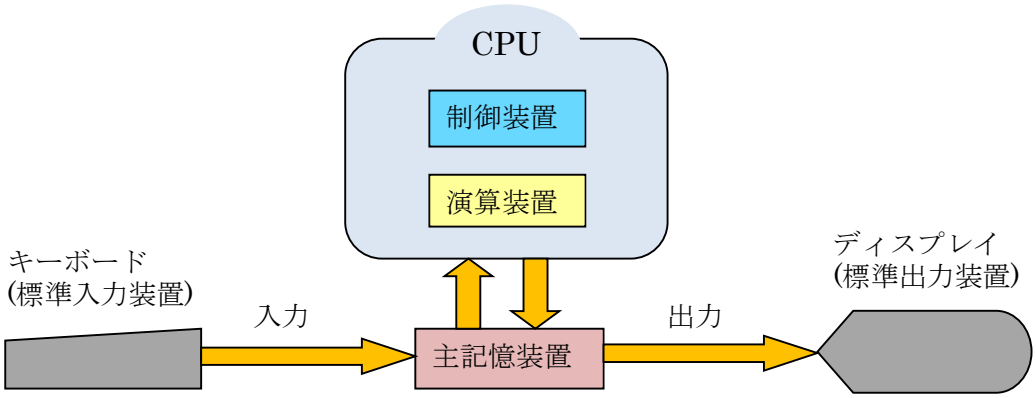
2024 年 4 月 18 日（木）実施

コンピュータのハードウェアとプログラム

機械（ハードウェア）としてのコンピュータの特徴

現在，普通に使われているコンピュータは**ディジタルコンピュータ**（2 値論理に基づくコンピュータ）であり，次の様な特徴を備えている。

1. 電子デバイス（主に半導体集積回路を用いた装置）により，データ処理を行う。
2. 2 値論理に基づいてデータを扱う。（電圧の高低，電荷や窪みの有無，磁化の向き等）
3. 主記憶装置（メインメモリ）に一時的に読み込んだプログラムを中央処理装置（CPU）で解釈・実行することにより，データ処理を行う。　【プログラム記憶方式】



注）ディスプレイには，ビデオ RAM（VRAM）に書き込まれたデータが表示される。

プログラムとは（機械語，アセンブラ，高級言語）

プログラムとは，**コンピュータに処理の手順を指令するもの**である。コンピュータが動作するための命令の種類とその書式は，CPU の設計段階で決定される命令セット（Instruction Set Architecture）に基づく。この命令セットを用いて作成されたプログラムを**機械語プログラム**と呼び，命令及びデータは 2 進数で表される。コンピュータが動作可能なプログラムは，機械語で記述されたもののみである。なお，命令セットは CPU ごとに異なるので，ある CPU 用の機械語プログラムは異なる設計の CPU 用には流用出来ない。

また，2 進数でプログラムを作成していくのは困難なので，命令に名前付けをした，**アセンブラ**（アセンブリ言語）が用いられる。アセンブラプログラムは機械語プログラムと 1 対 1 に対応するので，やはり CPU の設計に依存する。

通常のプログラミング（プログラム作成）には，CPU の設計に依存しない，**高級言語**と呼ばれるプログラム言語を用いることが多い。高級言語の多くは命令を英単語やその組み合わせまたはその省略形で表す人工的な言語である。高級言語で記述されたプログラムの各行は機械語に 1 対 1 には対応していないので，翻訳（コンパイル）・編集（リンク）という作業を通じて，コンピュータが実行可能な機械語プログラムに変換される。この作業は大変複雑なため，それぞれ**コンパ**

イラ，リンカというプログラムを通じて行われる。この場合，元の高級言語プログラムは**ソースプログラム**と呼ばれる。（ソースプログラムの各行を解釈・実行していく**インタプリタ**と呼ばれるプログラムも存在する）

この授業では，高級言語としては C#言語を用い，**Windows フォームアプリケーションソフトウエア**の作成について初歩から学ぶ。

Visual C# の特徴

Visual C#とは

Visual C#は，Microsoft 社がインターネット上でコンピュータの連携を図る**.NET（ドットネット）構想**に沿って構築した開発支援ツール Visual Studio（当初は Visual Studio .NET）に含まれるプログラミング言語 C#の処理系で，C#言語は C 言語及び C++言語をベースとして拡張し，Java 言語の機能を採り入れたものである。

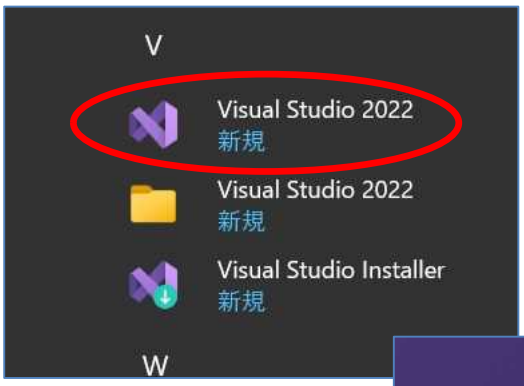
Visual Studio で開発したアプリケーションソフトウェアは**.NET Framework** という実行環境上で中間言語に翻訳されて実行される。

この授業では，Visual Studio Community 2022 を利用して，C#言語によるプログラミングを行う。

Visual Studio Community 2022 の使い方

1）Visual Studio Community 2022 の起動

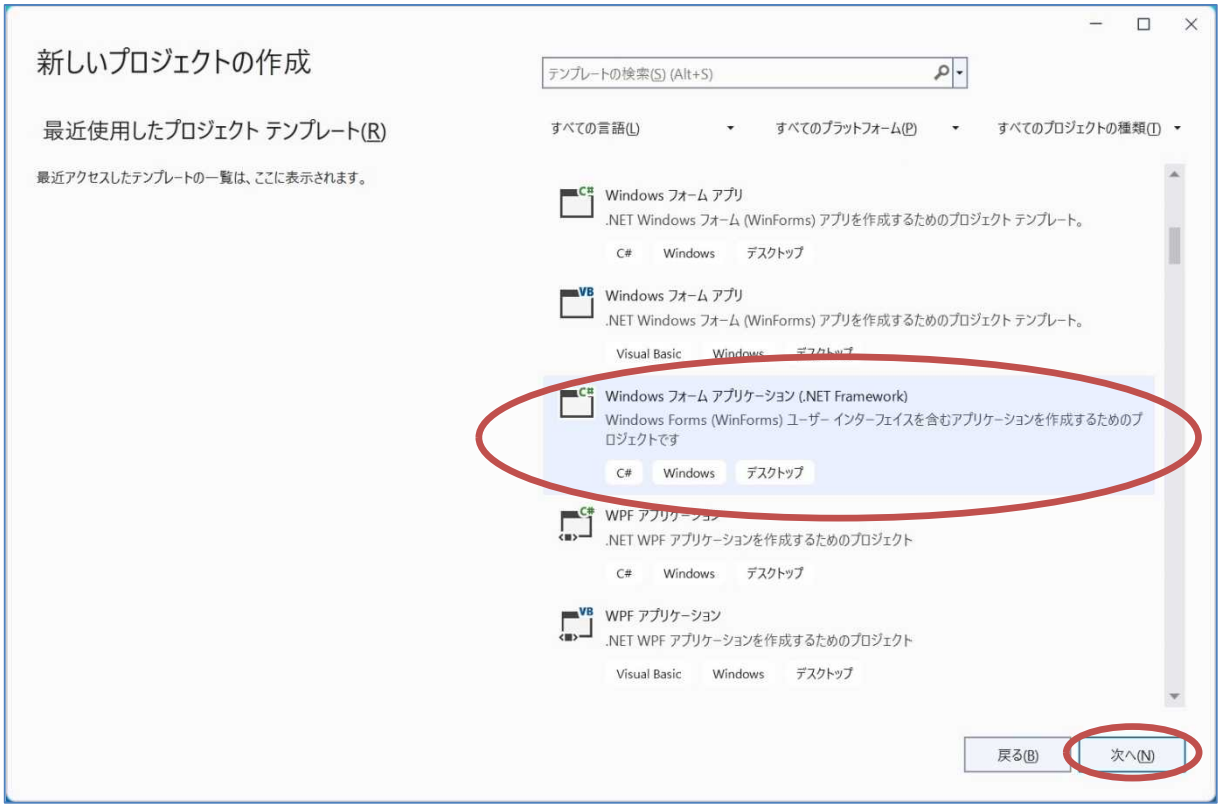
スタートメニューから『Visual Studio 2022』を選択して起動する。



