# Chapter8_リクエストにお答えします

リクエストがあったサンプルコードを公開する。

## 8-1 インターフェース

インターフェースを実装する場合、メソッドにoverrideキーワードを付ける必要がない。

抽象クラスの抽象メソッドを実装する場合、メソッドにoverrideキーワードを付ける必要がある。

EXAMPLE

例8-1 インターフェースの例

```
using System;
using System.Collections.Generic;

namespace Example2
{
	class Program : ASample, ISample
	{
		public string Name { get; set; }

		public Program(string name)
		{
			Name = name;
		}

		static void Main(string[] args)
		{
			IList<ISample> list = new List<ISample>();
			list.Add(new Program("インスタンス1"));
			list.Add(new Program("インスタンス2"));
			list.Add(new Program("インスタンス3"));

			foreach(ISample sample in list)
			{
				sample.Run();
				sample.Walk();
			}

			ASample aSample = new Program("抽象クラス実装");
			aSample.Talk();

		}

		// インターフェースの抽象メソッドを実装する際はoverride不要
		public void Run()
		{
			Console.WriteLine(Name + ":Runメソッドを実行した");
		}

		// インターフェースの抽象メソッドを実装する際はoverride不要
		public void Walk()
		{
			Console.WriteLine(Name + ":Walkメソッドを実行した");
		}

		// 抽象クラスの抽象メソッドを実装する際はoverride必要
		public override void Talk()
		{
			Console.WriteLine(Name + ":Talkメソッドを実行した");
		}
	}

	// インターフェース
	interface ISample
	{
		public abstract void Walk();

		public abstract void Run();
	}

	// 抽象クラス
	abstract class ASample
	{
		public abstract void Talk();
	}
}
```

RESULTS

例8-1の実行結果

```
インスタンス1:Runメソッドを実行した
インスタンス1:Walkメソッドを実行した
インスタンス2:Runメソッドを実行した
インスタンス2:Walkメソッドを実行した
インスタンス3:Runメソッドを実行した
インスタンス3:Walkメソッドを実行した
抽象クラス実装:Talkメソッドを実行した
```

## 8-2 複数のフォームを切り替える

プロジェクトからフォームを追加してもコードを書かなければフォームの切り替えはできない。

ここでは3つのフォームを用意し、切り替える方法を紹介する。

筆者はプロジェクト名を「MultiFormApp」で作成した。任意のプロジェクト名で良い。

### Step01 フォームを用意する

まずフォームを追加する。1つは既にあるはずなので、2つ追加すれば良い。

筆者の環境ではフォーム名をデフォルトの「Form2.cs」、「Form3.cs」とした。

#### ¶ フォームの追加方法

> ソリューションエクスプローラーのプロジェクトを右クリック > 追加 > 新しい項目 > 左側のメニューから「Windows Forms」を選択 > 「フォーム(Windowsフォーム)」を選択 > 適当なフォーム名を入力 > 追加

### Step02 フォームをデザインする

コントロールを配置し、デザインを行う。

#### ◆Form1

図8-1 Form1のデザイン

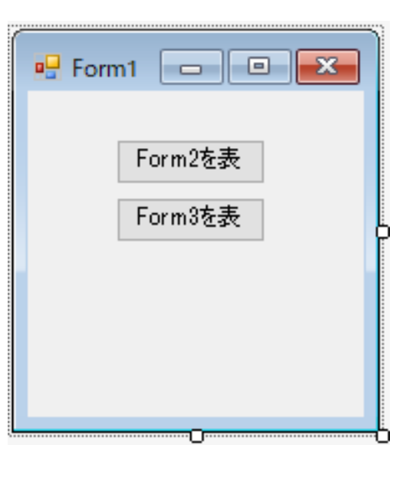

表8-1 Form1のコントロール一覧

| コントロール | Name | Text | 説明 |
|---|---|---|---|
| Form | Form1 | Form1 | プロジェクト作成時に自動的に生成されるフォーム。 |
| Button | button1 | Form2を表示 | クリックするとForm2を表示するためのボタン。 |
| Button | button2 | Form3を表示 | クリックするとForm3を表示するためのボタン。 |

