Windows ソフトを作ってみよう！

1 目的・ねらい

プログラミング言語 C#で、シンプルな GUI(Graphical User Interface)アプリケーションを作ることで、Windows アプリケーション開発の最初の一歩を踏み出します。 GUI をシンプルに書ける C#の長所を生かして、難しいことは置いておいて、まずプログラミングの楽しさ、そして気軽に作れるものだと感じてもらうことが第一のねらいです。

2 実施内容の概要

Windows(Vista 以降)に最初から入っている C#開発環境を使って、簡単なフォームベースのアプリケーションを作成します。 GUI ですので、フォームのサイズはどうか、ボタンの配置はどうか、といった外観のチェックなども必要になります。DotView などを使って自分で確認することも出来ますが、複雑なものを作りませんので、口頭で伝えたり、LEGO の様なブロックなどを使って説明する、というのもアリです。

3 講師用の実施手順の詳細

3-1 準備することがら、物品など

1. Windows の設定で、「拡張子を表示する」ようにする。

2. C#のコンパイラ(csc.exe)を探します。
- 通常、C:\Windows\Microsoft.NET\Framework\v(バージョン番号)にあります。
- 例えば、C:\Windows\Microsoft.NET\Framework\v4.0.30319
3. Windows の設定で、環境変数 PATH に上記の場所を追加します。
4. C#プログラムファイルの拡張子.cs を、生徒が使うエディタ(「MYEDIT」「メモ帳」など)で開くようにしておきます。
5. Documents フォルダなどの下に、プログラムを作成するフォルダを MyFirstCS といった名前をつけて作成しておきます。
6. コマンドプロンプトを立ち上げ(Win キー+R→cmd と入力→Enter キー)、上記のフォルダに移動します。
- コマンドプロンプト起動: Win キー+R→cmd と入力→Enter キー
- Documents フォルダの MyFirstCS に移動: コマンドプロンプトで、cd %userprofile%/Documents/MyFirstCS と入力→Enter キー

7. ダイアログの外観を確認するために DotView(DV-2) を使う場合は、「プログラミングで図を動かしてみよう」の設定を参考にしてください。

3-2 実施手順の詳細

基本的に、1 つの項目に対して以下の手順で進めます。

1. サンプルプログラムoooo.cs を開かせます。
- プログラムを見てもらいます。
- 長いプログラムではないので、極力、生徒が見終わるまで待ってください。
2. 簡単にどんなプログラムか(何が起こるプログラムか)を解説をする。
3. サンプルプログラムoooo.cs をコンパイルさせる。
- コマンドプロンプトで、csc oooo.cs
4. 出来た実行ファイルoooo.exe を実行して確かめる。
- コマンドプロンプトで、oooo.exe
5. プログラムの詳しい解説を行う。
6. プログラムの変更内容を示し、プログラムの変更・コンパイル・動作確認をさせる。
- コンパイルでエラーがでたら、コンパイルエラーを確認し、一緒に修正作業をする。
7. 課題を出して、各自、プログラムの変更・コンパイル・動作確認をさせる。

8. どんな変更を加えたか、発表してもらい、コメントをする。

3-2-1 最初のフォーム作成

まず、何もないフォームを表示させるプログラムから始めます。

1. サンプルプログラム 1.cs を開き、プログラムに目を通させてください。

サンプルプログラム "1.cs"

```
 1: using System;
 2: using System.Windows.Forms;
 3:
 4: class Hello: Form
 5: {
 6:   public static void Main()
 7:   {
 8:     Hello hw = new Hello();
 9:     Application.Run(hw);
10:   }
11: }
```

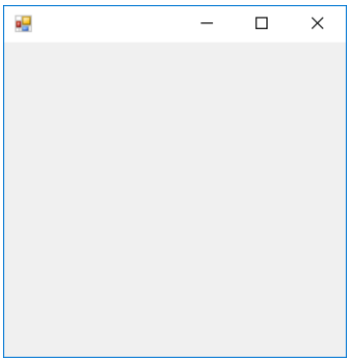
2. 「これはただ、空っぽの、何にも無いフォーム(ウィンドウ)を表示させるプログラムであること、これをベースにこれから色々といじってアプリケーションを作っていくこと」を説明してください。

3. このプログラムをコンパイルし、実行ファイル 1.exe が出来ることを確認させてください。

- コマンドプロンプトで、`csc 1.cs`
- 「コンパイル」については、簡単な解説で構いません。
  - 例えば、「人間が書いたプログラムを、コンピューターが分かる形に変換(翻訳)する作業」など。