2) プロジェクトの作成

起動後に，次のページのダイアログが出るので，新規のアプリケーションソフトウェアを作成するには，『新しいプロジェクトの作成』を選択する。

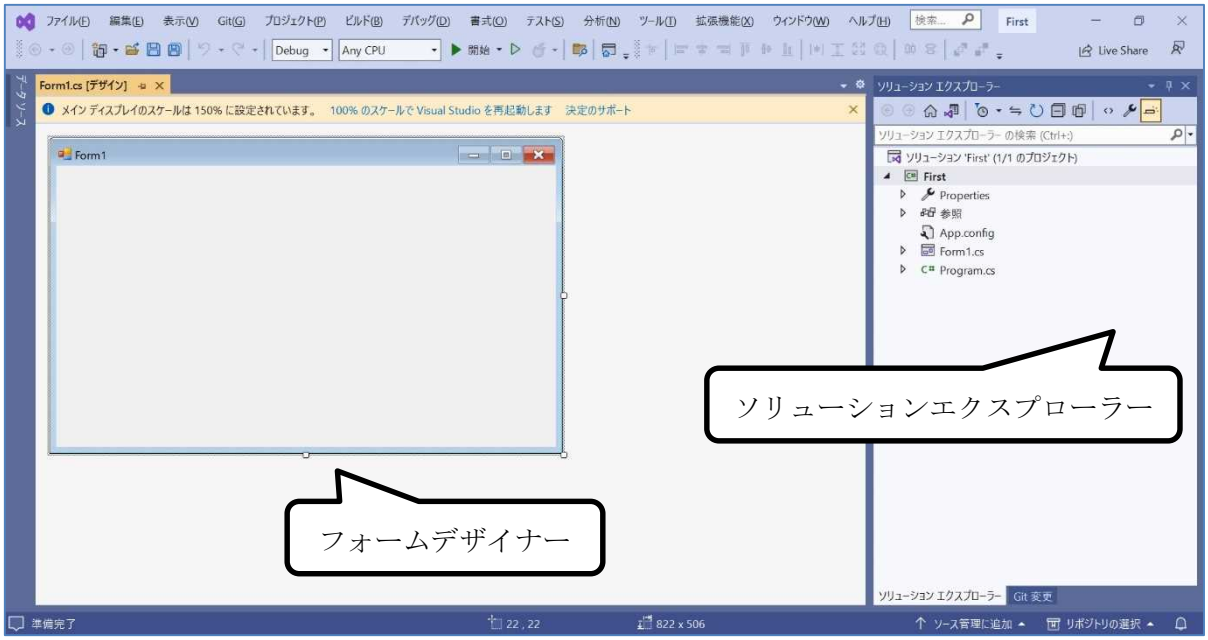
次の選択画面の右側のスクロールバーを下にずらして，『Windows フォームアプリケーション(.NET Framework)』（C# のアイコンが表示されている項目）を選択して，『次へ』ボタンをクリックする。



次ページの図の様なダイアログが開くので，『プロジェクト名』を「WindowsFormsApp1」から「First」に書き換えて，『作成』ボタンをクリックする。

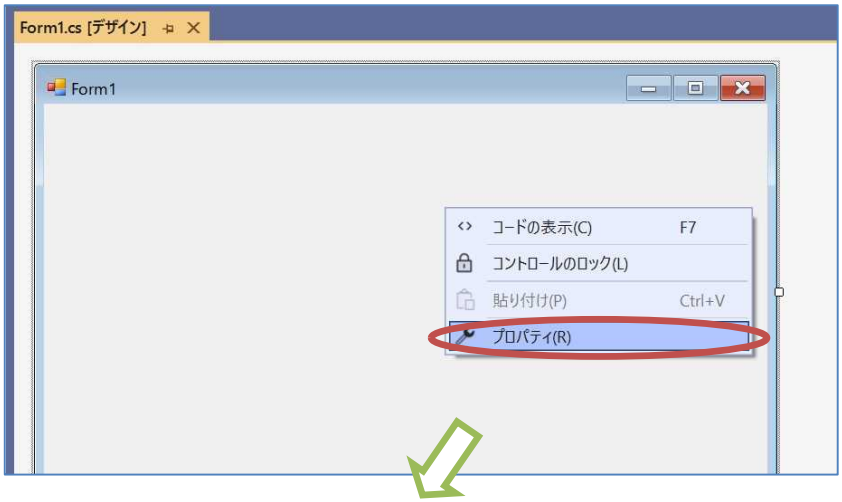
＊書き換え後，プロジェクトは C:Users¥ユーザー名¥source¥repos¥First¥First に作成される。

