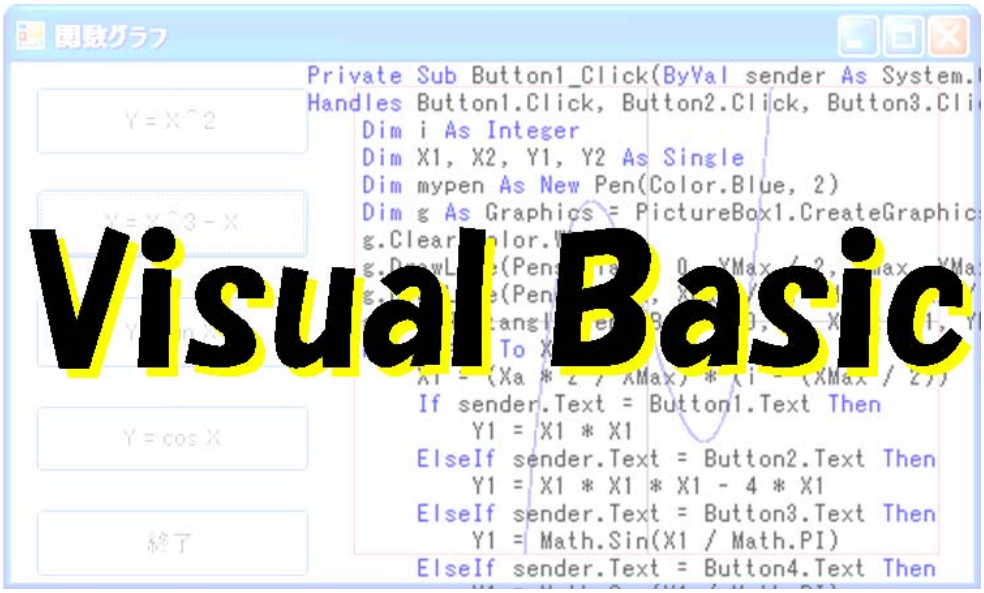
平成２０年度

VisualBasic研修講座



岩手県立総合教育センター

目　次

第１章　Visual Basicについて

１　Visual Basicとは

BASICとは「ビギナーのための言語」として1963年に開発された言語です。Windowsの時代になって、簡単にプログラムを作れるBASICソフトとしてMicrosoft社から販売されたのがVisual Basicです。デザイン画面でフォームを設計してコードを書くだけでWindows用のアプリケーションが作成できます。簡単な英単語で構成されているので、コードに記載されているだいたいの処理が分かる言語です。「C、C++、C#」のコードのように、中括弧{}がたくさんあり、「++」で１つ増加するなどの表記に比べるとずっと英文の表記に近いので読みやすいです。Excelなどのマクロ言語としても使われているVBAや、WindowsのスクリプトVisual Basic ScriptもVisual Basicの仲間です。

BASICは製品やバージョンによって仕様が異なっており、特に、Visual Basic 6.0（以後VB6)とそれ以降に出されたVisual Basic.NET、.NET 2003、2005は使える関数やコントロールの数や仕様が大きく違っています。

VisualBasic 2005は.NET Framework 2.0という巨大なライブラリーを利用できるようになったため、多機能なコントロールを活用して、数行のコードを書くだけでプログラムを完成させることが可能になっています。

日本のソフトウェア開発においてはVisualBaisiでの開発が18.9％と、COBOLの21.2％についで第２位となっており（ソフトウェア開発データ白書2006)、開発言語として広く利用されています。

２　Visual Basic 2005 Express Editionとは

Visual Basic 2005 Express EditionとはMicrosoft社が販売しているプログラム開発用ソフトウェアの１つです。Visual Basic 2005 Express Editionはプログラム入門者用の学習パッケージとして位置づけられており、次のような特徴があります。

- **無料である**（最大、最強の特徴）
- コンポーネントを貼り付けるだけでフォームが作成できる
- BASICを基本としているため、プログラムがわかりやすい
- 構造化、ブロック化の命令により、プログラムが読みやすい
- 入力支援機能（インテリセンスIntellisense)により、コードの記入が容易である
- 入門書や解説本がたくさん出されている
- ネット上の資料が豊富である
- **.NET Framework 2.0がインストールされている環境であれば、実行ファイルのコピーだけでアプリケーションが動く**
- VB6までの古いVisualBasicの書き方へも対応している

一方、欠点としては

- **.NET Framework 2.0がインストールされていないとアプリケーションは動かない**
- 起動に時間がかかる
- 機能がたくさんありすぎて、使い方に迷う

などがあげられます。

3　学習におけるコンピュータの利用

学習におけるコンピュータの利用は、一般的にCAI(Computer Assisted Instruction)といわれています。利用には次のような型があります。

(1)　ドリル学習型

すでに学習した内容を反復練習する型のソフトウェアです。学習者の記憶や技能を定着させるのに効果的です。

(2)　解説指導型

チュートリアル方式とよばれていたものです。学習者に対して個人授業を行うように、コンピュータから解説を与えて、質疑応答し、学習者の回答に応じた解説を提示します。学習内容の概念形成、知識・理解の定着、思考力の育成をねらいとする場合に用いられます。

(3)　問題解決型

学習者が問題解決をするために、既習の知識や情報と、コンピュータに用意されているデータを活用して学習を進めます。学習者に問題を明確に把握させること、問題解決方法に対応した多様なコースウェアを用意すること、解説指導型のソフトウェアを組み合わせることが必要といわれています。

(4)　シミュレーション型

人間の感覚ではとらえることが難しいことや、学習者にとって理解しにくい事象・現象をシミュレーションによって、わかりやすく提示するものです。

(5)　情報検索型

学習者は、情報を検索・抽出して思考を深めながら課題の解決を図る型です。学習者に課題を明確に把握させることが必要といわれています。

《コラム》Visual Basicの歴史

1991年 Visual Baisc 1.0	Windows版の他にMS-DOS版がありました
1993年 Visual Basic 2.0 for Wiondows	Windows3.1用のプログラム言語
1995年 Visual Basic 4.0	Windows95用のプログラム言語
1997年 Visual Basic 5.0	
1999年 Visual Basic 6.0	Webとデータベース関係を強化
2002年 Visual Basic .NET 2002	.NET Framework1.0を採用
2003年 Visual Basic .NET 2003	.NET Framework1.1を採用
2005年 Visual Basic 2005	.NET Framework2.0を採用
2008年 Visual Basic 2008　(最新版)	.NET Framework3.0を採用

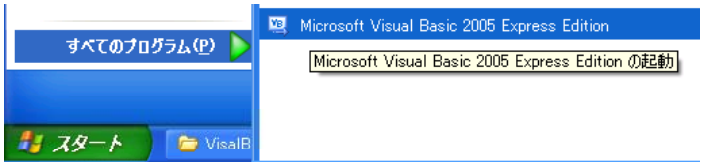
なおVBA(Visual Basic for Applications)は1994年Excel5.0から採用されています

第２章　VisualBasic2005 Express Editionの基本操作

１　起動と画面構成

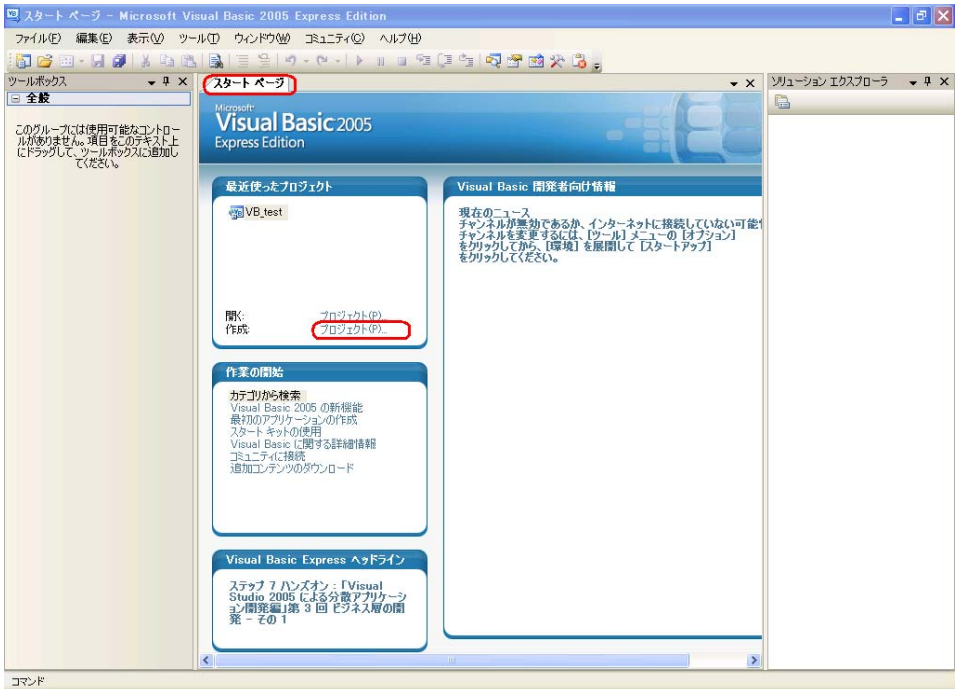
(1)　VisualBasic2005 Express Edition　の起動

①　「スタート」ボタンから「Microsoft Visual Basic 2005 Express Edition」を選択します。

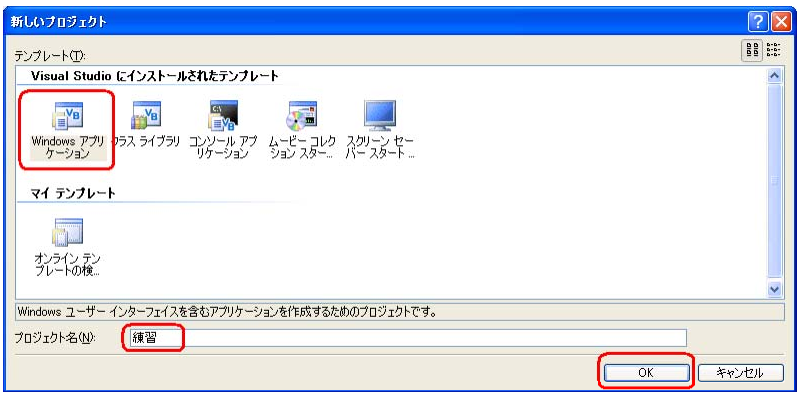


②　Microsoft Visual Basic 2005 Express Editionが起動します。(以後VB2005と記述)。起動までには若干の時間がかかります。

「スタートページ」から「作成　プロジェクト」を選択します。



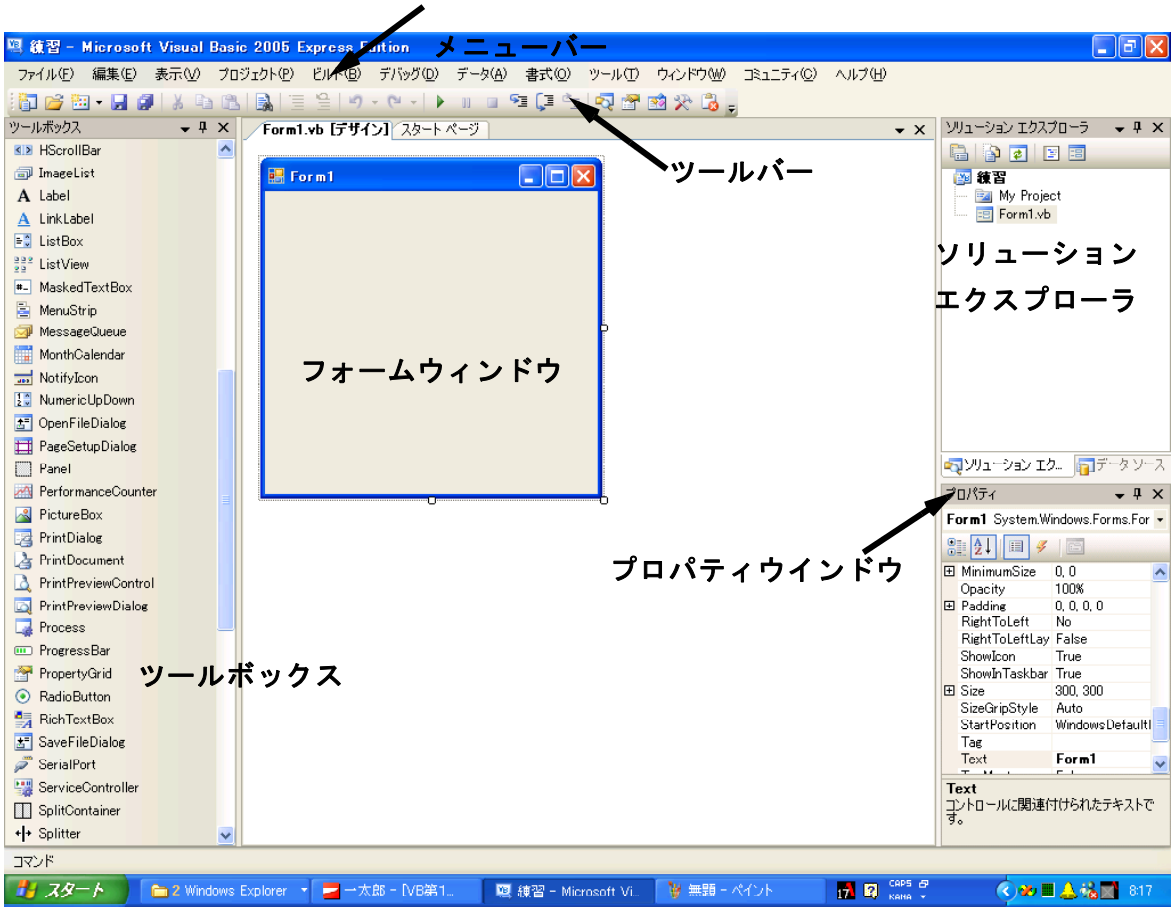
③　VB2005で普通のアプリケーションを作成する場合は、「新しいプロジェクト」のダイアログウィンドウで「Windowsアプリケーション」を選択します。「プロジェクト名」を練習と入力して「OK」をクリックします。



