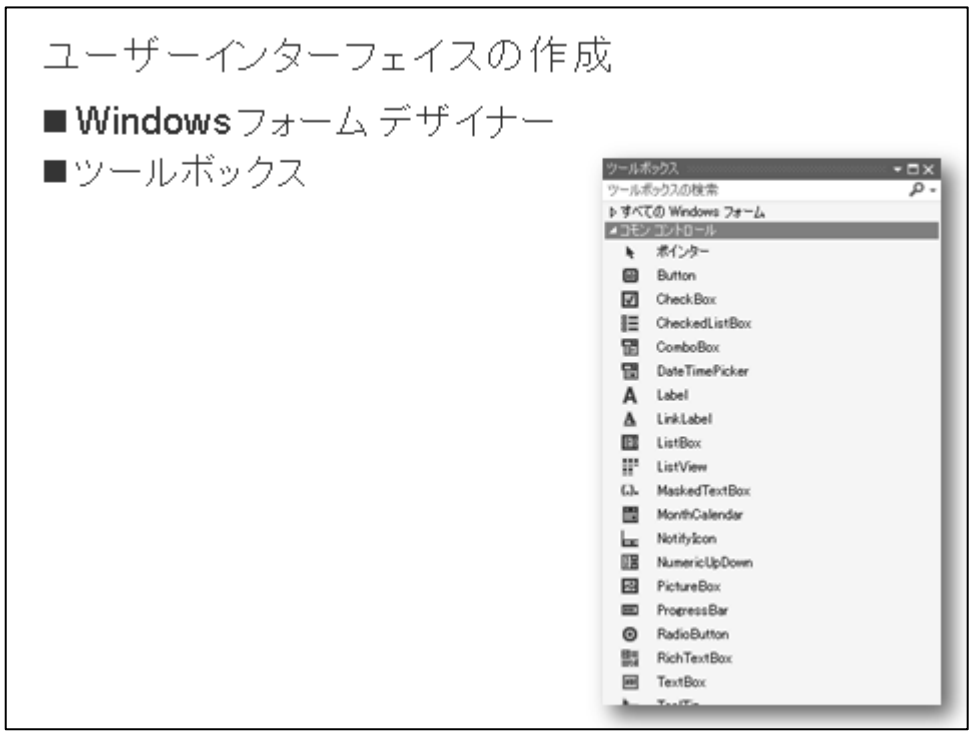
Visual Studio を起動すると、既定では「スタートページ」が表示されます。スタートページでは、新しいプロジェクトを作成したり、既存のプロジェクトを開いたりすることができます。

■　アプリケーションテンプレート

Visual Studio には、さまざまな種類のアプリケーションやライブラリの開発をサポートするため、複数のアプリケーションテンプレートが用意されています。新しいプロジェクトを作成するには、利用するテンプレートの種類を選択します。

■　作成手順

1. Visual Studio を起動します
2. スタートページで「新しいプロジェクト...」を選択するか、[ファイル]メニューから－[新規作成]－[プロジェクト...]を選択します
3. 「新しいプロジェクト」ダイアログが表示されるので、[インストールされたテンプレート]の一覧から[Visual Basic]または[Visual C#]を選択します。ダイアログ中央部のテンプレート一覧から、[Windows フォームアプリケーション] を選択します
4. [プロジェクト名] ボックスにプロジェクト名を入力し、必要に応じて [ソリューション名] ボックスにソリューション名を入力します
5. [OK]ボタンをクリックします。プロジェクト及びソリューションが作成されます



■　Windowsフォームデザイナー

Windows フォームのユーザインタフェースを作成するには、Windows フォームデザイナーを利用すると簡単に作成が行えます。[新しいプロジェクト]ダイアログで、Windows フォームアプリケーションのテンプレートを選択してプロジェクトを作成すると、Windows フォームアプリケーションのプロジェクトが作成され、既定のフォーム(Form1)が生成されます。

■　ツールボックス

Windows フォームデザイナーでは、ユーザインタフェース要素として、ボタンやテキストボックスなどの「コントロール」を利用して、ユーザインタフェースを作成します。コントロールの実体は、クラスから生成されるオブジェクトです。ツールボックスから、Windows フォームにコントロールをドラッグアンドドロップすると、オブジェクトを生成するコードも自動的に生成されます。

■　プロパティとは

プロパティとは、利用者から見るとフィールド、開発者はメソッドとして実装する、「論理的なフィールド」です。Windows フォームアプリケーションでは、コントロールやフォームなど各オブジェクトの色、表示文字列、サイズなど、オブジェクトのさまざまな属性がプロパティとして利用されます。