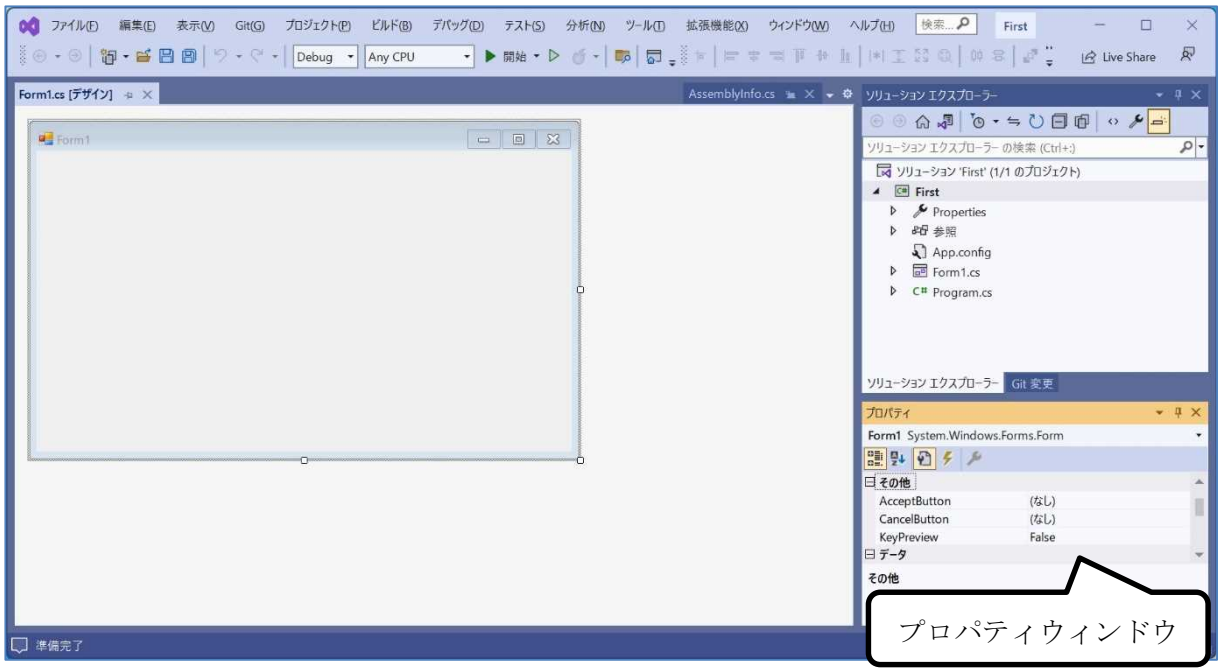
↑フォルダには『ユーザー』と表記される。



ここで，フォームデザイナーを右クリックして表示されるポップアップメニューで『プロパティ』をクリックして，プロパティウィンドウを表示させる。

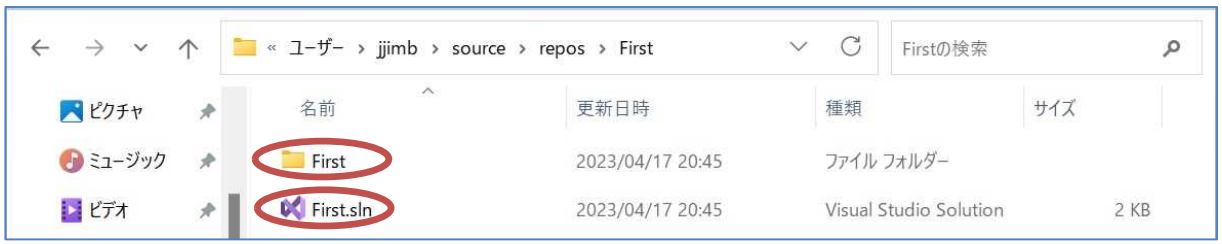
（図は次のページ）





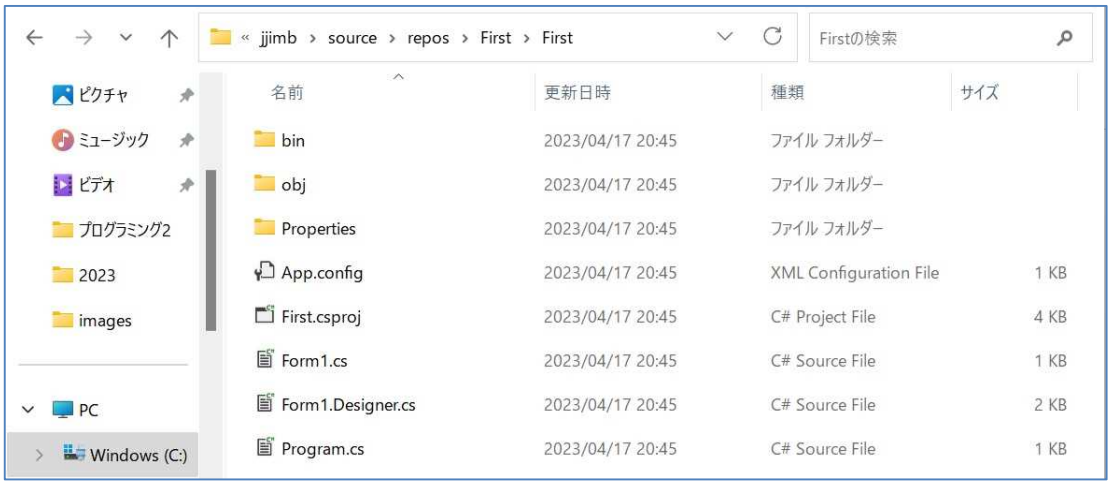
この時点で，フォームを利用するアプリケーションソフトウェアの枠組みは出来上がっている。このフォームにボタン等の部品（**コントロール**と呼ばれる）を配置し，ボタンをクリックした時の動作を**イベントハンドラ**と呼ばれる**メソッド**（処理をひとまとめにしたもの）に記述するといったスタイルでアプリケーションソフトウェアを開発していく。

プロジェクトが作成出来たら，エクスプローラーで，C ドライブの[ユーザー] → [ユーザー名] → [source] → [repos] → [First]と辿ると，ソリューションの定義を記述した拡張子.sln のファイルと『First』フォルダとが作成されている。



この中の『First』フォルダを開くと，拡張子.cs のソースプログラムのファイルが作成されてい

る。その中で，フォームのデザインに関するものが『Form1.Designer.cs』で，プログラムを記述するのが『Form1.cs』である。これらの編集は p.3 の図の様な統合開発環境で行う。