4. 出来た実行ファイル 1.exe を実行して確かめる。

- コマンドプロンプトで、`1.exe`
- 起動時にフォーカスが当たっていないこともあるので、それを説明し、`Alt` キー+`Tab` キーで切り替えて、起動したフォームにフォーカスを当てるようにしてください。
- 起動時にコマンドプロンプトが立ち上がります。これは、デバッグ用のためのものです。
- フォームを確認したら、`Alt` キー+`Space` キー→`C` などでフォームを閉じさせてください。



1.exe

5. プログラムの詳しい解説を行います。

- 最初にキッチリ解説をしておくと、後々楽になると思いますが、生徒によっては、退屈してしまうかもしれません。
- 生徒の食いつき具合を見て、適度に切り上げ、後のプログラム解説で分散して解説するのもいいかもしれません。
- ここに記しているのは、説明の一例ですので、生徒のレベルに応じて内容・詳細を変えて、もちろん構いません。

- 1 行目　「このプログラムで System という部品集(便利機能の集まり)を使いますよ。」と言った意味、という程度の解説で構いません。本当はちょっと違うのですが、最初はこの程度の認識で良いかと思います。

– System は非常によく使う、これがないと何も出来ない位のものですので、ほとんどのプログラムで使用されます。

- C#では、命令文の最後に;(セミコロン)をつけます。
- 2 行目　今回は、フォーム、要するにウィンドウを使ったアプリケーションを作るので、これらを作るための System.Windows.Forms も使う、ということを書いておきます。
- 3 行目　4 行目からプログラムの本体を書くので、区切りのための空行です。
- 4 行目　ここからプログラムの本体が始まります。
- 「class Hello: Form は、"Hello という、フォーム(ウィンドウ)の作り方"を書きますよ。」といった程度の説明で良いかと思います。
- 枠線や、タイトルバーといった、どんなフォーム(ウィンドウ)にも必要な部品は、全て自分でプログラムするわけではありません。
- 先程の System.Windows.Forms にある Form というところ(クラス)に、フォーム(ウィンドウ)に基本的に必要なことが既に用意されていて、それを使います。という意味の: Form です。
- あとは、必要な部品の配置や、処理の仕方をプログラムするだけでよい。というわけです。
- ここは、命令文ではないので;(セミコロン)をつけていません。本当は違うのですが・・・。
- 5 行目　4 行目で、Hello という、フォーム(ウィンドウ)の作り方を書く、としていますが、それがどこ

からどこまでか、を示すために、ここの{と、11 行目の}で、挟んで表します。
  - 6 行目から 10 行目まで、この{と}の中身ですよ、ということを表すために、右に字下げ(インデント)するのが一般的です。
  - 今回のサンプルでは、1 回の字下げを「スペース 2 つ」としています。
- 6 行目　ここから、プログラムの流れを書いていきます。
  - public static void Main()は、今のところは、決まり文句、といった程度で良いかと思います。
- 7 行目　6 行目の、public static void Main()の中身を、ここの{と、10 行目の}で挟んで表します。
  - 8 行目と 9 行目を、この{と}の中身ですよ、ということを表すために、右に更に字下げ(インデント)しています。
- 8 行目　「Hello という、"フォームの作り方"に従って、フォームを作って、それを変数 hw に代入(=)せよ。」という命令文、程度の説明で。これも決まり文句という扱いでも構いません。ただ、変数名 hw は自由につけられる、ということは伝えてください。
  - "代入"については、お決まりの「箱に入れる」といった説明で良いと思います。

- =の前後にスペースが入っているのは、見やすく・読みやすくするためです。
- 9 行目　Application という、プログラムを実行・制御する機能の集まりから、Run という、プログラムを実行する機能(これを関数・メソッド)を呼び出して、Hello というフォームの作り方に従って作った hw を起動せよ。という意味です。といった説明で。もう少し簡略化してもいいかもしれません。
- 10 行目　}で、public static void Main()の中身を閉じます。
- 11 行目　}で、class Hello: Form の中身を閉じて、プログラム終わりです。