■　プロパティウィンドウの利用

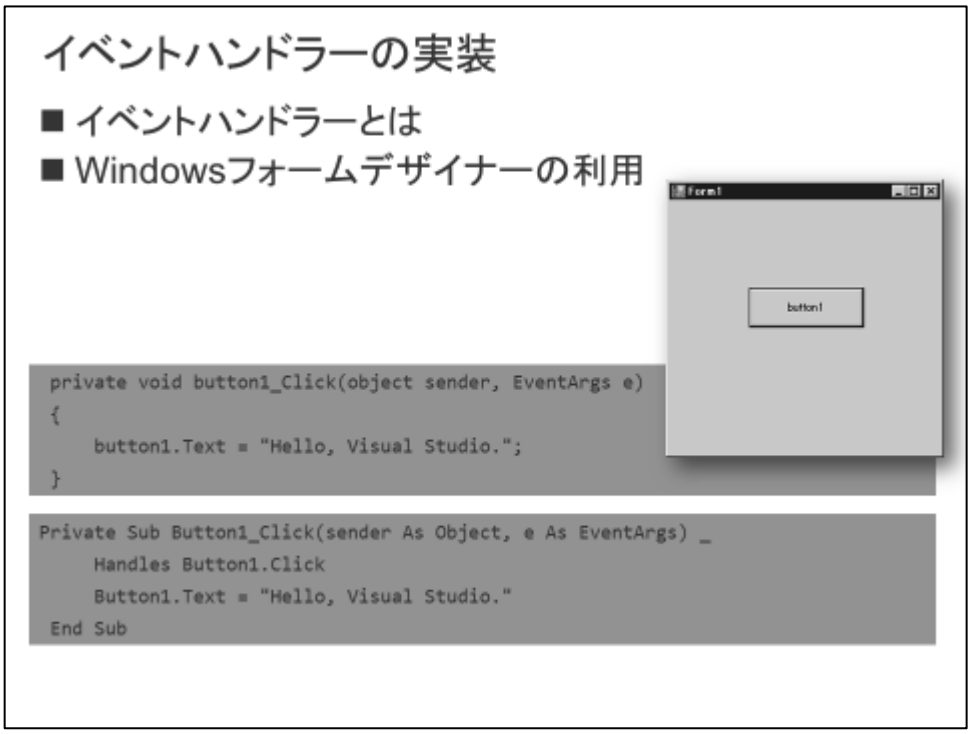
プロパティは、C#/VB コードから値の設定や取得を行えますが、Windows フォームアプリケーションの初期値として、プロパティウィンドウからグラフィカルに設定していくこともできます。コードからプロパティの値を設定するには、以下の書式を利用します。