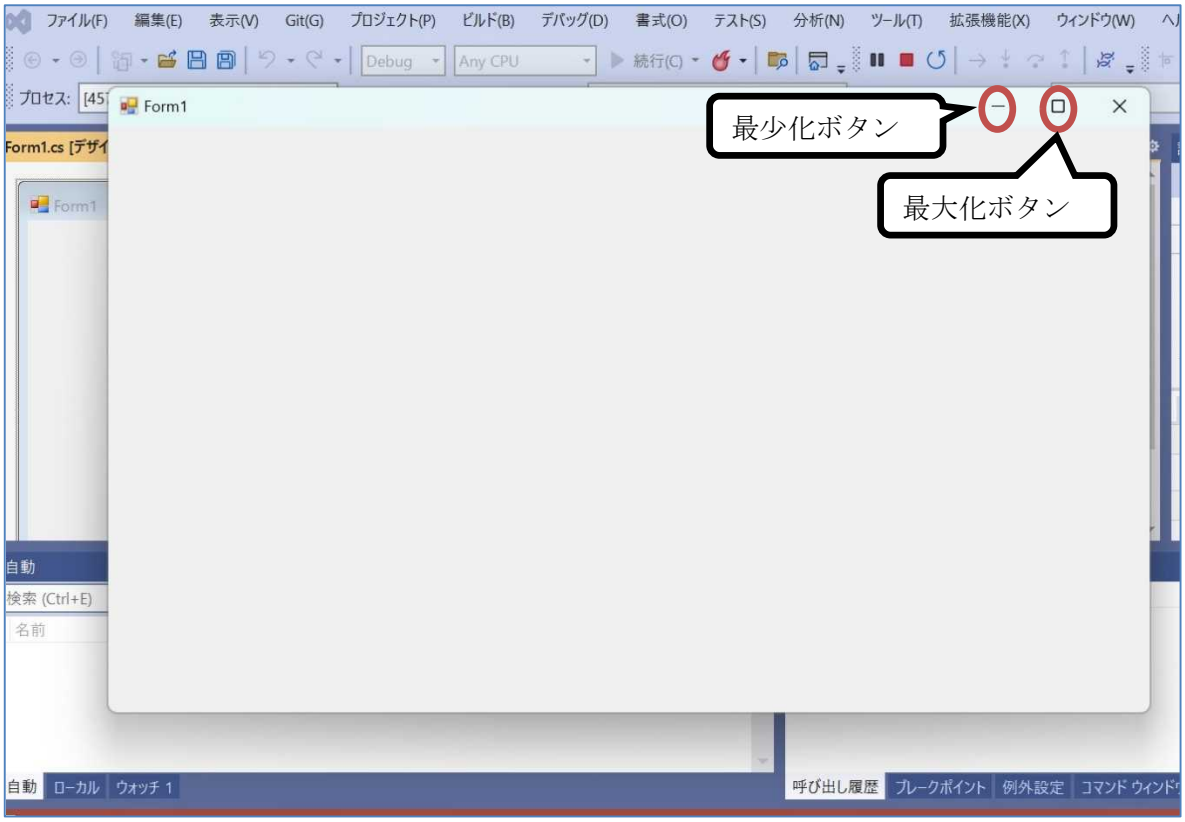


3) プログラムの実行

この状態でプログラムを実行してみる。先ず，『すべてを保存』のボタンを押して，その後に『開始』ボタンを押す。

Windows フォームアプリケーションプログラムが実行されると，次のページの図の様な窓が開かれるので，最大化ボタン及び最小化ボタンが機能することを確かめる。

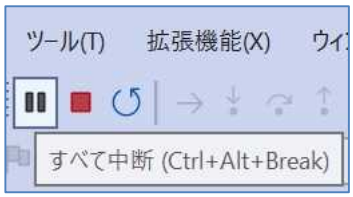
（図は次のページ）

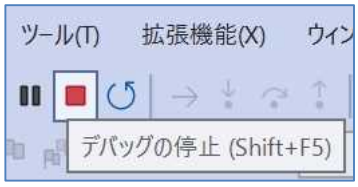


最大化ボタンをクリックすると，窓が画面一杯に広がり，元に戻るボタンに変わるので，それをクリックすると元に戻る。

最小化ボタンをクリックすると，窓が閉じるので，元に戻すにはタスクバー上に表示されたアイコンをクリックする。

プログラム実行中には，次の様なツールバーが表示される。





３つのボタンの意味に関しては，マウスポインタを近づけると，ヒントが表示される。

＊プログラムの実行中にはデザインもプログラムも編集出来なくなるので注意すること。

このツールバーで『デバッグの停止』を押すか，窓の閉じるボタン（×）を押すかして，プログラムを終了させる。

C# プログラムの基礎

基本構造

1) クラス

Visual C#のプログラムの基本単位を**クラス**と呼ぶ。Windows フォームアプリケーションを作成する際，プロジェクトを作成すると生成されるファイルのうち，Form1.cs を例にとれば，そのクラス名は Form1 である。クラスは class **キーワード**を用いて宣言する。

【Form1.cs】

```
using System;
 （中略）
using System.Windows.Forms;

namespace First
{
    public partial class Form1 : Form
    {
        public Form1()
        {
            InitializeComponent();
        }

        private void Form1_Load(object sender, EventArgs e)
        {
            // Form1 が読み込まれた際の処理
        }

        private void button1_Click(object sender, EventArgs e)
        {
            // button1 がクリックされた際の処理
        }

    }
}
```

初期状態ではこれらは無く，必要に応じて付け加える。

2) ブロック

中括弧『{』から『}』に囲まれたものを**ブロック**と呼ぶ。ブロックには複数の要素をひとまとめにする機能がある。

3) メンバ

クラスの中にある要素を**メンバ**と呼ぶ。メンバにはデータを扱うフィールドと呼ばれる要素，処理をまとめた**メソッド**（上の例では Form1_Load 及び button1_Click が**イベントハンドラ**と呼ばれるメソッド）及び初期化に用いる**コンストラクタ**（上の例では Form1）等がある。

4) コメント

プログラムの動作には影響を与えずに，注釈を書き込めるものを**コメント**と呼ぶ。『//』が書かれると，そこから改行の手前までがコメントとなる（複数行に渡るコメントを書くには，『/*』と『*/』とを用いて，それらの間に書く）。

＊ プログラム記述上の注意　コメントや文字列を表示する為の記述以外は，キーボードにある英字，数字及び記号の範囲で，半角文字で記述する。また，文末にはセミコロン『;』を書く。

名前空間

Visual C#で様々なプログラムを作成していく上で，他のプログラムと区別し，複数のクラスをまとめて管理する為に**名前空間**が用いられる。上の Form1.cs では，プロジェクト名として命名した First が namespace **キーワード**によって名前空間に設定されている。

また，.NET Framework の構成要素のクラスライブラリの名前空間には **System 名前空間**が含まれている。System 名前空間の下には，Windows 名前空間，更にその下には Forms 名前空間があるという様に階層的な構造がある。この構造を System.Windows.Forms の様にドットで繋いで表す。なお，フォームにラベルの様なコントロール（部品）を配置する際に，**完全修飾名**では System.Windows.Forms.Label と書くことになるが，**using ディレクティブ**を記述しておけば，そこに書かれた名前空間の記述を省略することが出来る（Form1.cs の例では using System.Windows.Forms;）。

本日の課題

【フォームのデザイン】

Windows フォームアプリケーション作成の第１歩は，フォームの大きさやフォームに配置するコントロールの種類や数を設計することである。本日はその基礎として，ツールボックスの使い方，フォームやコントロールのプロパティの設定の仕方及びプログラム作成の初歩を学ぶ。