3-2-2 タイトルバーにアプリケーション名を表示する。

1.cs を閉じて、先程のフォームのタイトルバーに文字を表示させるプログラムに移ります。

1. サンプルプログラム 2.cs を開き、先程のプログラムの、9 行目に命令を 1 行追加しただけであることを伝えて、プログラムに目を通させてください。

サンプルプログラム "2.cs"

```
1: using System;
2: using System.Windows.Forms;
3:
4: class Hello: Form
```

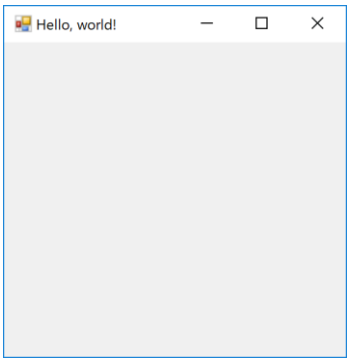
```
 5: {
 6:   public static void Main()
 7:   {
 8:     Hello hw = new Hello();
 9:     hw.Text = "Hello, world!";
10:     Application.Run(hw);
11:   }
12: }
```

2. 「9 行目に追加した命令で、タイトルバーに文字が表示される」事を確認していきます。

3. このプログラムをコンパイルし、実行ファイル 2.exe が出来ることを確認させてください。

4. 出来た実行ファイル 2.exe を実行して確かめる。

- 実行ファイルが出来たら、起動して、タイトルバー "Hello, world!"と出ていることを確認します。
- 利用しているスクリーンリーダーが、PC-Talker であれば、Ctrl+Alt+1 で読み上げます。
- 確認したら、Alt キー+Space キー→C などでフォームを閉じさせてください。



2.exe

5. プログラムの詳しい解説を行います。

- 追加した 9 行目　hw はフォームそのものを指しています。
- hw.Text は、hw の、hw が持っている、Text という要素・変数・パラメーターを示します。
- このように、C#では、.(ピリオド、今風に言えばドット)で、モノ(格好良くオブジェクト)の要素を示していきます。
- 日本語で「の」にあたる。「hw「の」Text という要素(C#ではプロパティ)」と言う意味。
- 9 行目は、「'hw'のプロパティ Text に文字列(文字の並び)Hello, world!を「代入」、格好良く言うと、セットする。」という意味になります。
- C#では、文字の並び、文字列は、"(ダブルクォーテーション)でくくります。
- フォーム hw のプロパティ Text は何か、というと、タイトルバーに表示する文字列を表すので、タイト

ルバーに Hello, world! と表示された、というわけです。

6. では、アプリケーションに名前をつけ、タイトルバーに表示されるようにします。

1. この、フォームアプリケーションの名前を考えてもらいます。
- 時間をかける必要はありませんが、愛着を持ってもらいたいので、できれば自分で考えてもらいたいところです。
- あとからいくらでも変更できる。ということを伝えて、気楽に考えてもらってください。
2. 2.cs を変更して、考えてもらった名前を、アプリケーションのタイトルバーに出るようにしてもらいます。
3. ファイルを保存して、コンパイルします。
4. エラーが出たら、エラーメッセージに従って、一緒に修正してください。
- この時、エラーメッセージの意味と対処をできるだけ説明してください。
- たとえば、; が必要です。と出れば、最後に;をつけ忘れていたね。など。
5. 実行ファイルが出来たら、起動して、タイトルバーに自分がつけた名前が出ていることを確認させてください。

7. 時間があれば、どんな名前をつけたか発表してもらい、コメントをする。

- 時間があれば、全員のものを確認し、感想を言ってあげてください。

3-2-3 ボタンを配置する。

2.cs を閉じて、フォームにボタンを配置するプログラムに移ります。

1. サンプルプログラム 3.cs を開き、先程のプログラムの、10 行目に 4 行追加したものであることを伝えて、プログラムに目を通させてください。

サンプルプログラム "3.cs"

```
 1: using System;
 2: using System.Windows.Forms;
 3:
 4: class Hello: Form
 5: {
 6:   public static void Main()
 7:   {
 8:     Hello hw = new Hello();
 9:      hw.Text = "アプリケーション名";
10:
11:     Button btn = new Button();
12:     btn.Parent = hw;
13:     btn.Text = "Push";
```

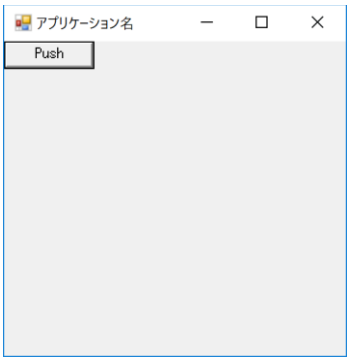
```
14:
15:         Application.Run(hw);
16:     }
17: }
```