※　プロジェクト：VB2005のプログラム作成のための集合体のことを言います。プロジェクト全体を管理するファイル、フォームの情報を管理するファイルなど多数のファイルで構成されています。

(2)　画面構成

VB2005を起動すると、下図のような画面が表示されます。VB2005によるプログラムの開発はこの画面で、コントロール（オブジェクトの一種）の配置やプログラムコードの編集、実行、デバッグ（作動確認）を行います。



ツールボックス：

フォームに配置して利用できるコントロールが表示されています。

フォーム：

この画面が、作成するアプリケーションの基本画面になります。フォームや、配置されたコントロールをダブルクリックすることにより、コード入力の画面に切り替えることができます。

ソリューションエクスプローラ：

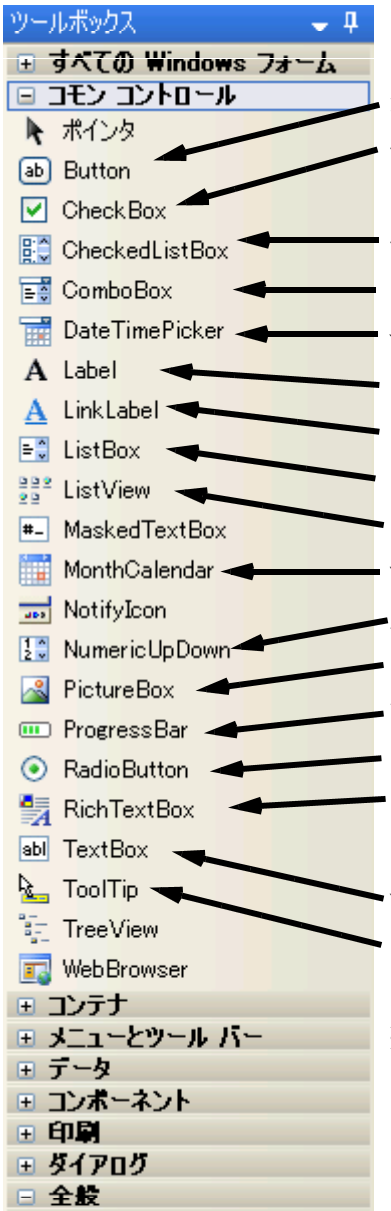
ファイルをダブルクリックすることで、それぞれの情報の画面を表示します。

プロパティウインドウ：

フォームや、コントロールのプロパティ（属性）の表示や設定ができます。

2　コントロールの種類とその機能

「＋」をクリックすることにより、表示するコントロールの種類を変えることができます。



主なコントロールの紹介をします。

ボタン：入力操作用のコントロール
チェックボックス：「使う／使わない」のような選択をするための入力操作用コントロール
チェックリストボックス：チェックボックスの組み合わせ
コンボボックス：プルダウンしてリストを表示する
デートタイムピッカー：日時をプルダウンして選択する
ラベル：文字列を表示する
リンクラベル：リンクのある文字列を表示する
リストボックス：複数のデータから選択を行う
リストビュー：画像などの選択を行う
マンスカレンダー：カレンダーを表示する
ヌメリックアップダウン：数値を増加、減少させる
ピクチャーボックス：画像を表示する
プログレスバー：処理の進行状況を表すときなどに用いる
ラジオボタン：複数から一つを選択するときに用いる
リッチテキストボックス：文字の色や大きさやフォントを変えることができる高機能なテキストボックス
テキストボックス：文字列を入力する場合に用いる
ツールチップ：操作のヒントを表示したい場合に用いる

※コントロールはアルファベット順に登録されています。

オブジェクト
　関連したデータ処理をひとつのまとまりにして、使えるようにしたものです。ツールバーにあるコントロール類もオブジェクトの１つです。

イベント
　プログラムに対して利用者が、ボタンを押したり文字を入力したりすると発生するのが「イベント」です。

3　プロパティの設定

フォームやコントロールの状態を表す値をプロパティといいます。プロパティの値を設定するためには、対象のコントロールを選択状態にして、プロパティウインドウで行います。プログラム上から設定をすることができます。

ここでは、「Form1」を選択した場合のプロパティウィンドウの内容について解説します。

ウィンドウスタイルの項目

コンロールボックス：Trueで表示
Falseで非表示

最大化ボタン：Trueで表示 

最小化ボタン：Trueで表示 

不透明度の割合：０％で透明

タスクバーへの表示：Trueで表示

デザインの項目

ネーム：コントロールの内容が分かるような名前をここでつけます

配置の項目

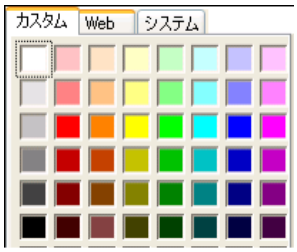
フォームの外形のサイズをここで設定します

＝横、縦

表示の項目

バックカラー：フォームの背景色を変えます

：で色選択のダイアログを表示します

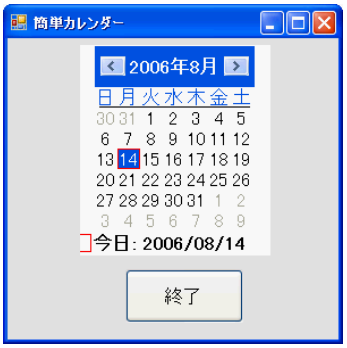


テキスト：タイトルバーに文字表示を行います

4　ソースコードとプログラムの実行

フォームにコントロールを貼り付けて、プロパティを設定することでアプリケーションのフォームができあがります。しかし、これだけでは形になっているものの、プログラムとしての機能はありません。そこで、コードの記述を行い、アプリケーションとしての機能をつけていきます。

ここでは、例題として、
簡単カレンダーを制作します。



(1)　フォームの設定

①　フォームをクリックして、プロパティウインドウのテキスト(Text)に「簡単カレンダー」と入力します。

②　フォームのバックカラーを好きな色に設定します。

(2)　マンスカレンダー(ManthCalender)の配置

①　ツールボックスのマンスカレンダー(ManthCalender)を選択します。

②　フォーム上でドラッグするとマンスカレンダーが配置されます。

※　ツールボックスのマンスカレンダーをダブルクリックしても配置されます。

③　ドラッグして位置を決めます。

④　プロパティウインドウでFontのSizeを選択して12に設定して字を大きくします。

(3)　ボタン(Button)の配置

①　ツールボックスのボタン(Button)を選択します。

②　フォーム上でドラッグするとボタンが配置されます。

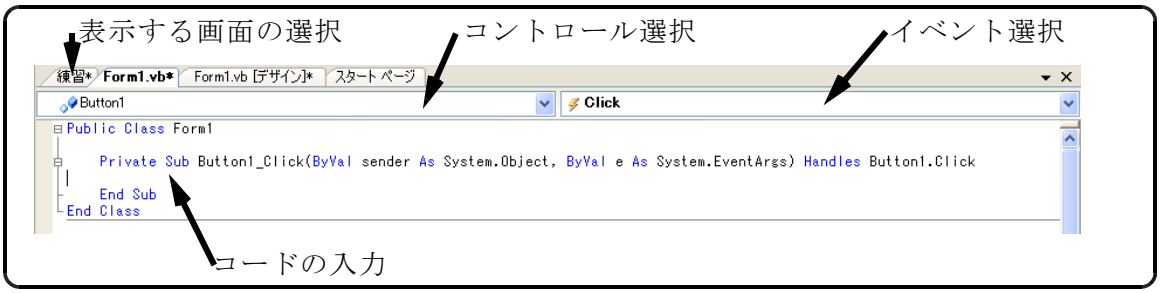
③　ドラッグして位置を決めます。

④　プロパティウインドウでTextに「終了」と入力します。

(4)　コードの入力

①　フォーム上のボタン（終了）をダブルクリックします。

②　コードウィンドウが表示されます。イベントプロシージャ（サブルーチン）が自動的に生成されます。



③　イベントプロシージャはオブジェクト名（コントロール名）とイベント名をアンダースコア（_）で合成したものがつけられています。

括弧内はイベントプロシージャの引数です。

括弧の後の「Handles Button1.Click」でオブジェクトとイベントを指定します。

半角英数で「End」と入力してください。

```
Private Sub Button1_Click(ByVal･･･ )Handles Button1.Click
  End
End Sub
```

入力にまちがいがある場合には、波下線が引かれます。

【ワンポイント】VB6とVB2005のプロシージャ名の扱いの違い

VB6ではプロシージャ名でオブジェクトとイベントを指定していました。VB2005では、オブジェクトとイベントの指定をHandlesの後ろに記載しているので、プロシージャ名を任意に変えることができます。

また、Handlesの後に、オブジェクトとイベントを複数記載することにより、複数のオブジェクトの処理を一つのプロシージャで処理することができます。

〈例〉Button1とButton2のイベントを１つのプロシージャで処理する場合

```
Private Sub Button(ByVal･･･ )Handles Button1.Click,Button2.Click
```

(5)　プログラムの実行　 

ツールバーにある「開始」ボタン  をクリックすることで、プロジェクトがビルド（コンパイルとリンク）されて実行されます。

「Ｆ５」キー、あるいは、メニューバーから「デバッグ」→「開始」でも可能です。

プログラムの終了は「停止」ボタン  をクリックしてください。または、実行しているアプリケーションを終了させてください。（  をクリックする）

(6)　ビルドとは

記述したプログラムをコンピュータが実行可能な形式に変換する作業をコンパイルといいます。コンパイルしたプログラムを、ライブラリ（アプリケーションを実行するための基本となるプログラム）と結びつけて、実行ファイル（拡張子が「exe」のファイル）を作成する作業をビルドといいます。

VB2005には次の２つのビルドがあります

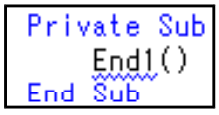
ビルド名	説明
デバッグビルド (Debug Build)	「開始」ボタン  でファイルが作成される ブレークポイントを設定できる bin¥Debug ディレクトリ内に実行ファイルが作成される
リリースビルド (Release Build)	メニューバーの「ビルド」「○○のビルド」で作成 最適化された実行ファイルを作成する bin¥Release ディレクトリ内に実行ファイルが作成される

作成したアプリケーションを配布する場合には、リリースビルドを行い、bin¥Release ディレクトリ内のファイルを使ってください。

(7)　デバッグのしかた

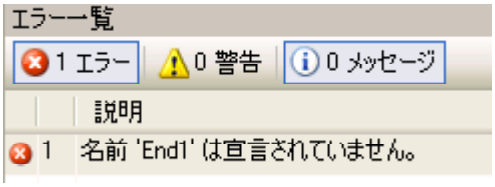
プログラムを実行させると、作動しない、作動しても思い通りの結果にならないことがあります。これを「バグ(Bug)がある」といいます。バグがある場合には、コードを見直して、バグを取り除くことが必要です。この作業を「デバッグ(Debug)」といいます。VB2005ではデバッグのための機能が充実しており、プログラム開発を快適にできるようになっています。

①　コード入力時のチェック機能



誤ったコードを入力すると、その部分に、波下線がつきます。

また、右図のように「エラー一覧」を表示します。説明の部分をダブルクリックすると、コード内のエラーの行を表示します



②　ビルド時のチェック機能

ツールバーにある「開始」ボタン  をクリックするとビルドされてプログラムが実行されます。このとき、エラーがあると次の図のように表示されます。

「いいえ」を選択して、プログラムを修正します。

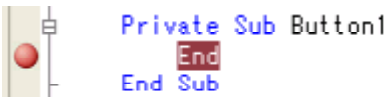
ビルド

ビルドとは、作成したプログラムを実行できる形式に変換することです。VB2005でビルドを実行すると、中間言語とよばれる言語に変換します。他の言語ではコンパイルとも言われます。

③　ブレークポイントの設定

ブレークポイントを設定すると、プログラムを途中で中断させることができます。中断した状態で、変数の確認、ワンステップ実行などにより、プログラムの誤りの訂正を行うことができます。

ブレークポイントの設定は、コードウィンドウの左側の灰色の部分をクリックすると設定することができます。赤丸がブレークポイントの印です。



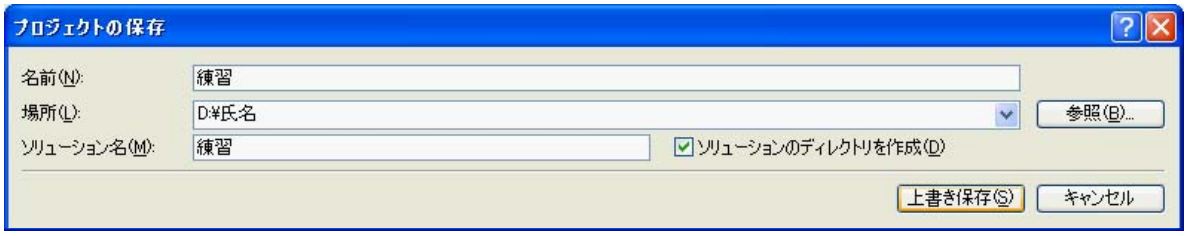
【ワンポイント】

VB2005では中断、プログラム修正した後の再開ができるようになりました。

(VB.NET2002,2003では不可、VB6では可能)