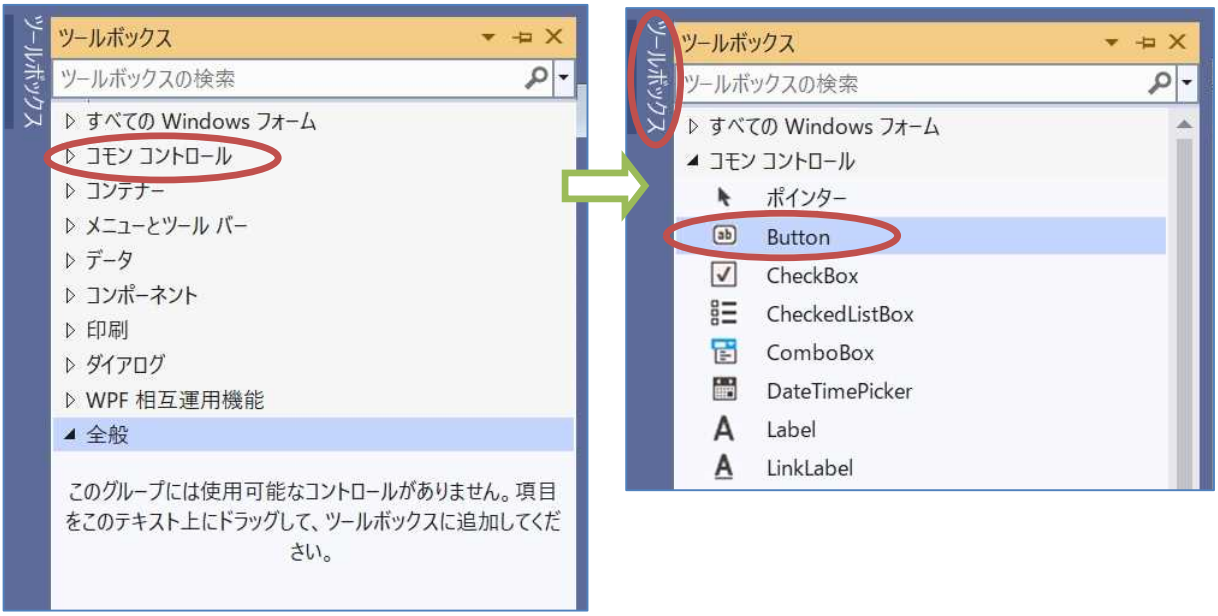
手順

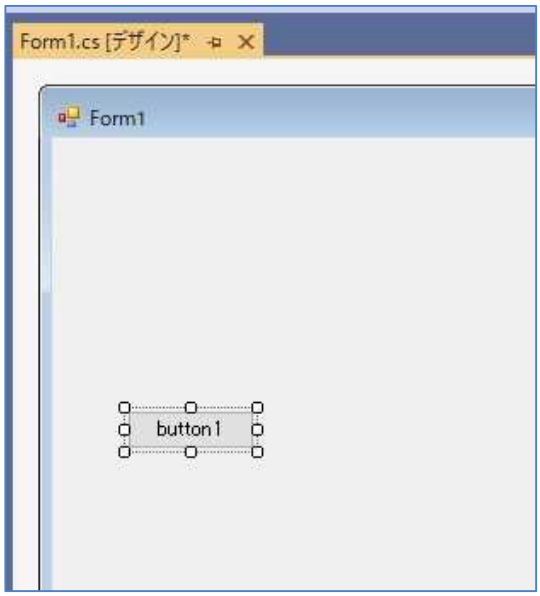
1）コントロールの配置

『Visual Studio Community 2022 の使い方』の箇所で作成した First を利用してコントロールを配置していく。

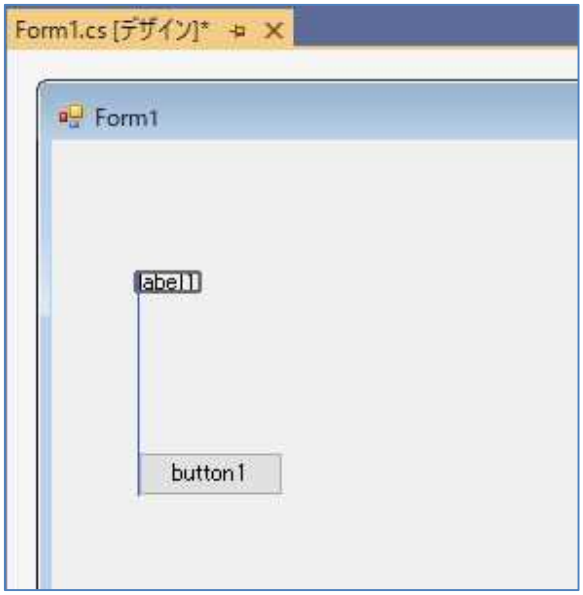
先ず，『表示』タブのメニューを開いて『ツールボックス』をクリックする。

次に『コモンコントロール』を展開して，『Button』を選択してからフォームデザイナー上の『Form1』をクリックすると，ボタンがフォームに配置される。続いてボタンをドラッグして左に寄せる。今後メニューを出すには，左にある縦の『ツールボックス』タブをクリックする。



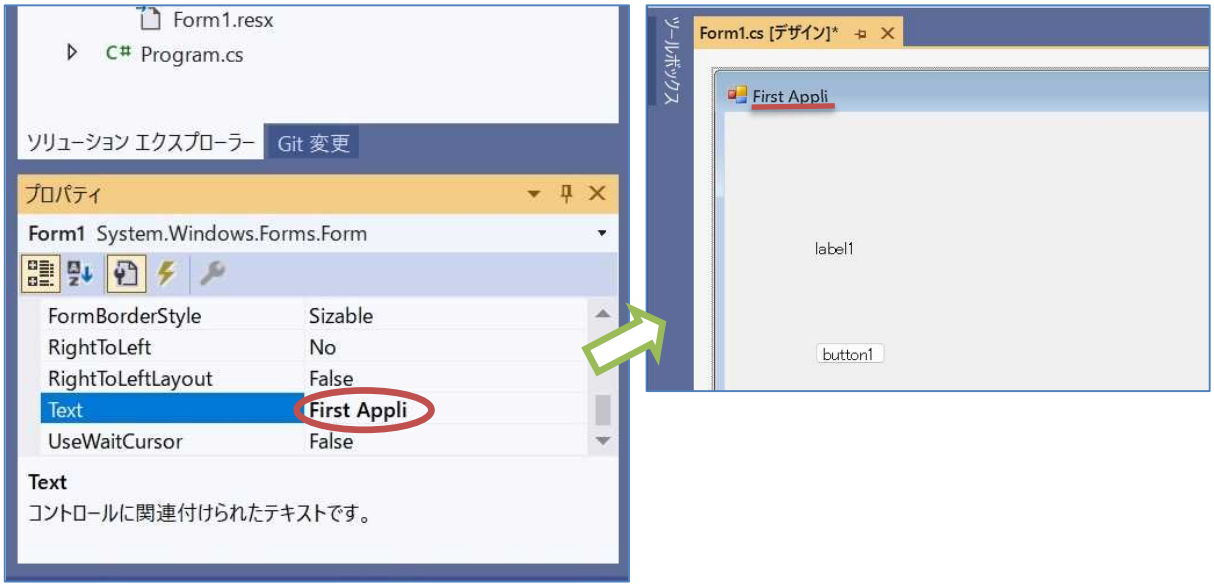


同様にして，『ツールボックス』タブをクリックしてメニューを出し，『Label』を選択してからフォームデザイナー上の『Form1』をクリックし，配置されたラベルをドラッグして左に寄せる。