2. 「追加した命令で、ボタンが配置される」事を確認していきます。

3. このプログラムをコンパイルし、実行ファイル 3.exe が出来ることを確認させてください。

4. 出来た実行ファイル 3.exe を実行して確かめる。

- 実行ファイルが出来たら、起動して、ボタンが配置されていることを確認します。
- Tab キーでフォーカスを移動させると、「Push」のボタンであることが読み上げられます。
- ボタンは、フォームの左上隅に配置されていることを伝えてください。
- 確認したら、Alt キー+Space キー→C などでフォームを閉じさせてください。



3.exe

5. プログラムの詳しい解説を行います。

- 10 行目と 14 行目の空行は、無くても構いませんが、その間の行がボタンに関するプログラムであることをわかりやすくするために入れています。
- 11 行目　基本的なボタンの作り方が、Button という名前(クラス)で用意されているので、この作り方でボタンを作り、btn に代入する(btn という名前をつける)。
- 12 行目　ボタン btn のプロパティ Parent は、貼り付け先を指定するので、フォーム hw をセットし、フォーム hw つまり、今作っているフォームに貼り付ける。
- 13 行目　ボタン btn のプロパティ Text は、ボタンに表示される文字列を示すので、文字列"Push"をセットする。
- 今回は、貼り付ける位置を指定していないので、左上隅に配置されました。

6. 再び、先程考えた名前が表示されるようにして、ボタンに表示される文字列を変えてもらいます。

1. 3.cs を変更して、再び、先程考えたアプリケーション名が表示されるようにしてもらいます。
2. ボタンに表示される文字列を考えてもらい、変更してもらいます。

3. ファイルを保存して、コンパイルします。
4. エラーが出たら、エラーメッセージに従って、一緒に修正してください。
5. 実行ファイルが出来たら、起動して、タイトルバーに自分がつけた名前、ボタンに変更した文字列が出ていることを確認させてください。

7. 時間があれば、全員のものを確認し、感想を言ってあげてください。

3-2-4 ボタンを押すとメッセージボックスが表示される。

3.cs を閉じて、ボタンを押すとメッセージボックスが表示されるプログラムに移ります。

1. サンプルプログラム 4.cs を開き、先程のプログラムの、6 行目に 4 行追加し、19 行目にも 1 行追加したものであることを伝えて、プログラムに目を通させてください。

ここから複雑になっていきますので、各自ついて行けているか確認しながら進んでください。

サンプルプログラム "4.cs"

```
1: using System;
2: using System.Windows.Forms;
3:
4: class Hello: Form
```

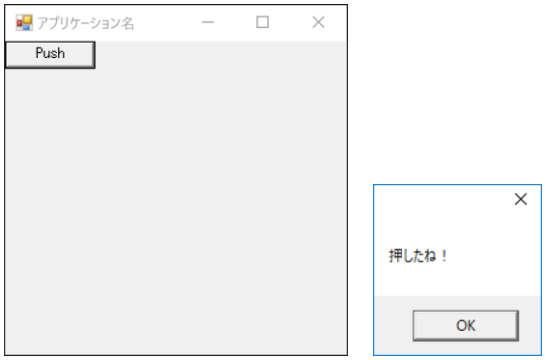
```
 5: {
 6:   static void btnOnClick(object send
er, EventArgs e)
 7:   {
 8:     MessageBox.Show("押したね！");
 9:   }
10:
11:   public static void Main()
12:   {
13:     Hello hw = new Hello();
14:      hw.Text = "アプリケーション名";
15:
16:     Button btn = new Button();
17:     btn.Parent = hw;
18:     btn.Text = "Push";
19:     btn.Click += new EventHandler(bt
nOnClick);
20:
21:     Application.Run(hw);
22:   }
23: }
```

2. 「追加した命令で、ボタンを押すとメッセージボックスがでる」事を確認していきます。

3. このプログラムをコンパイルし、実行ファイル 4.exe が出来ることを確認させてください。

4. 出来た実行ファイル 4.exe を実行して確かめる。

- 実行ファイルが出来たら、起動して、ボタンが配置されていることを確認します。
- ボタンを押すと、メッセージボックスが出ることと、メッセージボックスにあるメッセージを確認させてください。
- 確認したら、「OK ボタン」を押し、メッセージボックスを閉じて、Alt キー+Space キー→C などでフォームを閉じさせてください。