５　プロジェクトの保存

保存する場合には、ツールバーの「すべてを保存」ボタン　  　をクリックします。

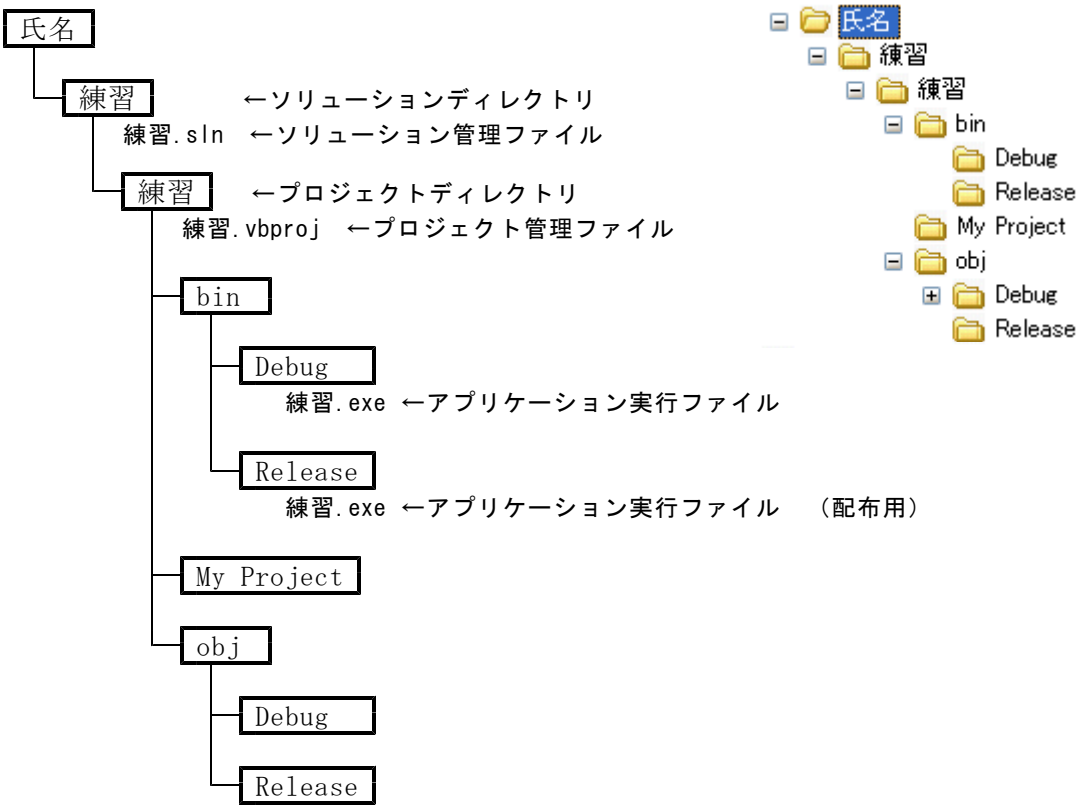


「プロジェクトの保存」ダイアログが表示されます。

「場所」に「D:¥氏名」と入力、または、参照から選択します。

「上書き保存」をクリックします。

VB2005のプログラムファイルのフォルダ構成は次のようになっています



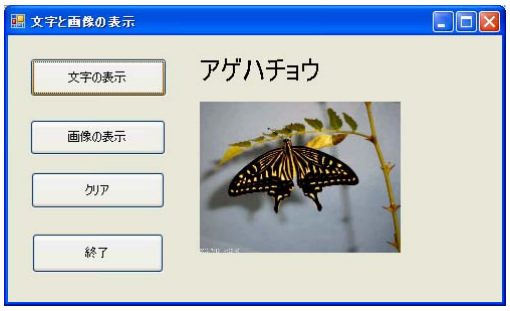
《コラム》.NET Framework2.0のインストール

VB2005で作成したアプリケーションを他のコンピュータで実行するためには、作成したファイル（例えば、練習.exe）をそのコンピュータにコピーしてダブルクリックで実行できます。もし、エラーが出る場合には、.NET Framework2.0をインストールする必要があります。

.NET Framework3.0、3.5 をインストールしても良いです。

第３章　教材作りのためのプログラミング基礎

１　文字と画像を表示させる

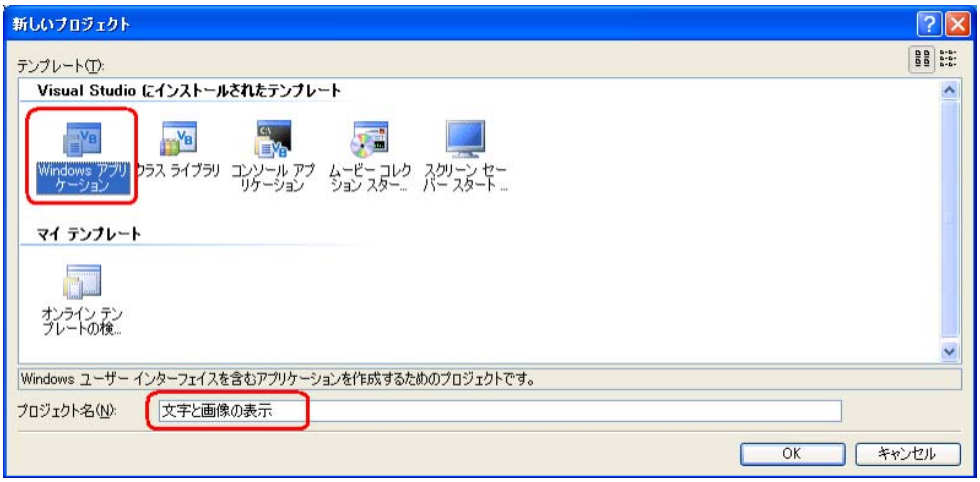


(1)　文字と画像の表示（同じウィンドウで表示）

文字と画像を表示するアプリケーションを作成します。ボタンをクリックすると、文字や画像を表示、消去ができます。

①　スタートページで「作成　プロジェクト」をクリックして、「新しいプロジェクト」のダイアログウィンドウを表示させます。

「Windowsアプリケーション」を選択し、プロジェクト名に「文字と画像の表示」と入力して、「ＯＫ」をクリックします。



②　デザイン画面でフォームを選択し、プロパティウインドウで次の値に設定します。

```
Size 500,300
Text 文字と画像の表示
```

③　ツールボックスからボタン(Button)を選択して４つボタンを配置します。

プロパティウィンドウで次のように設定します

```
Button1 Text 文字の表示
Button2 Text 画像の表示
Button3 Text クリア
Button4 Text 終了
```

④　ツールボックスからラベル(Label)を選択して配置し、プロパティウィンドウで次のように設定します

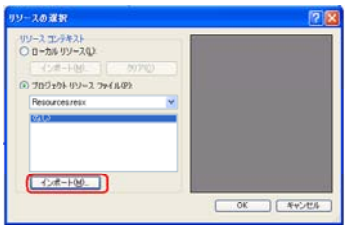
```
Label1 Text アゲハチョウ
```

⑤　ツールボックスからピクチャーボックス(PictureBox)を選択して配置し、プロパティウィンドウで次のように設定します

```
Size　200,150
```

Image の右側のボタンを押します

※ビデオやデジカメの画像は
横：縦=4:3 です。



リソースの選択のダイアログボックスが表示されます。「インポート」をクリックします。

ファイル選択のダイアログボックスから、「アゲハチョウ.jpg」を選択します。

リソースの選択のダイアログボックスの「ＯＫ」をクリックします。

さらに、PictureBox1 のプロパティウィンドウの動作の項目で、SizeMode を「StretchImage」にします。

アゲハチョウの画像が、ピクチャーボックスの大きさに合わせて表示されます。

⑥　フォームをダブルクリックして次のコードを記入します。

```
Label1.Visible = False
PictureBox1.Visible = False
```

【ワンポイント】
Ctrl ＋ スペースキーを押すと、コードの入力が容易になります

⑦　Button1「文字の表示」をダブルクリックして次のコードを入力します。

```
Label1.Visible = True
```

⑧　Button2「画像の表示」をダブルクリックして次のコードを入力します。

```
PictureBox1.Visible = True
```

⑨　Button3「クリア」をダブルクリックして次のコードを入力します。

```
Label1.Visible = False
PictureBox1.Visible = False
```

⑩　Button4「終了」をダブルクリックして次のコードを入力します。

```
Me.Close()
```

⑪　コードを入力し終えたら、開始ボタン を押して作動を確認しましょう。

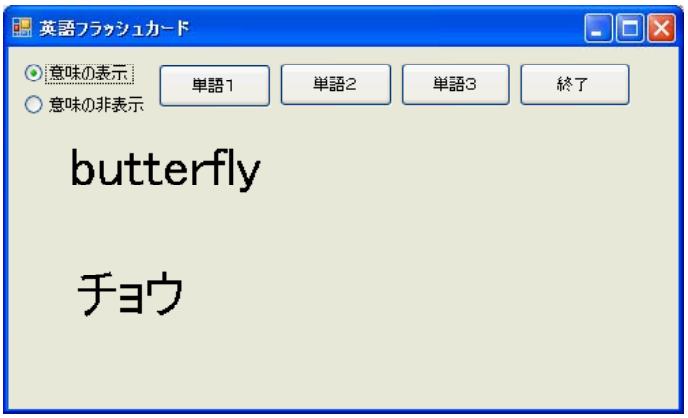
⑫　すべてを保存 をクリックして、ソリューションを保存しましょう。

※　発展問題

Private Sub Form1_Load 内でフォームやボタンのプロパティを設定してみましょう。

(2)　英語フラッシュカードの制作

ボタンをクリックすると単語を表示します。意味を表示するかどうかはラジオボタンで選択できます。



①　スタートページで「作成　プロジェクト」をクリックして、「新しいプロジェクト」のダイアログウィンドウを表示させます。

「Windowsアプリケーション」を選択し、プロジェクト名に「英語フラッシュカード」と入力して、「ＯＫ」をクリックします。

②　デザイン画面でフォームを選択し、プロパティウインドウで次の値に設定します。

```
Size 500,300
Text 英語フラッシュカード
```

③　ツールボックスからラジオボタン(RadioButton)を２つ配置します。

RadioButton1 Text を「意味の表示」にします

RadioButton2 Text を「意味の非表示」にします

④　ボタン(Button)を選択して横に４つボタンを配置します。

Button1 Text を「単語１」にします

Button2 Text を「単語２」にします

Button3 Text を「単語３」にします

Button4 Text を「終了」にします

⑤　ラベル(Label)を選択して２つ配置します。

Font Size を30にします。

⑥　フォームをダブルクリックして次のコードを記入します。

```
Label1.Text = ""
Label2.Text = ""
RadioButton1.Checked = True
```

⑦　RadioButton1「意味の表示」をダブルクリックして次のコードを入力します。

```
Label2.Visible = True
```

⑦　RadioButton2「意味の非表示」をダブルクリックして次のコードを入力します。

```
Label2.Visible = False
```

⑧　Button1「単語１」をダブルクリックして次のコードを入力します。

```
Label1.Text = "butterfly"
Label2.Text = "チョウ"
```

⑨　Button2「単語２」をダブルクリックして次のコードを入力します。

```
Label1.Text = "beetle"
Label2.Text = "カブトムシ"
```

⑩ Button3「単語３」をダブルクリックして次のコードを入力します。

```
Label1.Text = "dragonfly"
Label2.Text = "トンボ"
```

⑪ Button4「終了」をダブルクリックして次のコードを入力します。

```
Me.Close()
```

⑫ コードを入力し終えたら、開始ボタン を押して作動を確認しましょう。

⑬ すべてを保存 をクリックして、ソリューションを保存しましょう。

```
Public Class Form1
    Private Sub Form1_Load(ByVal sender As System.Object, _
      ByVal e As System.EventArgs) Handles MyBase.Load
        Label1.Text = ""
        Label2.Text = ""
        RadioButton1.Checked = True
    End Sub

    Private Sub RadioButton1_CheckedChanged(ByVal sender As System.Object, _
      ByVal e As System.EventArgs) Handles RadioButton1.CheckedChanged
        Label2.Visible = True
    End Sub

    Private Sub RadioButton2_CheckedChanged(ByVal sender As System.Object, _
      ByVal e As System.EventArgs) Handles RadioButton2.CheckedChanged
        Label2.Visible = False
    End Sub

    Private Sub Button1_Click(ByVal sender As System.Object, _
      ByVal e As System.EventArgs) Handles Button1.Click
        Label1.Text = "butterfly"
        Label2.Text = "チョウ"
    End Sub

    Private Sub Button2_Click(ByVal sender As System.Object, _
      ByVal e As System.EventArgs) Handles Button2.Click
        Label1.Text = "beetle"
        Label2.Text = "カブトムシ"
    End Sub

    Private Sub Button3_Click(ByVal sender As System.Object, _
     ByVal e As System.EventArgs) Handles Button3.Click
        Label1.Text = "dragonfly"
        Label2.Text = "トンボ"
    End Sub

    Private Sub Button4_Click(ByVal sender As System.Object, _
      ByVal e As System.EventArgs) Handles Button4.Click
        Me.Close()
    End Sub
End Class
```