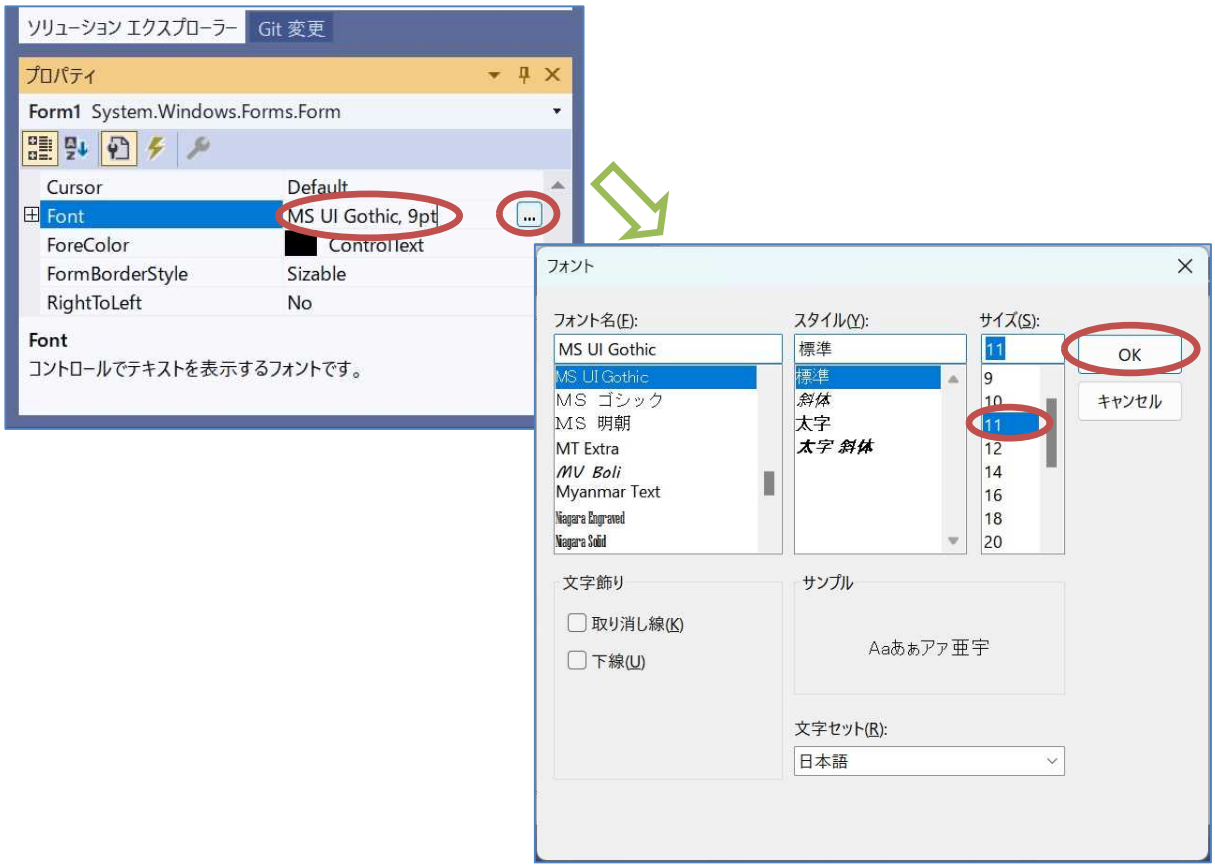


2) プロパティの設定

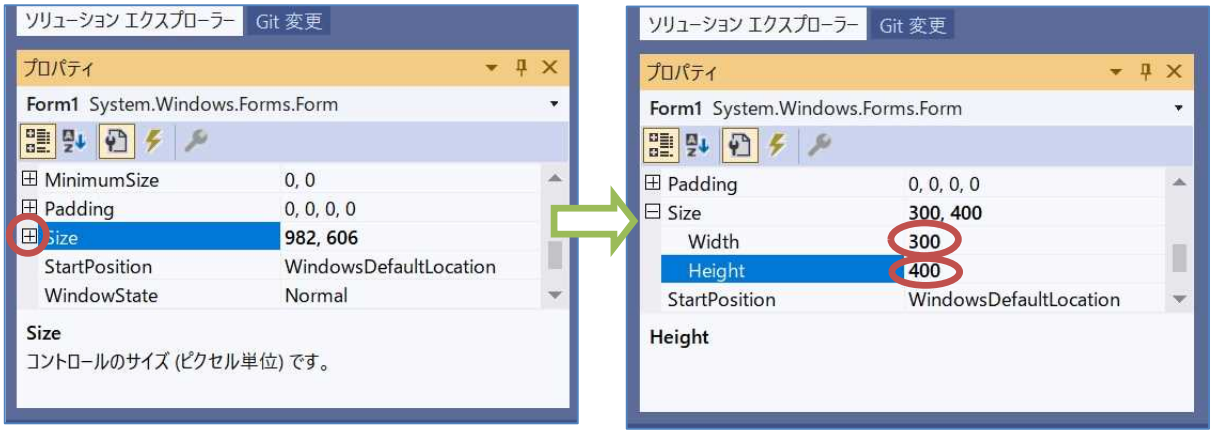
フォーム上のコントロールの無い箇所をクリックすることでフォームを選択し，『プロパティウィンドウ』で，『Text』プロパティを探し，その値を「Form1」から「First Appli」に書き換える。確定するには Enter キーを打つ。



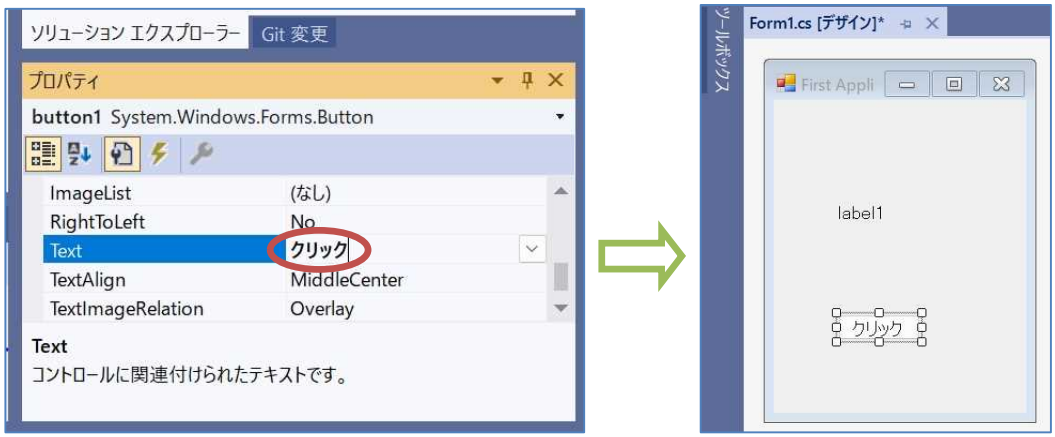
次に『Font』プロパティを探し，その値を変更する。値の欄をクリックするとオプション設定用の『…』ボタンが表示されるので，それをクリックする。出てきたダイアログでサイズを 11（単位はポイント）を選択し，『OK』ボタンをクリックする。