5. プログラムの詳しい解説を行います。

- 6 行目　ボタンを押したらどうするか、を記す**イベントハンドラ**と言われる関数(メソッド)を用意する必要があります。ここで、btnOnClick というイベントハンドラを定義するための記述です。

- イベントハンドラ名 btnOnClick は自由につけることが出来ますが、他の所は、決まり文句だと思ってください。
- 7 行目の{と 9 行目の}で、このイベントハンドラの中身をくくっています。
- 8 行目　ここで、あらかじめ用意されている MessageBox という、メッセージボックスを呼び出すための機能を使って、メッセージを表示させています。
- MessageBox の Show という関数(メソッド)に、表示するメッセージを与えて、メッセージボックスを表示させています。
- 10 行目は区切りを分かりやすくするための空行です。
- 19 行目　ボタン btn のプロパティ Click には、ボタンが押されたときの処理をセットします。
- 新しく動作を追加するために、先程作ったイベントハンドラ btnOnClick を追加しています。
- 追加なので、=ではなく、+=を使用していることに注意してください。
- この書き方も、決まり文句だと思ってください。

- これだけの作業で、ボタンを押すとメッセージが出るように出来ました。

6. 再び、先程考えた名前が表示されるようにして、ボタンに表示される文字列を変え、メッセージボックスのメッセージを変更してもらいます。

7. 時間があれば、全員のものを確認し、感想を言ってあげてください。

3-2-5 テキストボックスを追加する。

4.cs を閉じて、テキストボックスを追加するプログラムに移ります。

1. サンプルプログラム 5.cs を開き、先程のプログラムの、2 行目に 1 行追加し、16 行目から 5 行、23 行にも 1 行追加したものであることを伝えて、プログラムに目を通させてください。

ここから複雑になっていきますので、各自ついて行けているか確認しながら進んでください。

サンプルプログラム "5.cs"

```
1: using System;
2: using System.Drawing;
3: using System.Windows.Forms;
4:
5: class Hello: Form
6: {
7:   static void btnOnClick(object send
er, EventArgs e)
```

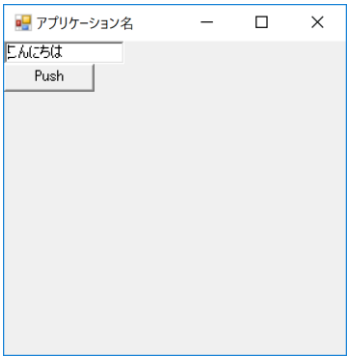
```
 8:   {
 9:     MessageBox.Show("押したね！");
10:   }
11:
12:   public static void Main()
13:   {
14:     Hello hw = new Hello();
15:      hw.Text = "アプリケーション名";
16:
17:     TextBox box= new TextBox();
18:     box.Parent = hw;
19:     box.Text = "こんにちは";
20:
21:     Button btn = new Button();
22:     btn.Parent = hw;
23:     btn.Location = new Point(0, box.
Height);
24:     btn.Text = "Push";
25:     btn.Click += new EventHandler(bt
nOnClick);
26:
27:     Application.Run(hw);
28:   }
29: }
```

2. 「追加した命令で、フォームに文字列が表示される」事を確認していきます。

3. このプログラムをコンパイルし、実行ファイル 5.exe が出来ることを確認させてください。

4. 出来た実行ファイル 5.exe を実行して確かめる。

- 実行ファイルが出来たら、起動して、テキストボックスに「こんにちは」と表示され、ボタンも配置されていることを確認します。
- 位置については、テキストボックスのすぐ下にボタンが表示されていることを伝えてください。
- テキストボックスの中は編集できることを確認させてください。
- 前回と同じく、ボタンを押すと、メッセージボックスが出ることと、メッセージボックスにあるメッセージを確認させてください。
- 確認したら、フォームを閉じさせてください。



5.exe

5. プログラムの詳しい解説を行います。

- 2 行目　ボタンの位置を変更する必要があるので、そのために System.Drawing を使うことを宣言します。

- 17 行目　テキストボックスを配置するために、box と言う名前でテキストボックスを作成します。
  - ボタンを作るときの書き方と同じです。
- 18 行目　ボタンの文字列をセットしたのと同様に、テキストボックス box の文字列をセットしています。