【ワンポイント】
「 _」(ｽﾍﾟｰｽ ｱﾝﾀﾞｰｽｺｱ)で次の行に続けることができます

(3)　ピクチャーカードの制作

ボタンをクリックするとbinフォルダにある画像ファイルを読み込んで画像を表示します。

①　スタートページで「作成　プロジェクト」をクリックして、「新しいプロジェクト」のダイアログウィンドウを表示させます。

「Windowsアプリケーション」を選択し、プロジェクト名に「ピクチャーカード」と入力して、「ＯＫ」をクリックします。

②　デザイン画面でフォームを選択し、プロパティウインドウで次の値に設定します。

Size 500,300
Text ピクチャーカード

③　すべてを保存  をクリックして、ソリューションを保存します。

④　マイコンピュータからＤドライブの氏名フォルダの中の「ピクチャーカード」の「binフォルダ」の「Debugフォルダ」に次の画像をコピーします。

(D:¥氏名¥ピクチャーカード¥bin¥Debug)

釜渕の滝.jpg
雫石のヒマワリ.jpg
中尊寺.jpg
岩手山.jpg
石割桜.jpg
羅須地人協会.jpg
小岩井農場.jpg
白鳥.jpg

③　ボタン(Button)を選択して縦に４つボタンを配置します。

Button1 Text を「画像１」にします
Button2 Text を「画像２」にします
Button3 Text を「画像３」にします
Button4 Text を「終了」にします

④　ツールボックスからピクチャーボックス(PictureBox)を選択して配置し、プロパティウィンドウで次のように設定します

Size　280,210
SizeMode を「StretchImage」にします

※画像なので横：縦= 4:3

⑤　ラベル(Label)を選択して１つ配置します。

⑥　次のコードを記入します。

```
Public Class Form1
  Private Sub Form1_Load(ByVal sender As System.Object, _
     ByVal e As System.EventArgs) Handles MyBase.Load
       Label1.Text = ""
  End Sub

  Private Sub Button1_Click(ByVal sender As System.Object, _
    ByVal e As System.EventArgs) Handles Button1.Click
      PictureBox1.Image = Image.FromFile("釜渕の滝.jpg")
      Label1.Text = "釜渕の滝.jpg"
  End Sub

  Private Sub Button2_Click(ByVal sender As System.Object, _
    ByVal e As System.EventArgs) Handles Button2.Click
      PictureBox1.Image = Image.FromFile("雫石のヒマワリ.jpg")
      Label1.Text = "雫石のヒマワリ.jpg"
  End Sub

  Private Sub Button3_Click(ByVal sender As System.Object, _
    ByVal e As System.EventArgs) Handles Button3.Click
      PictureBox1.Image = Image.FromFile("中尊寺.jpg")
      Label1.Text = "中尊寺.jpg"
  End Sub

  Private Sub Button4_Click(ByVal sender As System.Object, _
    ByVal e As System.EventArgs) Handles Button4.Click
      Me.Close()
  End Sub
End Class
```

⑦　コードを入力し終えたら、開始ボタン  を押して作動を確認しましょう。

⑧　すべてを保存  をクリックして、ソリューションを保存しましょう。

※　画像ファイルは、表示したい画像ファイルに変更してください。

※　画像が表示されない場合には、画像ファイルが
「D:¥氏名¥ピクチャーカード¥bin¥Debug」に入っているか確認してください。

※　プログラムを他のコンピュータで実行する場合には、画像ファイルも同じフォルダに入れてください。

（4） 超簡単ブラウザ

URLを入力して「移動」ボタンをクリックするとWebページを表示します。コードを書き込むのはたった**７行だけ**です。これだけで、ブラウザとして動きます。当然、ハイパーリンクで他のページを表示しますし、ページを移動した場合にはそのURLも表示します。ウィンドウの大きさを変えることもできます。

① スタートページで「作成　プロジェクト」をクリックして、「新しいプロジェクト」のダイアログウィンドウを表示させます。

「Windowsアプリケーション」を選択し、プロジェクト名に「Myブラウザ」と入力して、「ＯＫ」をクリックします。

② デザイン画面でフォームを選択し、プロパティウインドウで次の値に設定します。

```
Size 500,500
```

③ ツールボックスからラベル(Label)を１つ配置します。

```
Label1 Text　を「URL」と入力します。
```

④ テキストボックス(TextBox)を選択して１つ配置します。

⑤ ボタン(Button)を選択して１配置します。

```
Button1 Text を「表示する」にします
```

⑥ 「ツールボックス」で「すべてのWindowsフォーム」を選択して、一番下のウェブブラウザ(WebBrowser) 

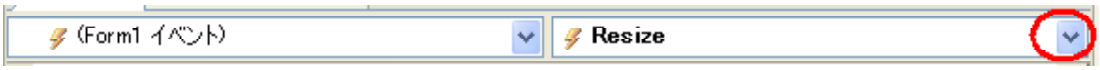
を選択して１つ配置します。プロパティウィンドウで次のように設定します。

```
Location 0, 30
Sise    490, 434
```

⑦ フォームをダブルクリックして「Private Sub Form1_Load」の中に次のコードを記入します。

```
TextBox1.Text = "http://www1.iwate-ed.jp/"
WebBrowser1.Navigate(TextBox1.Text)
```

⑧ フォームのイベントから「⌄」をクリックして「Resize」を選択します



「Private Sub Form1_Resize」の中に次のコードを記入します。

```
WebBrowser1.Width = Me.Width - 10
WebBrowser1.Height = Me.Height - 65
```

⑨ Button1「表示する」をダブルクリックして次のコードを入力します。

```
WebBrowser1.Navigate(TextBox1.Text)
```

⑩ ウェブブラウザ(WebBrowser)をダブルクリックして
「Private Sub WebBrowser1_Navigated」に次のコードを入力します。

```
Me.Text = WebBrowser1.Document.Title.ToString() + " - MyBrowser"
Me.TextBox1.Text = WebBrowser1.Url.ToString()
```

⑫ コードを入力し終えたら、開始ボタン  を押して作動を確認しましょう。

⑬ すべてを保存  をクリックして、ソリューションを保存しましょう。

```
Public Class Form1

 Private Sub Form1_Load(ByVal sender As System.Object, ByVal e As _
  System.EventArgs) Handles MyBase.Load
     TextBox1.Text = "http://www1.iwate-ed.jp/"          ホームページURL
     WebBrowser1.Navigate(TextBox1.Text)
 End Sub

 Private Sub Form1_Resize(ByVal sender As Object, ByVal e As _
  System.EventArgs) Handles Me.Resize
     WebBrowser1.Width = Me.Width - 10                   ﾌｫｰﾑの大きさに合わ
     WebBrowser1.Height = Me.Height - 65                 せて大きさを変更
 End Sub

 Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As _
  System.EventArgs) Handles Button1.Click
     WebBrowser1.Navigate(TextBox1.Text)                 TextBox1のURLを
 End Sub                                                 表示させます

 Private Sub WebBrowser1_Navigated(ByVal sender As Object, ByVal e As _
  System.Windows.Forms.WebBrowserNavigatedEventArgs) _
  Handles WebBrowser1.Navigated
     Me.Text = WebBrowser1.Document.Title.ToString() + " - MyBrowser"
     TextBox1.Text = WebBrowser1.Url.ToString()          ページを移動したとき
 End Sub                                                 のURLを表示

End Class                                                .ToString()は文字列へ
                                                         の変換です
```

２　変数と演算子

(1)　変数の基礎知識

変数はデータを入れる容器です。変数をプログラムの中で使うためには、まず最初に「変数の宣言」をします。変数が使える有効範囲をスコープといいます。

①　変数の宣言

Dim　　：　普通の変数の宣言に用います
　　プロシージャ内で宣言した場合には、そのプロシージャ内で実行している間だけ有効です（ローカル変数)。

Public　：　アプリケーションのどこからでも使用可です。

Static　：　プロシージャ内で使用します。プロシージャが終了しても変数の値を保持します（静的変数)。

const　：　値を参照することだけが可能な変数を定義することができます。

②　変数の宣言方法

例：　**Dim 変数 As 型**

　　Public 変数 As 型

(2)　変数名について

・変数名は英数または_（アンダスコア）であること

・漢字、ひらがなの変数名をつけることもできます（普通はつけませんが・・・）

・変数名にVB2005のコマンド名、オブジェクト名は使用できない。

・英字の大文字と小文字は区別しない（Cなど他の言語は区別するのが普通です)

(3)　代表的なデータ型について

型名	内容	ｻｲｽﾞ	値の範囲
Boolean	論理型	2	True（真）またはFalse（偽）
Integer	整数型	4	-2147483648 ～214748367の整数
Single	単精度 浮動小数点型	4	負の値 約-3.4×10(38乗)～-1.4×10(-45乗) 正の値 約 4.9×10(-45乗)～1.8×10(38乗)
Double	長精度 浮動小数点型	4	負の値 約-1.8×10(308乗)～-4×10(-324乗) 正の値 約 4.9×10(-324乗)～1.8×10(308乗)
String	文字列型	不定	最大20億個

(4)　変数への値の代入について

変数「Hensu」に数値の「1」を代入する場合には、次のように記入します。

```
Dim Hensu As Integer
Hensu = 1
```

または、次のように記入することにより、変数の宣言と代入ができます。

```
Dim Hensu As Integer =1
```

(5)　型変換

VB2005では、データの型を持っているため、型が異なると計算ができません。例えば、テキストボックスに入力された文字を数値にするためには型変換を行います。

```
Dim Suuti As Single
Dim Mozi As String
   Suuti = Convert.ToSingle(Mozi)
```

逆に、数値を文字列に変換する場合には、

```
   Mozi = Convert.ToString(Suuti)
```

(5) 演算子

下表の上位が優先となります。優先順位を変える、または明確にしたい場合には括弧（）を用いてください。

〈算術演算子〉

^	べき乗
*	乗算
/	除算
¥	整数の除算
Mod	整数の除算の余り
+	加算
-	減算
&	文字列の連結（+でも連結できます）

〈関係演算子〉

=	等しい	
<>	等しくない	not(A=B)
<	小さい、未満	
<=	以下（小さいか等しい）	
>	大きい	
>=	以上（大きいか等しい）	

(6) 関数

```
Int(x)      その数を超えない最大の整数   Int(1.5)は1、Int(-1.5)は-2
Fix(x)      その数の整数部分             Int(1.5)は1、Int(-1.5)は-1
Rnd()       0～1までの乱数の発生
Math.Round(x)  四捨五入
Math.Abs(x)    絶対値
Math.Sin(x)    サイン、xの単位はラジアン
   Sin45°の値をAに代入するためには次のように記入します。
  A = Math.Sin(45/180*Math.PI)
Math.Cos(x)    コサイン、xの単位はラジアン
Math.Tan(x)    タンジェント、xの単位はラジアン
Math.Sqr(x)    平方根
```

(7) 配列

配列は、同じ性質を持った値を効率的に管理するためのデータ構造です。変数の集合体ともいえます。配列の各要素に書き込みを行うためには、インデックス（番号、添字）で指定します。インデックスが一つのものを一次元配列、二つのものを二次元配列といいます。配列も宣言を行ってから使います。

① 整数型の一次元配列宣言の例

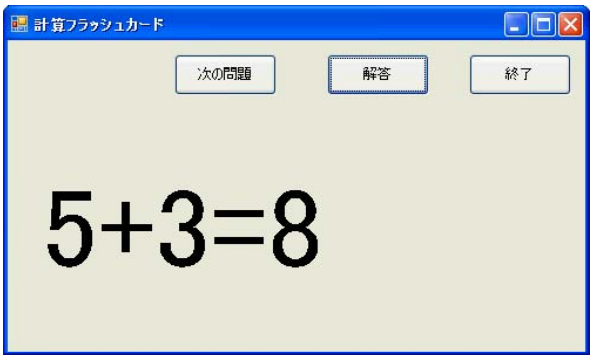
```
Dim Hairetu(10) As Integer
```

② 文字列の２次元配列宣言の例

```
Dim Mozi(10,2) As String
```

(8)　計算フラッシュカードの制作

小学校１年生用のたし算の計算フラッシュカードを作成します。少し改変することにより「ひき算」「かけ算」「わり算」に変更することができます。



①　スタートページで「作成　プロジェクト」をクリックして、「新しいプロジェクト」のダイアログウィンドウを表示させます。

「Windowsアプリケーション」を選択し、プロジェクト名に「計算フラッシュカード」と入力して、「ＯＫ」をクリックします。

②　デザイン画面でフォームを選択し、プロパティウインドウで次の値に設定します。

```
Size 500,300
Text 計算フラッシュカード
```

③　ツールボックスからボタン(Button)を選択して３つボタンを配置します。

プロパティウィンドウで次のように設定します

```
Button1 Text 次の問題
Button2 Text 解答
Button3 Text 終了
```

④　ツールボックスからラベル(Label)を選択して配置し、プロパティウィンドウで

次のように設定します

```
Font Size 72
```

⑤　コードを記入します

```
Public Class Form1

    Public Suuti1 As Single                    } 共通の変数はここで宣言します
    Public Suuti2 As Single                    }

    Private Sub Form1_Load(ByVal sender As System.Object, _
      ByVal e As System.EventArgs) Handles MyBase.Load
        Label1.Text = ""
    End Sub

    Private Sub Button1_Click(ByVal sender As System.Object, _
      ByVal e As System.EventArgs) Handles Button1.Click
```

```
        Suuti1 = Int(Rnd() * 9 + 1)
        Suuti2 = Int(Rnd() * 9 + 1)
        Label1.Text = Suuti1.ToString + "+" + Suuti2.ToString + "="
    End Sub

    Private Sub Button2_Click(ByVal sender As System.Object, _
      ByVal e As System.EventArgs) Handles Button2.Click
        Label1.Text = Suuti1.ToString + "+" + Suuti2.ToString + "=" _
                    + (Suuti1 + Suuti2).ToString
    End Sub

    Private Sub Button3_Click(ByVal sender As System.Object, _
      ByVal e As System.EventArgs) Handles Button3.Click
        Me.Close()
    End Sub

End Class
```