#### ◆Form2

図8-2 Form2のデザイン

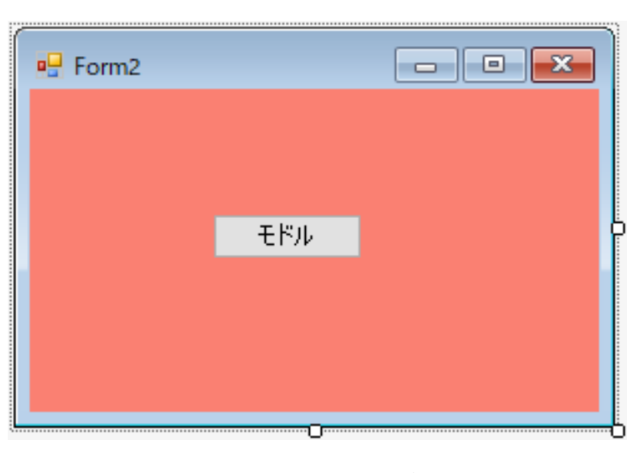

表8-2 Form2のコントロール一覧

| コントロール | Name | Text | 説明 |
|---|---|---|---|
| Form | Form2 | Form2 | フォーム追加時に自動的に生成されるフォーム。<br>※フォームを判別しやすいようにBackColorプロパティを変更済 |
| Button | button1 | モドル | クリックするとForm1を表示するためのボタン。 |

#### ◆Form3

図8-3 Form3のデザイン

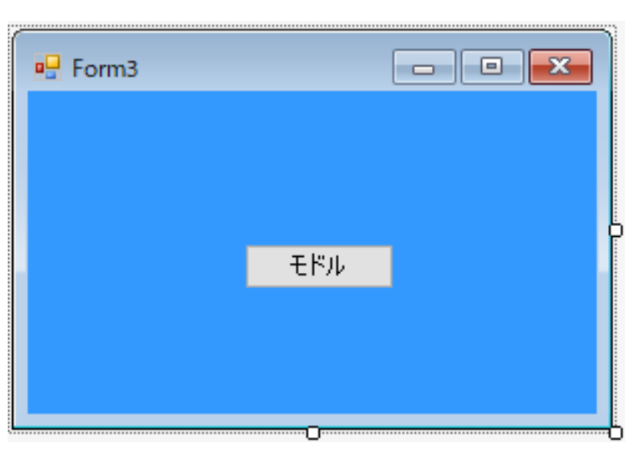

表8-2 Form2のコントロール一覧

| コントロール | Name | Text | 説明 |
|---|---|---|---|
| Form | Form3 | Form3 | フォーム追加時に自動的に生成されるフォーム。<br>※フォームを判別しやすいようにBackColorプロパティを変更済 |
| Button | button1 | モドル | クリックするとForm1を表示するためのボタン。 |

### Step03 イベントハンドラを用意する

イベントハンドラを用意し、イベント(ボタンクリック)に対応する。

#### ◆Form1

ボタン「button1」をダブルクリックし、イベントハンドラ「button1_Click」を用意。

ボタン「button2」をダブルクリックし、イベントハンドラ「button2_Click」を用意。

#### ◆Form2

ボタン「button1」をダブルクリックし、イベントハンドラ「button1_Click」を用意。

#### ◆Form3

ボタン「button1」をダブルクリックし、イベントハンドラ「button1_Click」を用意。

### Step04 イベントハンドラの処理内容を書く ＋α

イベントハンドラの処理内容を書くのだが、＋αで書かなければいけない点がある。

#### ◆Form1

以下の点を書く。

1. メンバ変数　自身以外のフォーム
2. コンストラクタ　Form2,Form3をインスタンス化してメンバ変数へ代入
3. イベントハンドラ「button1_Click」 Form1を非表示にする、Form2を表示する
4. イベントハンドラ「button2_Click」 Form1を非表示にする、Form3を表示する

コード8-1 Form1.cs

```
using System;
using System.Windows.Forms;

namespace MultiFormApp
{
	public partial class Form1 : Form
	{
		// 1. メンバ変数　自身以外のフォーム
		private Form form2, form3;

		// 2. コンストラクタ　Form2,Form3をインスタンス化してメンバ変数へ代入
		public Form1()
		{
			InitializeComponent();
			form2 = new Form2(this);
			form3 = new Form3(this);
		}

		// 3. イベントハンドラ「button1_Click」 Form1を非表示にする、Form2を表示する
		private void button1_Click(object sender, EventArgs e)
		{
			Hide();
			form2.Show();
		}

		// 4. イベントハンドラ「button2_Click」 Form1を非表示にする、Form3を表示する
		private void button2_Click(object sender, EventArgs e)
		{
			Hide();
			form3.Show();
		}
	}
}
```