● **C#**

```
オブジェクトの変数名.プロパティ名 = 設定する値;
```

● **Visual Basic**

```
オブジェクトの変数名.プロパティ名 = 設定する値
```

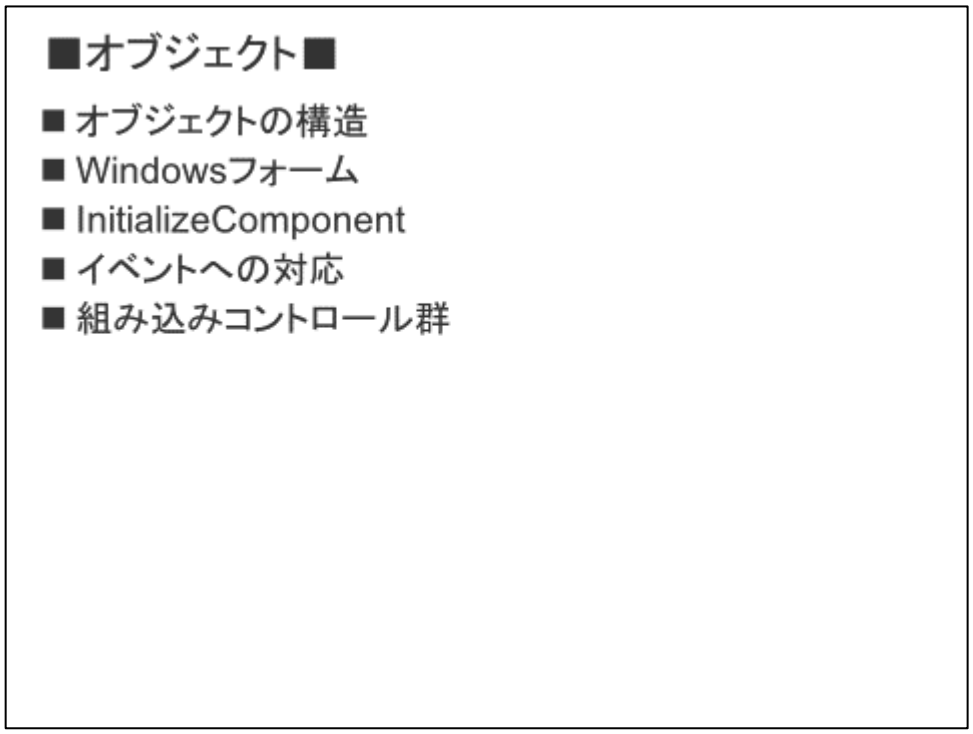


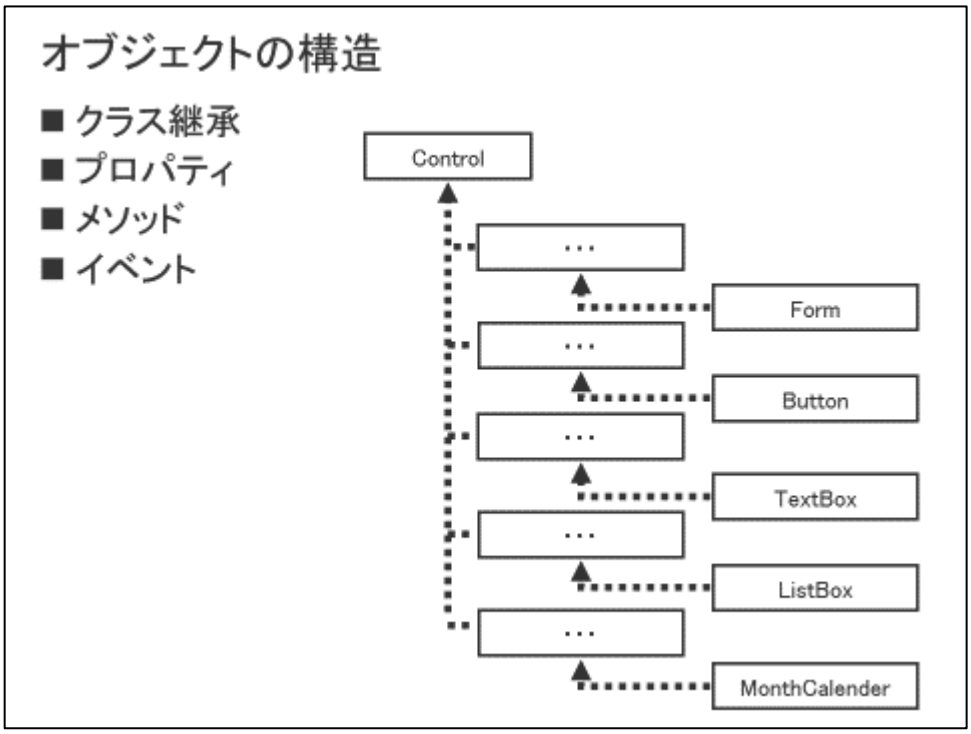
■　イベントハンドラーとは

イベントハンドラーとは、イベントに応答するためのメソッドです。Windows フォームアプリケーションでは、Windows フォームと各コントロールに対してプロパティを設定した後で、イベントハンドラーとして実行するコードを追加できます。イベントは、各オブジェクトでさまざまなアクションが実行されたときに発生します。たとえば、ユーザーがボタンをマウスでクリックすると、そのボタンの Click イベントが発生します。

■　Windowsフォームデザイナーの利用

各コントロールの既定のイベントハンドラーは、Windows フォームデザイナーでオブジェクトをダブルクリックすることで生成できます(詳細は後述)。


■オブジェクト■
■ オブジェクトの構造
■ Windowsフォーム
■ InitializeComponent
■ イベントへの対応
■ 組み込みコントロール群



■　クラス継承

Windows フォームや各コントロールなどの、ユーザインタフェースを構成するオブジェクトは、各コントロールのためのクラスから生成されます。コントロールのための各クラスは、多くのものが System.Windows.Forms.Control クラスを直接または間接的に継承します。そのため、生成されたオブジェクトが提供する機能は、同じ機能であれば同じ名前で利用することが出来ます。以下に System.Windows.Forms.Control クラスで定義されている代表的なメンバーを示します。

■　プロパティ

●　(Name)

コントロールの名前(識別子)を示します。フォームでは、フォームのクラス名として扱われます。

●　Text

コントロールに関連付けられている文字列を示します。Form クラスではタイトルバーの文字列、Button クラスや TextBox クラスではコントロール上の文字列です。

●　Font

コントロールに関連付けられている文字列のフォントを示します。

●　BackColor

コントロールの背景色を示します。

●　Anchor

コンテナ(コントロールの配置先となるオブジェクト)の各辺(上下左右)との位置関係を示します。

●　Dock

コンテナへのドッキングの有無およびドッキングする場合の辺を示します。

● **Visible**

コントロールやコンテナとなっているコントロールの表示/非表示を制御します。

■ メソッド

● **Focus**

コントロールにフォーカスを割り当てます。

● **Hide**

コントロールを画面上で非表示にします。

● **Show**

コントロールを画面上に表示します。

● **SetBounds**

コントロールの位置とサイズを変更します。

● **FindForm**

コントロールが配置されているフォームを取得します。

■ イベント

● **Click**

コントロールがクリックされたときに発生します。

● **Enter**

コントロールでフォーカスが有効になると発生します。フォームでは、Enter イベントの代わりに、Activated イベントを使用します。

● **Leave**

コントロールがフォーカスを失うと発生します。フォームでは、Leave イベントの代わりに、Deactivate イベントを使用します。

● **MouseEnter**

コントロールの領域内にマウスポインタが入ったときに発生します。

● **MouseHover**

コントロールの領域内にマウスポインタが重なったときに発生します。

● **MouseLeave**

コントロールの領域内からマウスポインタが出たときに発生します。

● **Resize**

コントロールのサイズが変更されたときに発生します。

これらの各メンバーは、 System.Windows.Forms.Control クラスから継承したものを派生クラス側でオーバーライドしている場合もあります。