Rnd()で乱数を発生させます
Intで整数部分のみ取り出します
.ToStringで数値を文字列に変換します

※　数値を文字列に変換するには

数値.ToString
Convert.ToString(数値)
Str(数値)

このように複数の方法があります
.ToStringは.NETからの新しい表記です
Str関数を用いるのはVB6の古い表現です
どれを用いても良いです。

⑥　コードを入力し終えたら、開始ボタン  を押して作動を確認しましょう。

⑦　すべてを保存  をクリックして、ソリューションを保存しましょう。

※　発展問題

- 文字を大きくしましょう。
- 文字の色を変えてみましょう。
- ２桁の数の計算に変えてみましょう。
- 「かけ算」のカードを作ってみましょう。
- 「ひき算」「わり算」のカードを作ってみましょう。

３　流れ制御文

流れ文制御には、条件判断、繰り返しなどがあります。ここでは代表的なステートメント（命令）について基本の説明をします。

(1)　If文

ある条件が成立すれば「処理１」を実行し、成立しないときには「処理２」を実行させたい場合には、IF文を使います。

例：Ａの値が10以上ならＡから10を引き、そうでなければＡに２を加える。

プログラム例

```
If A >= 10 Then              Ａの値が10以上なら
   A = A - 10                条件が成立した場合の処理
Else
   A = A + 2                 条件が不成立の場合の処理
End If
```

条件式の例

Not(A >= 10)	Ａが10未満なら
(A >= 10) And (B >= 20)	Ａが10以上、かつ、Ｂが20以上なら
(A >= 10) Or (B >= 20)	Ａが10以上、または、Ｂが20以上なら

And	Ｂが真	Ｂが偽
Ａが真	真	偽
Ａが偽	偽	偽

Or	Ｂが真	Ｂが偽
Ａが真	真	真
Ａが偽	真	偽

※　Andの代わりに「AndAlso」、Orの代わりに「OrElse」を使うと速度が向上します

(2)　Select Case文

Select Caseは、条件式の値にしたがって、対応するブロックを実行する分岐文です。多分岐するときにはIf文よりすっきりした記述ができます。

```
Select Case テスト式
    Case 式１
        文１
    Case 式２
        文２

    Case Else
        文ｎ
End Select
```

・テスト式には数式、または文字式
・Case 式 には一致させたい式を書く
・どの式とも一致しない場合にはCase Elseが実行される
・Case Elseが不要なら省略できる
・Case 式の記述例

Case 1	テスト式が1なら
Case 1 To 6	1～6なら
Case 1,3,5	1,3,5なら
Case "AB"	文字列が"AB"なら
Case "AB","CD"	文字列が"AB"か"CD"なら

(3)　For　～Next文

For文は指定回数のループを行います。回数をループ変数で管理します。

```
For ループ変数 = 初期値　To　最終値　Step 増分
  文
Next ループ変数
```

※　Nextの後ろのループ変数は省略できます。

※　ループを途中で終了する場合には「Exit For」を用います。
「If 条件 Then Exit For」でFor文から抜け出すことができます。

(4)　Do～Loop文

Do文は条件が真である間、または、条件が真になるまで、処理を繰り返します。

① 前判定、間ループ
- 条件式が真の間、ループ処理する
- ループ処理の前に条件式を判定する

```
Do While 条件式
    文
Loop
```

② 前判定、までループ
- 条件式が真になるまで、ループ処理する
- ループ処理の前に条件式を判定する

```
Do Until 条件式
    文
Loop
```

③ 後判定、間ループ
- 条件式が真の間、ループ処理する
- ループ処理の後に条件式を判定する

```
Do
    文
Loop While 条件式
```

④ 後判定、までループ
- 条件式が真になるまで、ループ処理する
- ループ処理の後に条件式を判定する

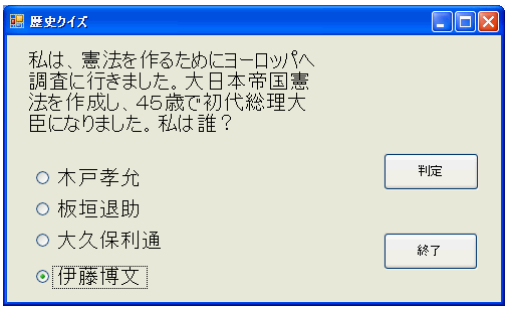
```
Do
    文
Loop Until 条件式
```

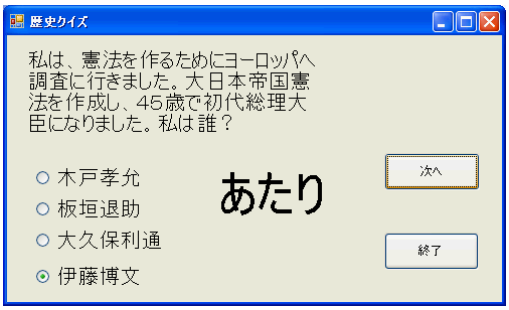
※　ループから脱出するためには次の文を用いる。
If 条件 Then Exit Do

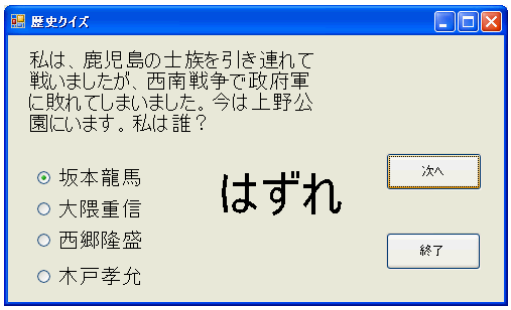
(5)　歴史クイズ

歴史の４択クイズを作成します。ボタンの表示を「判定」と「次へ」に切り替えて、If文で判断しています。

選択肢は乱数で並べ替えをして、正答と誤答の位置を毎回変えるようにしています。







①　スタートページで「作成　プロジェクト」をクリックして、「新しいプロジェクト」のダイアログウィンドウを表示させます。

「Windowsアプリケーション」を選択し、プロジェクト名に「歴史クイズ」と入力して、「ＯＫ」をクリックします。

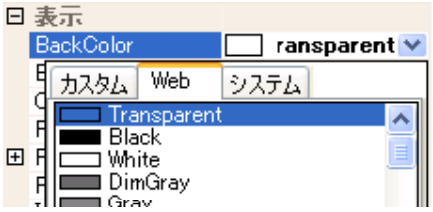
②　デザイン画面でフォームを選択し、プロパティウインドウで次の値に設定します。

```
Size 500,300
Text 歴史クイズ
```

④　ツールボックスからラベル(Label)を２つ選択して配置し、プロパティウィンドウで次のように設定します

```
Label1
  AutoSize   False
  Size       300,70
  Font Size 16
Label2
  Font Size 36
  BackColor   Transparent
```

【ワンポイント】
ラベルのバックカラーはWebのタグのTransparentで透明にできます



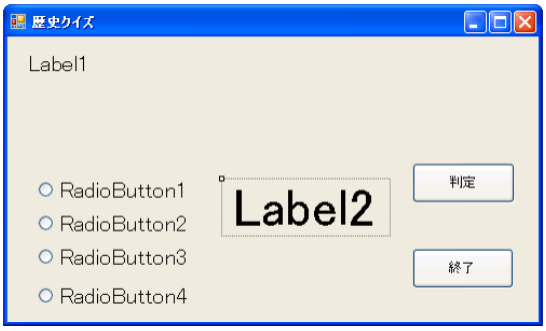
⑤　ツールボックスからラジオボタン(RadioButton)を選択して４つ配置します。

４つのラジオボタンのフォントの大きさを16にします

```
Font Size 16
```

⑥　ツールボックスからボタン(Button)を選択して２つボタンを配置します。
プロパティウィンドウで次のように設定します

```
Button1 Text 判定
Button2 Text 終了
```



⑦　コードを記入します

問題文は「歴史クイズ問題文.txt」のファイルに記載していますので、そちらを開いてコピー&ペーストしてください。

```
Public Class Form1

   Public Mondai(10, 4) As String
   Public Syutudai1(4) As Single
   Public Syutudai2(4) As Single
   Public Bangou As Integer
   Public Tokuten As Integer

   Private Sub Form1_Load(ByVal sender As System.Object, _
     ByVal e As System.EventArgs) Handles MyBase.Load
       Randomize()
       Label1.Text = "わたしの名前を当ててください" + vbCrLf + "出題をクリック"
       Label2.Visible = False
       RadioButton1.Text = ""
       RadioButton2.Text = ""
       RadioButton3.Text = ""
       RadioButton4.Text = ""
       Button1.Text = "出題"
       Bangou = 0
       Mondai(1, 0) = "私は、憲法を作るためにヨーロッパへ調査に行きました。大日本帝国憲法を作成し、４５歳で初代総理大臣になりました。私は誰？"
       Mondai(1, 1) = "伊藤博文"
       Mondai(1, 2) = "木戸孝允"
       Mondai(1, 3) = "板垣退助"
       Mondai(1, 4) = "大久保利通"
       Mondai(2, 0) = "私は、鹿児島の士族を引き連れて戦いましたが、西南戦争で政府軍に敗れてしまいました。今は上野公園にいます。私は誰？"
       Mondai(2, 1) = "西郷隆盛"
       Mondai(2, 2) = "坂本龍馬"
       Mondai(2, 3) = "大隈重信"
```

共通の変数の定義
問題文と正答誤答用
選択肢順番変更用
選択肢順番変更用
問題の順番
得点用

問題文
正答
誤答
誤答
誤答

問題文
正答
誤答
誤答

```
    Mondai(2, 4) = "木戸孝允"
    Mondai(3, 0) = "私は海援隊を結成しました。私は誰？"
    Mondai(3, 1) = "坂本龍馬"
    Mondai(3, 2) = "西郷隆盛"
    Mondai(3, 3) = "勝海舟"
    Mondai(3, 4) = "伊藤博文"
End Sub

Private Sub Button1_Click(ByVal sender As System.Object, _
  ByVal e As System.EventArgs) Handles Button1.Click
    Dim Hyouzi As String
    Dim s1, s2 As Single
    Dim i, k As Integer
    Hyouzi = Button1.Text
    If (Hyouzi = "出題") Or (Hyouzi = "次へ") Then
        Label2.Visible = False
        Bangou = Bangou + 1
        If Bangou >= 4 Then
            Bangou = 1
        End If
        Label1.Text = Mondai(Bangou, 0)
        RadioButton1.Text = Mondai(Bangou, 1)
        RadioButton2.Text = Mondai(Bangou, 2)
        RadioButton3.Text = Mondai(Bangou, 3)
        RadioButton4.Text = Mondai(Bangou, 4)
        Button1.Text = "判定"
    Else
        Label2.Text = "はずれ"
        If RadioButton1.Checked = True And _
          RadioButton1.Text = Mondai(Bangou, 1) Then
            Label2.Text = "あたり"
        End If
        If RadioButton2.Checked = True And _
          RadioButton2.Text = Mondai(Bangou, 1) Then
            Label2.Text = "あたり"
        End If
        If RadioButton3.Checked = True And _
          RadioButton3.Text = Mondai(Bangou, 1) Then
            Label2.Text = "あたり"
        End If
```

誤答
問題文
正答
誤答
誤答
誤答

ボタン1のText表示で分岐させる
あたりの表示を消す
次の問題にする
最後の問題の場合には最初の問題に戻す
問題文の表示
ラジオボタン1に表示
ラジオボタン2に表示
ラジオボタン3に表示
ラジオボタン4に表示

ボタン１の表示変更
ﾗｼﾞｵﾎﾞﾀﾝ1であたり
ﾗｼﾞｵﾎﾞﾀﾝ2であたり
ﾗｼﾞｵﾎﾞﾀﾝ3であたり

```
            If RadioButton4.Checked = True And _
              RadioButton4.Text = Mondai(Bangou, 1) Then
                Label2.Text = "あたり"                    ﾗｼﾞｵﾎﾞﾀﾝ4であたり
            End If
            Label2.Visible = True                         あたりorはずれの表示
            Button1.Text = "次へ"                         ボタン１の表示変更
        End If
    End Sub

    Private Sub Button2_Click(ByVal sender As System.Object, _
      ByVal e As System.EventArgs) Handles Button2.Click
        Me.Close()                                        ｱﾌﾟﾘｹｰｼｮﾝの終了
    End Sub

End Class
```