#### 解説 Form1のしくみ

**1.** メンバ変数　自身以外のフォーム

メンバ変数にform2, form3を定義しているのは、フォームに配置したコントロール以外は(プロパティNameを利用した)変数として扱えないため。最初から存在しているフォームのForm1だけは例外で、変数として扱える。

**2.** コンストラクタ　**Form2,Form3**をインスタンス化してメンバ変数へ代入

プロジェクトから追加したクラスやフォームはインスタンス生成することで使うことができる。クラスForm1はプログラム実行時に自動でインスタンス生成される(Visual Studio自体がそのようにコードを自動生成する)ため、わざわざ自分でインスタンス生成する必要はない。

Form2およびForm3のインスタンス生成時、引数にForm1のインスタンスを渡している。これは、Form2およびForm3は初期状態ではForm1のインスタンスを持っていないため、引数として渡すことでForm1のインスタンスを使えるようにする試みである。

※クラスForm2とクラスForm3に引数ありコンストラクタを定義していないとエラーが出る。

**3.** イベントハンドラ「**button1_Click**」 **Form1**を非表示にする、**Form2**を表示する

フォームを非表示にするにはHideメソッドを使えば良い。フォームを表示するにはShowメソッドを使う。自分自身のフォームを非表示にしてから、別フォームを表示する制御にしている。別フォームを表示してから自身のフォームを非表示にする場合だと一瞬ではあるが、フォームが2つ表示されることになってしまう。

**4.** イベントハンドラ「**button2_Click**」 **Form1**を非表示にする、**Form3**を表示する

(3. と同じ内容なので説明を省略する)

#### ◆Form2

以下の点を書く。

1. メンバ変数　Form1のフォーム
2. コンストラクタ　引数をメンバ変数へ代入、Form2を非表示にする
3. イベントハンドラ「button1_Click」 Form2を非表示にする、Form1を表示する

コード8-2 Form2.cs

```
using System;
using System.Windows.Forms;

namespace MultiFormApp
{
	public partial class Form2 : Form
	{
		// 1. メンバ変数　Form1のフォーム
		private Form menuForm;

		// 2. コンストラクタ　引数をメンバ変数へ代入、Form2を非表示にする
		public Form2(Form form)
		{
			InitializeComponent();
			menuForm = form;
			Hide();
		}

		// 3. イベントハンドラ「button1_Click」 Form2を非表示にする、Form1を表示する
		private void button1_Click(object sender, EventArgs e)
		{
			Hide();
			menuForm.Show();
		}
	}
}
```

#### 解説 Form2のしくみ

**1.** メンバ変数　**Form1**のフォーム

Form1のインスタンス情報を持っておかないと、Form1を表示することができない。

**2.** コンストラクタ　引数をメンバ変数へ代入、**Form2**を非表示にする

Form1のインスタンス情報はコンストラクタの引数で受け取る。値を保持するためにメンバ変数menuFormに代入した後、フォーム(Form2)を非表示にしておく。初期状態でフォームが表示されていると、Form1とForm2が同時に表示されることになってしまう。なお、コンストラクタは初期状態では引数なしのコンストラクタのみがコードに存在する。コンストラクタの引数は自由に変更しても問題無い。

※Form1だけは例外で、プログラム実行時にForm1の引数なしコンストラクタが呼び出されるため、必ず引数なしのコンストラクタにしておくこと。

**3.** イベントハンドラ「**button1_Click**」 **Form2**を非表示にする、**Form1**を表示する

自分自身(Form2)を非表示にしてからForm1を表示する。

#### ◆Form3

以下の点を書く。

1. メンバ変数　Form1のフォーム
2. コンストラクタ　引数をメンバ変数へ代入、Form3を非表示にする
3. イベントハンドラ「button1_Click」 Form3を非表示にする、Form1を表示する

コード8-3 Form3.cs

```
using System;
using System.Windows.Forms;

namespace MultiFormApp
{
	public partial class Form3 : Form
	{
		// 1. メンバ変数　Form1のフォーム
		private Form menuForm;

		// 2. コンストラクタ　引数をメンバ変数へ代入、Form3を非表示にする
		public Form3(Form form)
		{
			InitializeComponent();
			menuForm = form;
			Hide();
		}

		// 3. イベントハンドラ「button1_Click」 Form3を非表示にする、Form1を表示する
		private void button1_Click(object sender, EventArgs e)
		{
			Hide();
			menuForm.Show();
		}
	}
}
```

#### 解説 Form3のしくみ

Form2のしくみと同様。

### Step05 動作確認

動作確認を行う。次のイメージ通りに動作すれば成功。

図8-4 複数のフォームを切り替えるアプリの動作