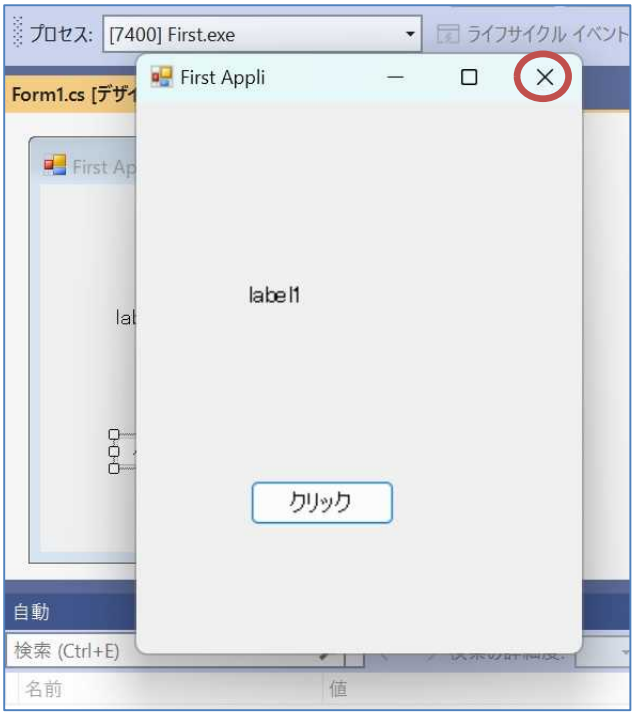
更に，『Size』プロパティを探し，その値を変更する。このまま変更しても良いが，左側の『+』ボタンをクリックして展開し，Width（幅）及び Hight（高さ）を個別に設定する方が間違えにくい。ここでは，Width の値を 300（単位はドット），Hight の値を 400 とする。



最後に，『フォームデザイナー』上でボタンをクリックして選択し，『プロパティウィンドウ』で，『Text』プロパティを探し，その値を「button1」から「クリック」に書き換える。



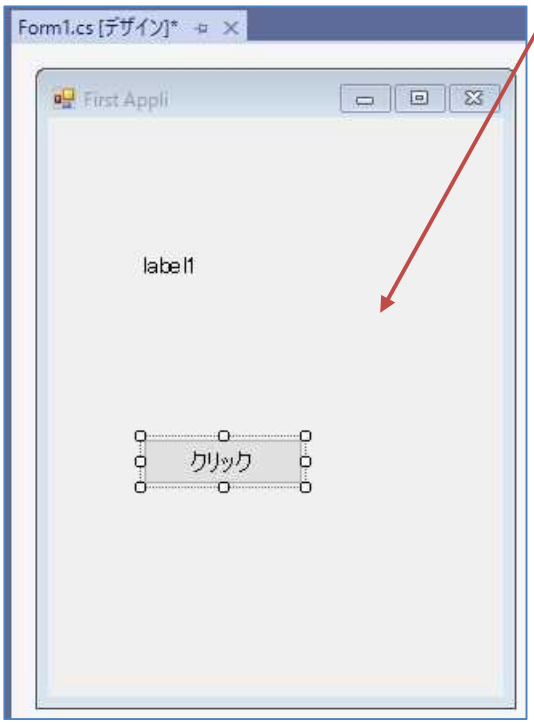
『Form1.cs の保存』ボタンを押してから，『開始』ボタンを押して，プログラムを実行してみる。



確認が済んだら，閉じるボタンを押してプログラムを終了する。

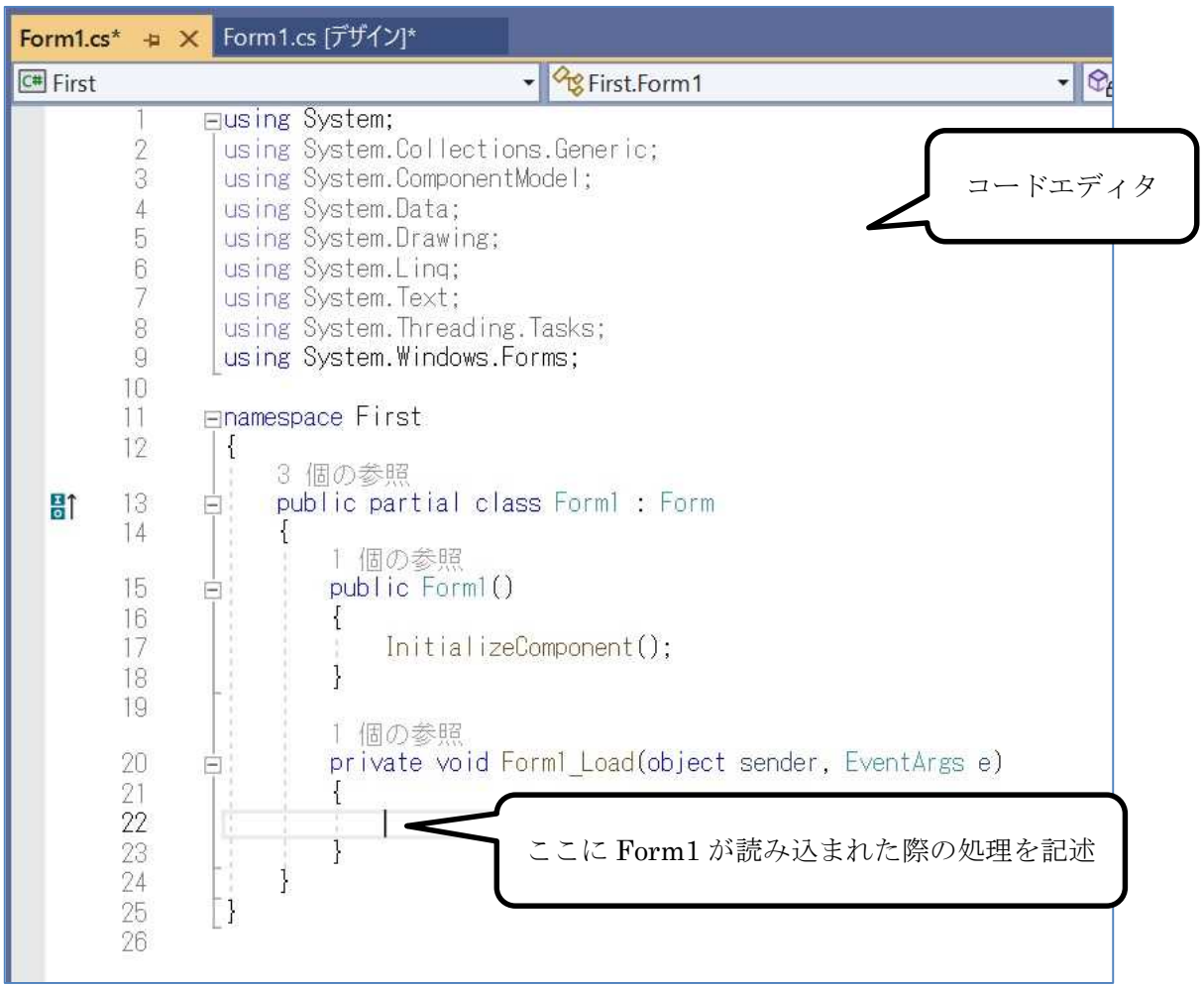
1）プログラムの作成（コーディング）

フォームデザイナー上でラベルやボタンの無い箇所をダブルクリックする。



コードエディタが開き，Form1.cs のプログラムのソースコードが表示される。Form1_Load メソッド（）の枠組みが作成されているので，Form1 が読み込まれた際の処理を記述して行く。