⑧　コードを入力し終えたら、開始ボタン を押して作動を確認しましょう。

⑨　すべてを保存 　をクリックして、ソリューションを保存しましょう。

※　発展問題

・　回答選択肢の位置をランダムで表示させます。

Private Sub Button1_Click　の「問題文の表示」の部分を次のように書き換えます。

```
        Label1.Text = Mondai(Bangou, 0)
        For i = 1 To 4
            Syutudai1(i) = Rnd()                  乱数を記憶
            Syutudai2(i) = i                      番号を記憶
        Next
        For i = 1 To 4
            For k = 1 To 4 - i
                If Syutudai1(k) > Syutudai1(k + 1) Then   乱数の大きさを
                    s1 = Syutudai1(k + 1)                 比較して，k+1
                    Syutudai1(k + 1) = Syutudai1(k)       よりk が大きい
                    Syutudai1(k) = s1                     場合には入れ替え
                    s2 = Syutudai2(k + 1)                 をする。
                    Syutudai2(k + 1) = Syutudai2(k)
                    Syutudai2(k) = s2
                End If
            Next
```

```
        Next
        RadioButton1.Text = Mondai(Bangou, Syutudai2(1))
        RadioButton2.Text = Mondai(Bangou, Syutudai2(2))
        RadioButton3.Text = Mondai(Bangou, Syutudai2(3))
        RadioButton4.Text = Mondai(Bangou, Syutudai2(4))
        Button1.Text = "判定"
```

- フォームの色、文字の色、大きさを変えてみましょう。
- 問題数を増やしましょう。
- クイズで出題順を変える

4　テキストファイルの読み込み、保存

ファイル処理には次の３種類があります。

- シーケンシャルアクセス：テキストファイルを先頭から順番に読み込む
- ランダムアクセス　　：レコード単位に、任意の位置から読み込む
- バイナリアクセス　　：バイナリデータを扱う

ここでは基本となる、テキストファイルの読み込み、保存について説明します。

(1)　VB2005のファイルの処理手段

VB2005でファイル処理をする手段としては、次の３つの方法があります。

① FileOpen関数などを使用、昔からのVBのファイル処理方法
② FileStreamとFileクラスなどを使用、.NET System.IOを用いる方法
③ My.Computer.FileSystemを用いる方法

ここでは、新しい方法である③の方法を中心に説明をします。

(2)　ファイルの一括読み込み、一括保存

テキストボックスにテキストファイルの内容を読み込みたい場合には、次のようにします。

```
TextBox1.Text = My.Computer.FileSystem.ReadAllText("Test.txt",System.Text.Encoding.Default)
```

テキストボックスの内容をファイルに保存したい場合には、次のようにします。

```
                                    ↓ﾌｧｲﾙ名      ↓保存ﾃｷｽﾄ    ↓追記ﾓｰﾄﾞ
My.Computer.FileSystem.WriteAllText("Test.txt", TextBox1.Text, False,System.Text.Encoding.Default)
```

【ワンポイント】
読み込む文字コードを指定する必要があります。
シフトJISを指定する場合　「System.Text.Encoding.Default」を記入します。
指定しないときには、シフトJISにはなりません。文字化けします。

(3)　テキストファイルを１行ずつ読み込む、保存する

１行ずつ読み込む場合には、はじめに、ファイルをオープンして、ファイルが終わりでない場合には読み込み、最後にファイルをクローズします。

サンプルファイル

```
Dim Fr As System.IO.TextReader                                  ﾃｷｽﾄﾘｰﾀﾞｰを定義
Dim Tx As String                                                文字変数を定義
Fr = My.Computer.FileSystem.OpenTextFileReader _                ﾌｧｲﾙ名とｴﾝｺｰﾄﾞを
     ("Test.txt", System.Text.Encoding.Default)                 定義してﾌｧｲﾙｵｰﾌﾟﾝ
Tx = Fr.ReadLine                                                １行読み込み
Do While Tx IsNot Nothing                                       ﾌｧｲﾙの終わり判断
    TextBox1.AppendText(Tx + vbCrLf)                            ﾃｷｽﾄﾎﾞｯｸｽに追加
    Tx = Fr.ReadLine                                            １行読み込み
Loop                                                            ループ
Fr.Close()                                                      ﾌｧｲﾙｸﾛｰｽﾞ
```

１行ずつ書き込む場合も、同様です。

サンプルファイル

```
Dim Fw As System.IO.TextWriter
Fw = My.Computer.FileSystem.OpenTextFileWriter _
     ("Test.txt", False, System.Text.Encoding.Default)
Fw.WriteLine("１行目です")
Fw.Write("２行目です、")
Fw.Write("２行目に続きます")
Fw.Close()
```

WriteLineは１行書き込み
Writeは後に続けて書き込みができます

保存したテキストファイルは右図のようになります。

```
1行目です
2行目です、2行目に続きます
```

(4)　ファイルダイアログを使う

オープンファイルダイアログ(OpenFileDialog)を用いると、読み込むファイルの選択を容易にすることができます。オープンファイルダイアログはツールボックスの「ダイアログ」の項目の中にあります。フォームの上にドラッグすると、フォーム上ではなく、コンポーネントトレイ内に入ります。

フィルター(Filter)プロパティで表示するファイルを設定します。

```
OpneFileDialog1.Filter = "txtﾌｧｲﾙ(*.txt)|*.txt|全てのﾌｧｲﾙ|*.*"
```

「説明文字列｜ファイルタイプ｜次の説明｜次のファイルタイプ」と記入します。

※　「｜」は「Shift + ¥」で表示されます。

ファイルダイアログを表示させるときには、

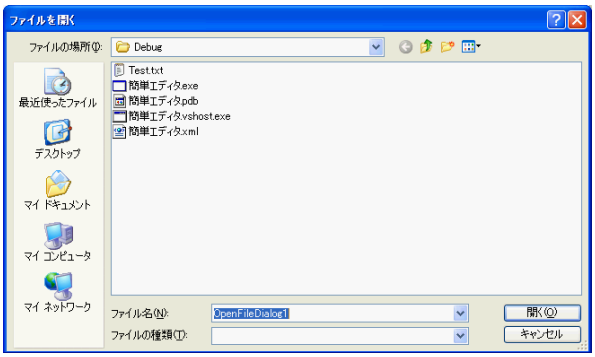
```
OpenFileDialog1.ShowDialog()
SaveFileDialog1.ShowDialog()
```

ファイルが選択されたときには、次の結果を返します。

```
DialogResult.OK          正しくファイルが選択された
DialogResult.Cansel      キャンセルまたは×が選択された
```

正しくファイルが選択された場合、次のようにしてファイル名を取得できます。

```
Dim Fn As String
Fn = OpenFileDialog1.FileName
Fn = SaveFileDialog1.FileName
```



※　数行でファイル入出力のダイアログを操作できます。

(5)　簡単エディタ

テキストファイルを読み込んで、テキストボックスに表示し、変更や保存ができる簡単エディタを作成してみましょう。

①　スタートページで「作成　プロジェクト」をクリックして、「新しいプロジェクト」のダイアログウィンドウを表示させます。

「Windowsアプリケーション」を選択し、プロジェクト名に「簡単エディタ」と入力して、「ＯＫ」をクリックします。

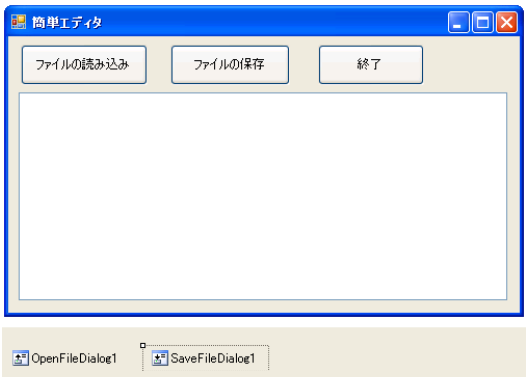
②　デザイン画面でフォームを選択し、プロパティウインドウで次の値に設定します。

```
Size 500,300
Text 簡単エディタ
```

③　ツールボックスからボタン(Button)を選択して３つボタンを配置します。

プロパティウィンドウで次のように設定します

```
Button1 Text ファイルの読み込み
Button2 Text ファイルの保存
Button3 Text 終了
```



④　ツールボックスからテキストボックス(TextBox)を選択して１つ配置します。

プロパティウィンドウで次のように設定します

```
動作　Multiline　True
配置　Size　　470,200
```

⑤　オープンファイルダイアログ(OpenFileDialog)を配置します。

⑥　セーブファイルダイアログ(SaveFileDialog)を配置します。

⑦　コードを記入します

```
Public Class Form1


    Private Sub Button1_Click(ByVal sender As System.Object, _
      ByVal e As System.EventArgs) Handles Button1.Click
        Dim Fn As String
        If OpenFileDialog1.ShowDialog = Windows.Forms.DialogResult.OK Then
            Fn = OpenFileDialog1.FileName
            TextBox1.Text = My.Computer.FileSystem.ReadAllText _
                            (Fn, System.Text.Encoding.Default)
```

```
        End If
    End Sub

    Private Sub Button2_Click(ByVal sender As System.Object, _
      ByVal e As System.EventArgs) Handles Button2.Click
        Dim Fn As String
        If SaveFileDialog1.ShowDialog = Windows.Forms.DialogResult.OK Then
            Fn = SaveFileDialog1.FileName
            My.Computer.FileSystem.WriteAllText _
            (Fn, TextBox1.Text, False, System.Text.Encoding.Default)
        End If
    End Sub

    Private Sub Button3_Click(ByVal sender As System.Object, _
      ByVal e As System.EventArgs) Handles Button3.Click
        Me.Close()
    End Sub

End Class
```

⑧　コードを入力し終えたら、開始ボタン  を押して作動を確認しましょう。

⑨　すべてを保存  をクリックして、ソリューションを保存しましょう。

※　発展問題

・ファイルダイアログにフィルタをつけてみましょう。

```
OpneFileDialog1.Filter = "txtﾌｧｲﾙ(*.txt)|*.txt|全てのﾌｧｲﾙ|*.*"
SaveFileDialog1.Filter = "txtﾌｧｲﾙ(*.txt)|*.txt|全てのﾌｧｲﾙ|*.*"
```

５　タイマーコントロール

タイマーコントロール(Timer)をフォーム上にドラッグすると、コンポーネントトレイ内に表示されます。タイマーコントロールの代表的なプロパティは次の２つです。

Interval　　イベント発生の間隔を設定、単位はミリ秒(1000分の１秒)

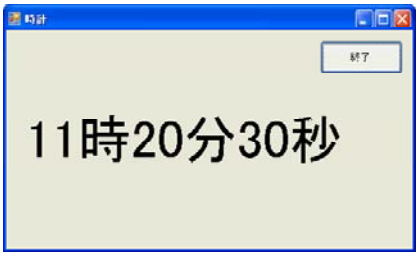
Enabled　　タイマーイベントを有効、無効にします。Trueで有効、Falseで無効

実際にはVB2005のシステムが時刻をカウントするのは18分の１秒なので、間隔の精度は最大で1/18秒です。よって、正確な時刻や時間をこのタイマーだけでは測定できないので注意してください。

(1)　時計の作成

時刻を表示するだけの簡単な時計です。

タイマー内で現在の時刻を取得して、ラベルの書き換えをしています。



①　スタートページで「作成　プロジェクト」をクリックして、「新しいプロジェクト」のダイアログウィンドウを表示させます。

「Windowsアプリケーション」を選択し、プロジェクト名に「時計」と入力して、「ＯＫ」をクリックします。

②　デザイン画面でフォームを選択し、プロパティウインドウで次の値に設定します。

```
Size 500,300
Text 時計
```

③　ツールボックスからボタン(Button)を選択して１つボタンを配置します。

プロパティウィンドウで次のように設定します

```
Button1 Text 終了
```

④　ツールボックスからラベル(Label)を選択して１つ配置します。

プロパティウィンドウで次のように設定します

```
Font Size  72
```

⑤　ツールバーのコンポーネントの項目からタイマーを選択して配置します。

⑥　コードを記入します

```
Public Class Form1
    Public Tn As Date                                    時刻用の変数を宣言

    Private Sub Form1_Load(ByVal sender As System.Object, _
      ByVal e As System.EventArgs) Handles MyBase.Load
        Timer1.Interval = 500
        Timer1.Enabled = True
    End Sub
```

```
Private Sub Timer1_Tick(ByVal sender As System.Object, _
  ByVal e As System.EventArgs) Handles Timer1.Tick
    Tn = Now                                          現在の時刻を保存
    Label1.Text = Tn.Hour.ToString + "時" _
                + Tn.Minute.ToString + "分" _
                + Tn.Second.ToString + "秒"
End Sub

Private Sub Button1_Click(ByVal sender As System.Object, _
  ByVal e As System.EventArgs) Handles Button1.Click
    Me.Close()
End Sub

End Class
```

⑦　コードを入力し終えたら、開始ボタン  を押して作動を確認しましょう。

⑧　すべてを保存  をクリックして、ソリューションを保存しましょう。

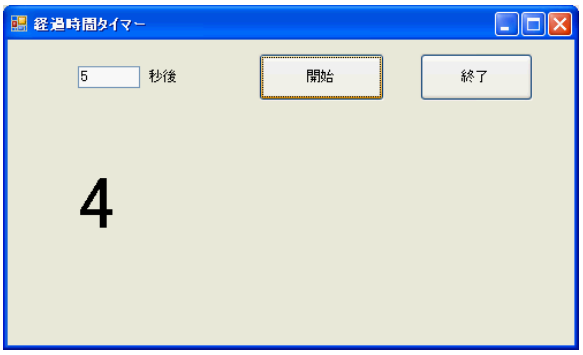
※　発展問題

Private Sub Timer1_TickのLabel1.Textを次のように書き換えてみましょう。

```
Label1.Text = Tn.Month.ToString + "月" _
            + Tn.Day.ToString + "日" _
            + vbCrLf _
            + Tn.Hour.ToString+"時" _
            + Tn.Minute.ToString + "分" _
            + Tn.Second.ToString + "秒"
```