- 23 行目　そのままでは、テキストボックス box も、ボタン btn も、フォームの左上隅に配置されて重なってしまうので、ボタンの位置 Location をセットします。
  - new Point(0, box.Height)は、ボタンの左上隅の x 座標を 0 とし、y 座標は、テキストボックス box の高さだけ下にずらした所を指定します。
  - フォーム上の座標は、フォームの左上隅を(0, 0)とし、右に行けば x 座標は増加し、下に行けば y 座標が増加します。
  - 数学のグラフで出てくる座標とは上下が逆です。
  - テキストボックス box の高さは、box.Height で取得できます。
- これだけの作業で、テキストボックスを配置し、その下にボタンを配置することが出来ました。

6. 再び、先程考えた名前が表示されるようにして、ボタンに表示される文字列を変え、メッセージボックスのメッセージを変更し、テキストボックスの文字列を変更してもらいます。

7. 時間があれば、全員のものを確認し、感想を言ってあげてください。

3-2-6 発展させて、クイズゲームを作りました。

5.cs を閉じて、これまでの応用のクイズゲームのプログラムに移ります。

1. サンプルプログラム 6.cs を開き、プログラムに目を通させてください。

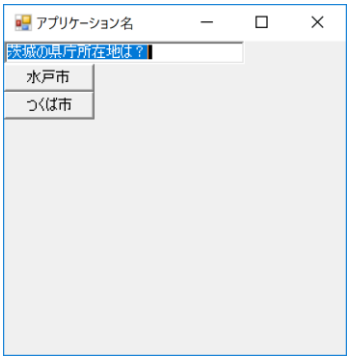
追加したプログラムで、何をしているか分かるところがあるか、あれば、何をしているか発表させてください。

サンプルプログラム "6.cs"

```
1: using System;
2: using System.Drawing;
3: using System.Windows.Forms;
4:
5: class Hello: Form
6: {
7:   static void btn1OnClick(object sender,
EventArgs e)
```

```
 8:   {
 9:     MessageBox.Show("正解！");
10:   }
11:
12:   static void btn2OnClick(object sender,
 EventArgs e)
13:   {
14:     MessageBox.Show("残念！");
15:   }
16:
17:   public static void Main()
18:   {
19:     Hello hw = new Hello();
20:     hw.Text = "アプリケーション名";
21:
22:     TextBox box= new TextBox();
23:     box.Parent = hw;
24:     box.Width = box.Width*2;
25:     box.Text = "茨城の県庁所在地は？";
26:
27:     Button btn1 = new Button();
28:     btn1.Parent = hw;
29:     btn1.Location = new Point(0, box.Hei
ght);
30:     btn1.Text = "水戸市";
31:     btn1.Click += new EventHandler(btn1O
nClick);
```

```
32:
33:     Button btn2 = new Button();
34:     btn2.Parent = hw;
35:     btn2.Location = new Point(0, box.Hei
ght + btn1.Height);
36:     btn2.Text = "つくば市";
37:     btn2.Click += new EventHandler(btn2O
nClick);
38:
39:     Application.Run(hw);
40:   }
41: }
```



6.exe

2. 時間があれば、このプログラムを変更して、クイズゲーム、更にオリジナルのアプリケーションを作ってみてください。

3-3 注意すべき点

- あまり用語など厳密にやる必要はありませんが、本当の意味・解釈は別にあるということは伝えておく必要があると思います。

- できれば、それを知りたい・勉強したい、と思わせるように持って行けるのが理想です。

3-4 到達目標

- プログラムをした、という経験だけで十分。
- 今回は、サンプルプログラムをいじった程度なので、フラストレーションをバネに、もっと自分でやってみたい。と思ってくれれば更によい。

4 生徒用資料

4-1 概要と目的、目標の説明

プログラミング言語 C#で、シンプルな GUI(Graphical User Interface)アプリケーションを作ることで、Windows アプリケーション開発の最初の一歩を踏み出します。 時間が限られていますので、難しいことは置いておいて、まずプログラミングの楽しさ、そして気軽に作れるものだということを感じて、今後、自分で色々と作ってみる入口としてください。

4-2 実施内容の説明

サンプルプログラムを見ながら、少し変更を加えて、アプリケーションを作っていきます。

4-3 当日ワークショップ内で使う資料

あらかじめ用意したサンプルプログラムのみです。 持って帰って、色々といじってみてください。