＊ イベントハンドラと呼ばれるメソッドは統合開発環境側で管理されているので，この方法を取らずに手入力すると機能しない。




```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace First
{
    3 個の参照
    public partial class Form1 : Form
    {
        1 個の参照
        public Form1()
        {
            InitializeComponent();
        }

        1 個の参照
        private void Form1_Load(object sender, EventArgs e)
        {

        }
    }
}
```

イベントハンドラ Form1_Load メソッドのブロック内に Form1 が読み込まれた際の処理（赤枠の部分）を記述する。

```
private void Form1_Load(object sender, EventArgs e)
{
    label1.Text = "ボタンをクリックしてください。";
}
```

ここでは，label1 という名前のラベルの Text プロパティに**文字列リテラル**（二重引用符で挟まれた，固定された文字列）を当てはめて（**代入**と呼ぶ）設定している。代入には『=』を用い，文末には『;』（セミコロン）を用いる。

コードエディタでは文字列を途中まで打つと候補が現れる補完機能がある。適切なものを選んで Enter キーを打つとプログラムにそれが入力される。

次に，フォームデザイナー上で『クリック』が表示されているボタンをダブルクリックして，Form1.cs のプログラムのソースコードを表示する。イベントハンドラ button1_Click メソッドの

ブロック内にボタンがクリックされた際の処理（次のページの赤枠の部分）を記述していく。

```
private void button1_Click(object sender, EventArgs e)
{
    label1.Text = "ボタンがクリックされました。";
}
```

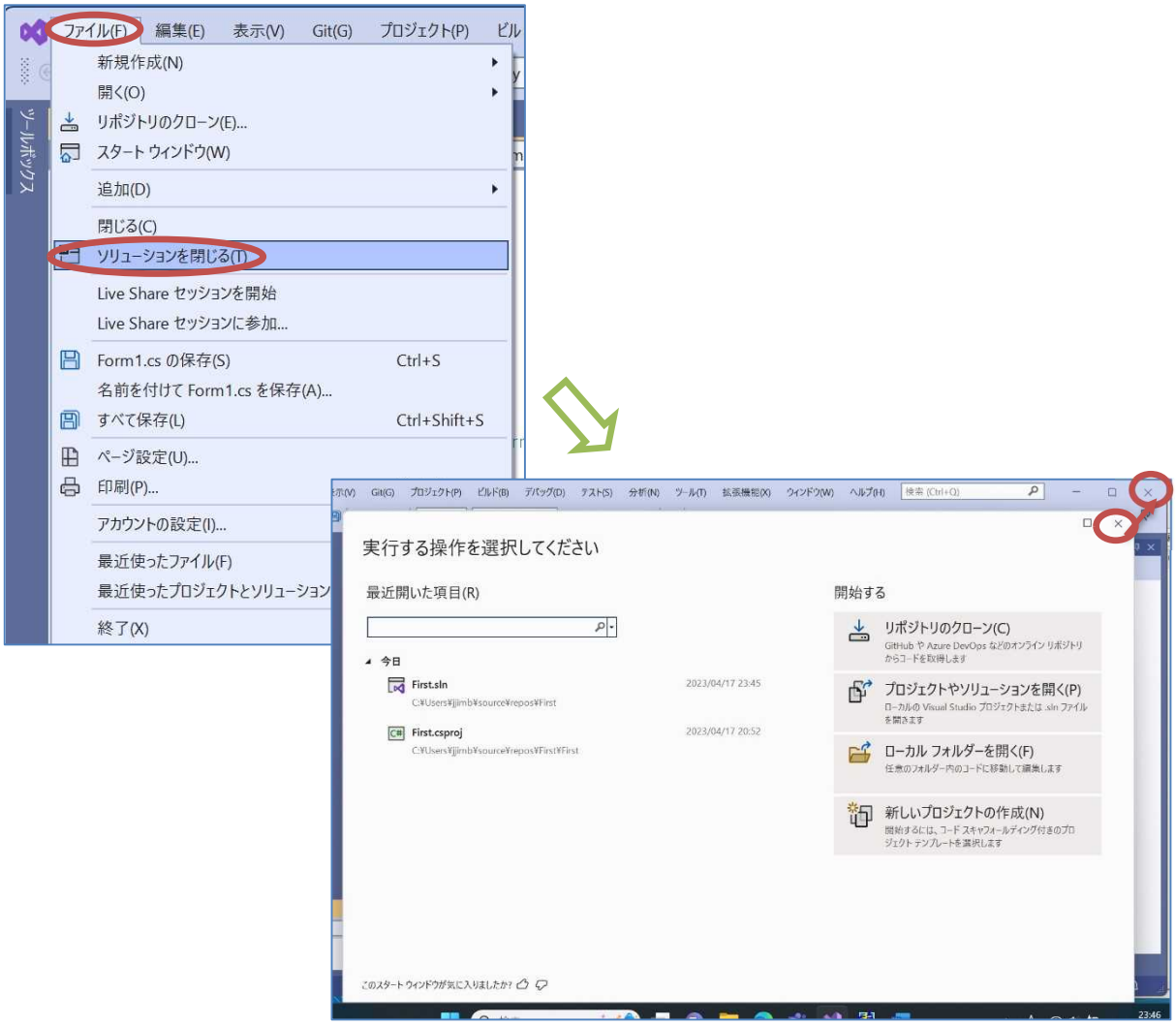
ここで，『すべて保存』のボタンを押してから，『開始』ボタンを押して，プログラムを実行する。

これらに相当する図（**スクリーンショット**）を自分のアプリの実行した直後及びボタンをクリックした後に作成して，課題提出用にする。なお，タイトルの背景色等は Windows の環境設定によって異なる。それぞれのスクリーンショットの名称は『第 1 回実行結果 1.jpg』，『第 1 回実行結果 2.jpg』とする。

＊ スクリーンショットの作成には，Alt キーを押しながら PrtScr キーを打って画面のイメージを取り込み，アクセサリの**ペイント**で jpg ファイルとして名前を付けて保存する。

次回はこのプロジェクトの作業の続きを行わない（別のプロジェクトを新規に作成する）ので，『ファイル』→『ソリューションを閉じる』を選択し，ダイアログの閉じるボタン，開発環境の閉じるボタンを押して終了する。

（図は次のページ）



提出物：

1) フォームのデザインファイル Form1.Designer.cs をメールに添付して提出する。
2) コードエディタで編集したソースファイル Form1.cs をメールに添付して提出する。
3) 実行の初期状態のスクリーンショット第 1 回実行結果 1.jpg（.png も可）をメールに添付して提出する。
4) 実行の最終結果のスクリーンショット第 1 回実行結果 2.jpg（.png も可）をメールに添付して提出する。
5) 質問を記述したファイル Prog2_Questions_1st.txt に解答を書き込んで保存し，メールに添付して提出する。

＊ Form1.Designer.cs 及び Form1.cs の格納されている場所については p.6 を参照