【ワンポイント】
ラベルやテキストボックス内で改行したいときには、「vbCrLf」を入れましょう

(2)　経過時間タイマーの作成



　指定された時間になると「時間です」と表示するタイマーを作ります。
経過時刻を計算するにはDateAdd関数を用います。
　例えば、現在の時刻に10秒を加算した時刻を求める場合は、

Tn = DateAdd(DateInterval.Second, 10, Now)

になります。

①　スタートページで「作成　プロジェクト」をクリックして、「新しいプロジェクト」のダイアログウィンドウを表示させます。
　「Windowsアプリケーション」を選択し、プロジェクト名に「経過時間タイマー」と入力して、「ＯＫ」をクリックします。

②　デザイン画面でフォームを選択し、プロパティウインドウで次の値に設定します。

```
Size 500,300
Text 経過時間タイマー
```

③　ツールボックスからボタン(Button)を選択して２つボタンを配置します。
プロパティウィンドウで次のように設定します

```
Button1 Text 開始
Button2 Text 終了
```

④　ツールボックスからラベル(Label)を選択して２つ配置します。
プロパティウィンドウで次のように設定します

```
Label1 Text　　秒後
Label2　Font Size　72
```

⑤　テキストボックスを１つ配置します。

⑥　ツールバーのコンポーネントの項目からタイマーを選択して配置します。

⑦　コードを記入します

```
Public Class Form1
    Public Tn As Date                          時刻用の変数を宣言
    Public Ts As Date

    Private Sub Form1_Load(ByVal sender As System.Object, _
      ByVal e As System.EventArgs) Handles MyBase.Load
        TextBox1.Text = "5"
        Label2.Text = ""
    End Sub
```

```
    Private Sub Button1_Click(ByVal sender As System.Object, _
      ByVal e As System.EventArgs) Handles Button1.Click
        Tn = DateAdd(DateInterval.Second, Val(TextBox1.Text), Now)
        Ts = Now
        Timer1.Interval = 500
        Timer1.Enabled = True
    End Sub

    Private Sub Timer1_Tick(ByVal sender As System.Object, _
      ByVal e As System.EventArgs) Handles Timer1.Tick
        If Tn >= Now Then                                  経過時間を計算します
            Label2.Text = (Now - Ts).Seconds.ToString
        Else
            Label2.Text = "時間です"
            Timer1.Enabled = False
        End If
    End Sub

    Private Sub Button1_Click(ByVal sender As System.Object, _
      ByVal e As System.EventArgs) Handles Button1.Click
        Me.Close()
    End Sub

End Class
```

⑧　コードを入力し終えたら、開始ボタン  を押して作動を確認しましょう。

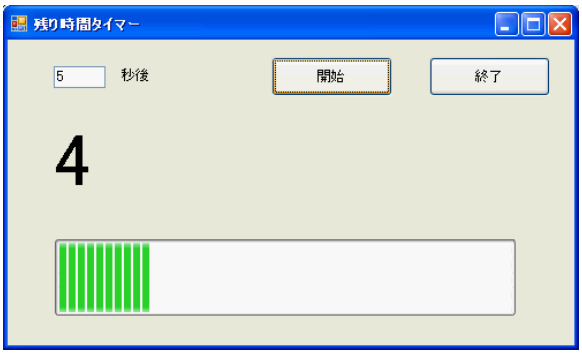
⑨　すべてを保存  をクリックして、ソリューションを保存しましょう。

※　発展問題

- 表示に「秒」をつけてみましょう。
- 次のように変更すると秒を分に変えることができます。

```
Tn = DateAdd(DateInterval.Minute, Val(TextBox1.Text), Now)
Label2.Text = (Tn - Now).Minutes.ToString
```

(3)　残り時間タイマーの作成



指定された時間まで秒数を減算表示します。プログレスバーをつけて、時間が減っていく様子が分かるようにしています。

基本的構造は「経過時間タイマー」と一緒です。ちょっと改変するだけで作ることができます。

①　スタートページで「作成　プロジェクト」をクリックして、「新しいプロジェクト」のダイアログウィンドウを表示させます。

「Windowsアプリケーション」を選択し、プロジェクト名に「残り時間タイマー」と入力して、「ＯＫ」をクリックします。

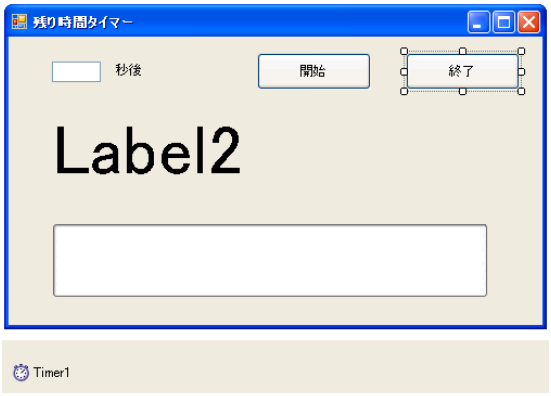
②　デザイン画面でフォームを選択し、プロパティウインドウで次の値に設定します。

```
Size 500,300
Text 残り時間タイマー
```

③　ツールボックスからボタン(Button)を選択して２つボタンを配置します。

プロパティウィンドウで次のように設定します

```
Button1 Text 開始
Button2 Text 終了
```



④　ツールボックスからラベル(Label)を選択して２つ配置します。

プロパティウィンドウで次のように設定します

```
Label1 Text　　秒後
Label2　Font Size　72
```

⑤　テキストボックスを１つ配置します。

⑥　ツールバーのコンポーネントの項目からタイマーを選択して配置します。

⑦　プログレスバーを配置します

⑧　コードを記入します（経過時間タイマーと異なる部分に波下線をつけています）

```
Public Class Form1
    Public Tn As Date
    Public Ts As Date

    Private Sub Form1_Load(ByVal sender As System.Object, _
      ByVal e As System.EventArgs) Handles MyBase.Load
        TextBox1.Text = "5"
        Label2.Text = ""
    End Sub

    Private Sub Button1_Click(ByVal sender As System.Object, _
```

時刻用の変数を宣言

```
    ByVal e As System.EventArgs) Handles Button1.Click
      Tn = DateAdd(DateInterval.Second, Val(TextBox1.Text), Now)
      Ts = Now
      Timer1.Interval = 500
      Timer1.Enabled = True
      ProgressBar1.Minimum = 0
      ProgressBar1.Maximum = (Tn - Ts).Seconds
  End Sub

  Private Sub Timer1_Tick(ByVal sender As System.Ob
```

経過時間を計算します

```
ject, _
    ByVal e As System.EventArgs) Handles Timer1.Tick
      If Tn >= Now Then
          Label2.Text = (Tn - Now).Seconds.ToString + 1
          ProgressBar1.Value = (Now - Ts).Seconds
      Else
          Label2.Text = "時間です"
          Timer1.Enabled = False
      End If
  End Sub

  Private Sub Button1_Click(ByVal sender As System.Object, _
    ByVal e As System.EventArgs) Handles Button1.Click
      Me.Close()
  End Sub

End Class
```

⑨　コードを入力し終えたら、開始ボタン を押して作動を確認しましょう。

⑩　すべてを保存 をクリックして、ソリューションを保存しましょう。

６　グラフィックス

フォームやピクチャーボックスには、直線や円などのグラフィックスを表示することができます。

左上隅が原点(0,0)になります。右方向をＸの正の向き、下方向をＹの正の向きです。

(1)　直線、長方形、円の表示

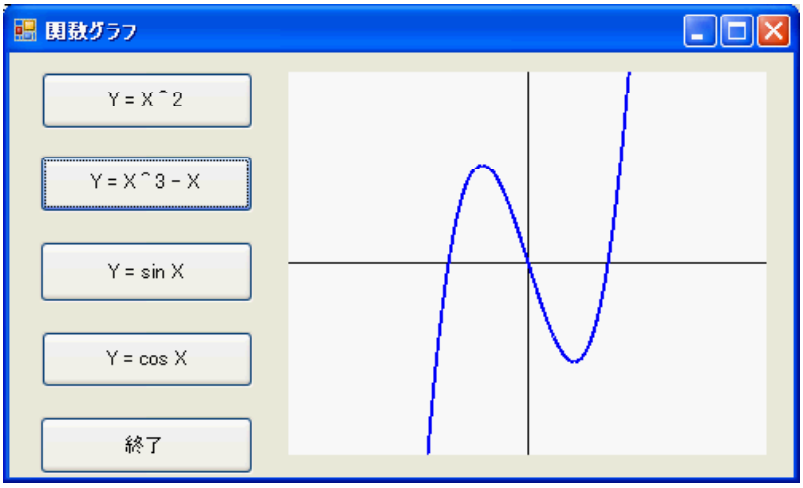
VB6までと異なり、グラフィックスを描くためには、次のように宣言をしてから描く必要があります。

```
Dim mypen As New Pen(Color.Blue, 2)          青色の太さ２のペンを宣言
Dim g As Graphics = PictureBox1.Create       ピクチャーボックス１に描く
g.Clear(Color.White)                         白色で描画面を消去
g.DrawLine(mypen, x1, y1, x2, y2)            mypenで定義した色で線を引く
g.DrawLine(Pens.Black, x, y, width, height)      黒色で線を引く
g.DrawRectangle(Pens.Red, x, y, width, height)   赤色の線で四角形を描く
g.FillRectangle(Brushes.Green, x, y, width, height)  緑色で塗った四角形
g.DrawEllipse(Pens.Yellow,  x, y, width, height)     黄色の線で楕円
g.FillEllipse(Brushes.Violet,  x, y, width, height)  紫色で塗った楕円
```

(2)　関数グラフの作成

グラフィックの機能を使って、関数のグラフを描いてみましょう。

式が異なるだけなので、１つのプロシージャで複数のイベントを処理しています。



①　スタートページで「作成　プロジェクト」をクリックして、「新しいプロジェクト」のダイアログウィンドウを表示させます。

「Windowsアプリケーション」を選択し、プロジェクト名に「関数グラフ」と入力して、「ＯＫ」をクリックします。

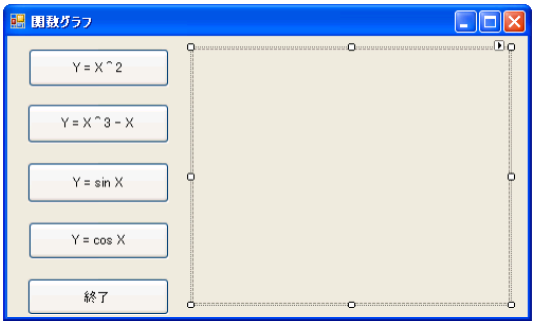
②　デザイン画面でフォームを選択し、プロパティウインドウで次の値に設定します。

```
Size 500,300
Text 関数グラフ
```

③　ツールボックスからピクチャーボックス(PictureBox)を選択して配置します。

```
Size 300,240
```

④　ツールボックスからボタン(Button)を選択して５つボタンを配置します。
プロパティウィンドウで次のように設定します




```
Button1 Text   Y = X ^ 2
Button2 Text   Y= X ^ 3 - 4X
Button3 Text   Y= sin X
Button4 Text   Y = cos X
Button5 Text   終了
```

⑤　コードを記入します

```
Public Class Form1

    Public XMax, YMax As Single                          共通の変数を定義します
    Public Xa, Ya As Single

    Private Sub Form1_Load(ByVal sender As System.Object, _
      ByVal e As System.EventArgs) Handles MyBase.Load
        XMax = PictureBox1.Width                         グラフを描く場所の
        YMax = PictureBox1.Height                        大きさを設定します
        Xa = 6                                           横の倍率
        Ya = 6                                           縦の倍率
    End Sub

                          Button1～4のイベントを受け取れるように
                           Handlesの後に記述します
    Private Sub Button1_Click(ByVal sender As System.Object, _
      ByVal e As System.EventArgs) _
      Handles Button1.Click, Button2.Click, Button3.Click, Button4.Click
        Dim i As Integer
        Dim X1, X2, Y1, Y2 As Single
        Dim mypen As New Pen(Color.Blue, 2)
        Dim g As Graphics = PictureBox1.CreateGraphics
        g.Clear(Color.White)
        g.DrawLine(Pens.Black, 0, YMax / 2, XMax, YMax / 2)
        g.DrawLine(Pens.Black, XMax / 2, YMax, XMax / 2, 0)
        g.DrawRectangle(Pens.Brown, 0, 0, XMax - 1, YMax - 1)
        For i = 0 To XMax
            X1 = (Xa * 2 / XMax) * (i - (XMax / 2))
            If sender.Text = Button1.Text Then           sender.Textでどのボタン
                Y1 = X1 * X1                             からのイベントなのかを
            ElseIf sender.Text = Button2.Text Then       判断します
```

```
            Y1 = X1 * X1 * X1 - 4 * X1
        ElseIf sender.Text = Button3.Text Then
            Y1 = Math.Sin(X1 / Math.PI)
        ElseIf sender.Text = Button4.Text Then
            Y1 = Math.Cos(X1 / Math.PI)
        End If
        Y1 = -Y1 * YMax / Ya / 2 + (YMax / 2)
        If i > 0 Then
            g.DrawLine(mypen, X2, Y2, i, Y1)
        End If
        X2 = i
        Y2 = Y1
    Next
End Sub

Private Sub Button5_Click(ByVal sender As System.Object, _
  ByVal e As System.EventArgs) Handles Button5.Click
    Me.Close()
End Sub

End Class
```

iが0の時にはX2,Y2が0なので描きません

⑦　コードを入力し終えたら、開始ボタン を押して作動を確認しましょう。

⑧　すべてを保存 をクリックして、ソリューションを保存しましょう。

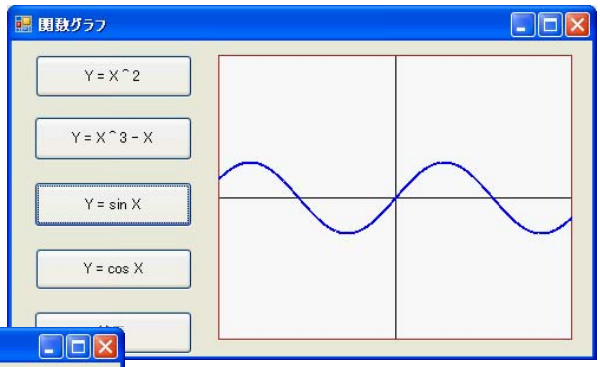
※　発展問題

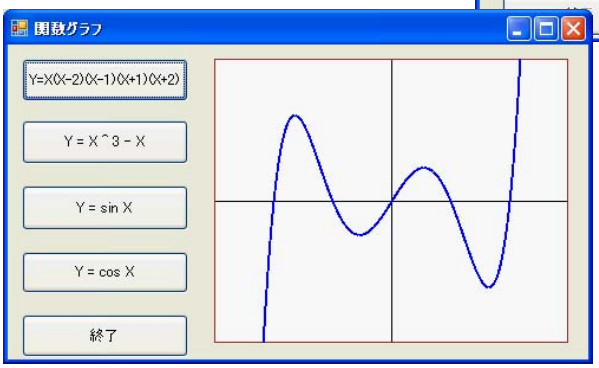
・　表示する倍率を変えてみましょう

・　関数を変えてみましょう。

例：　Y = 2X ^ 2 -4x +4

Y=X(X-2)(X-1)(X+1)(X+2)





(3)　フォームの色を変える

コンピュータで表示している色は、RGBの数値（0～255）で表すことができます。トラックバーをドラッグするとそれぞれの数値が変化し、フォームの色が変化します。

透明度（アルファチャンネル）を変化させるとフォームの透明度も変えることができます。

①　スタートページで「作成　プロジェクト」をクリックして、「新しいプロジェクト」のダイアログウィンドウを表示させます。

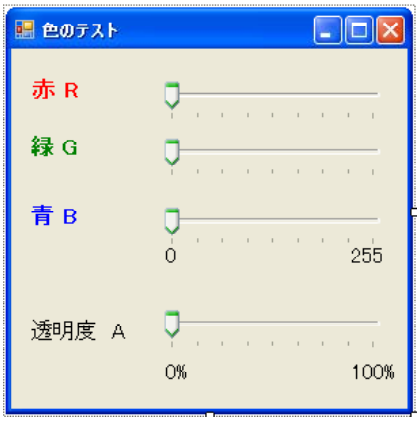
「Windowsアプリケーション」を選択し、プロジェクト名に「色のテスト」と入力して、「ＯＫ」をクリックします。

②　デザイン画面でフォームを選択し、プロパティウインドウで次の値に設定します。

Text	色のテスト

③　ツールボックスからラベル(Label)を８つ配置します。

Label1 Text	赤　R
Label2 Text	緑　G
Label3 Text	青　B
Label4 Text	透明度　A
Label5 Text	0
Label6 Text	255
Label7 Text	0%
Label8 Text	100%



④　ツールボックスの「すべてのWindowsフォーム」から「トラックバー(TrackBar)」を選択して４つ配置します。プロパティウインドウで次の値に設定します。

Maximum	255	最大値の設定
TickFrequency	32	目盛りの設定

⑤　フォームをダブルクリックして「Private Sub Form1_Load」にコードを入力します。

⑥　トラックバー１をダブルクリックして「Private Sub TrackBar1_Scroll」にコードを入力します。

Handles TrackBar1.Scroll のあとに「, TrackBar2.Scroll, TrackBar3.Scroll」を追加して、トラックバー１２３をすべてここで処理します。

⑥　トラックバー４をダブルクリックして「Private Sub TrackBar4_Scroll」にコードを入力します。

```
Public Class Form1

 Private Sub Form1_Load(ByVal sender As System.Object, ByVal e _
 As System.EventArgs) Handles MyBase.Load
     TrackBar1.Value = Me.BackColor.R
     TrackBar2.Value = Me.BackColor.G
     TrackBar3.Value = Me.BackColor.B
     TrackBar4.Value = CInt(Me.Opacity * 255)
 End Sub

 Private Sub TrackBar1_Scroll(ByVal sender As System.Object, ByVal e _
 As System.EventArgs) _
 Handles TrackBar1.Scroll, TrackBar2.Scroll, TrackBar3.Scroll

     Me.BackColor = Color.FromArgb(TrackBar1.Value.ToString, _
                                   TrackBar2.Value.ToString, _
                                   TrackBar3.Value.ToString)

     Label1.Text = "赤R " + TrackBar1.Value.ToString
     Label2.Text = "緑G " + TrackBar2.Value.ToString
     Label3.Text = "青B " + TrackBar3.Value.ToString
 End Sub

 Private Sub TrackBar4_Scroll(ByVal sender As System.Object, ByVal e _
 As System.EventArgs) Handles TrackBar4.Scroll
     Me.Opacity = CSng(TrackBar4.Value / 255)
 End Sub

End Class
```

３つをここで処理します

色の設定

色の数値を表示します

透明度の設定

⑦　コードを入力し終えたら、開始ボタン  を押して作動を確認しましょう。

⑧　すべてを保存  をクリックして、ソリューションを保存しましょう。

7　ツールの追加（コンポーネントの追加）

(1)　メディアプレイヤを制御

Windows Media Playerコンポーネントを用いて，ビデオファイルや音楽ファイルなどを再生することができます。

Media Playerを使用するためには，関連するコンポーネントをプロジェクトに追加します。

①　スタートページで「作成　プロジェクト」をクリックして、「新しいプロジェクト」のダイアログウィンドウを表示させます。

「Windowsアプリケーション」を選択し、プロジェクト名に「MediaPlayer制御」と入力して、「ＯＫ」をクリックします。

②　デザイン画面でフォームを選択し、プロパティウインドウで次の値に設定します。

```
Size　500,500
Text　MediaPlayer制御
```

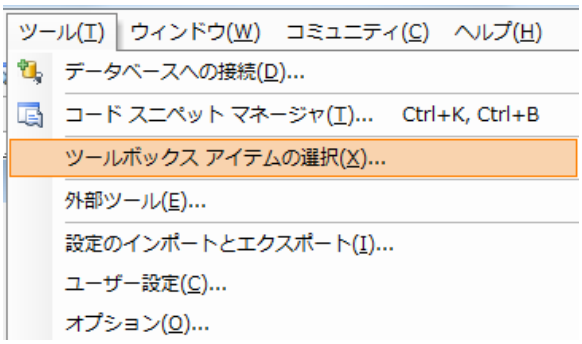
③　コンポーネントの追加を行います。

「ツール」

→「ツールボックスの選択」

を選択します。

（ツールボックスが表示されるまでに若干の時間がかかります）

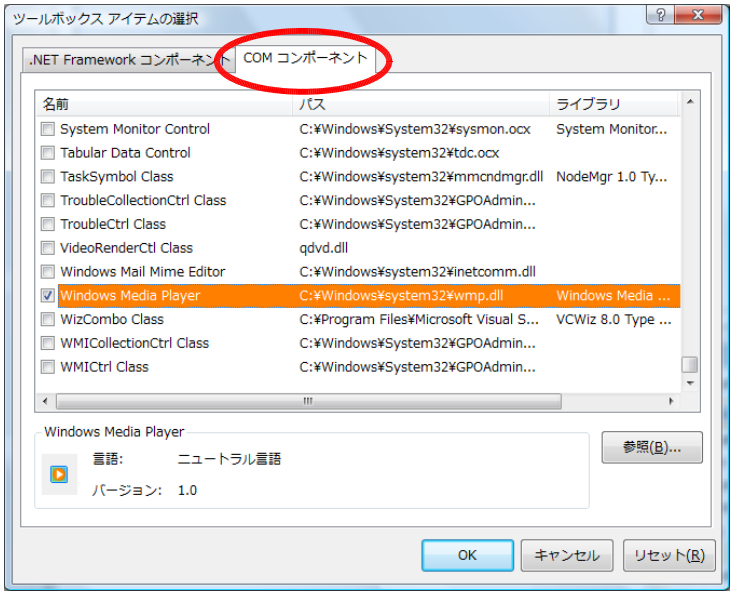


④　「ツールボックスアイテムの選択」のダイアログから

「COMコンポーネント」のタグを選択します。

「Windows Media Player」を選択します。

「OK」をクリックします。

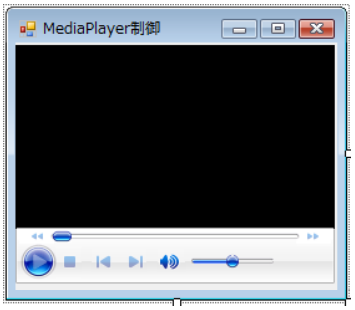


⑤　Windows Media Playerのアイコンがツールボックスに追加されます，

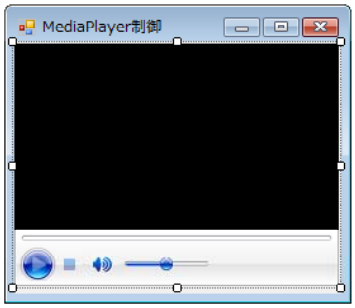
⑥　ツールボックスのWindows Media Player をフォームに貼り付けます。

⑦　プロパティウィンドウでインターフェイスを変更してみます。

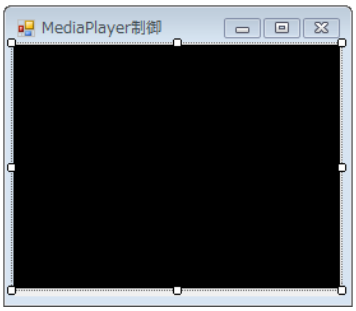
uniMode　full



uniMode　mini



uniMode　none

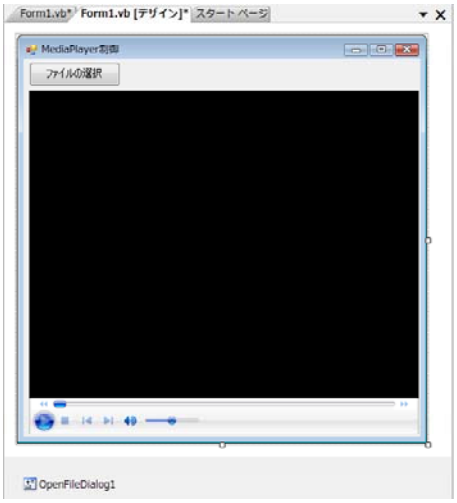


⑦　Button を追加します。

Textを「ファイルの選択」にします。

Text　ファイルの選択

⑧　OpenFileDialog を追加します



⑨　Button1 をダブルクリックして，コードを記入します。

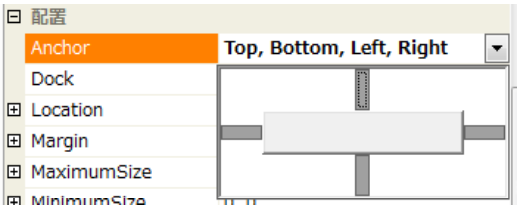
```
Public Class Form1

 Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As _
 System.EventArgs) Handles Button1.Click
     OpenFileDialog1.Filter = "*.wmv|*.wmv"
     If OpenFileDialog1.ShowDialog = Windows.Forms.DialogResult.OK Then
         AxWindowsMediaPlayer1.URL = OpenFileDialog1.FileName
     End If
 End Sub

End Class
```

⑩　コードを入力し終えたら、開始ボタン  を押して作動を確認しましょう。

⑪　すべてを保存  をクリックして、ソリューションを保存しましょう。

※　発展問題

Media Playerのプロパティを右図のように設定します。ウィンドウの大きさを変えたとき，貼り付けたMedia Playerの大きさも一緒に変わるようになります。



8　メニューコントロール

メニューには、フォームの上部に表示されるメインメニュー（メニューバー）と、右クリックで表示されるコンテキストメニュー（ポップアップメニュー）があります。どちらもツールボックスからフォームにドラッグすることにより使うことができるようになります。

(1)　メインメニュー

MenuStripをフォームにドラッグすると、コンポーネントトレイに入り、「ここへ入力」 

と表示されます。この部分にメニューの項目を入力します。

メニューは他のウィンドウズアプリケーションと同じような構造にすると、迷わずに使うことができます。

例：ﾌｧｲﾙ　編集　表示　挿入・・・・ﾍﾙﾌﾟ

設定したメニューをダブルクリックすると、そのメニュー項目のプロシージャを入力するコード画面に切り替わります。

半角のマイナス「-」を入力すると区切線表示になります（右クリックからも選択できます)。

(2)　コンテキストメニュー

ContextMenuStripをフォームにドラッグすると、コンポーネントトレイに入ります。右図のように「ここへ入力」が表示されます。この部分にメニューの項目を入力します。

9　メッセージボックス

アプリケーションの中でメッセージを表示するには､ダイアログボックスを使います。ダイアログボックスには、MsgBox()、MessageBox、InputBox()があります。

①　MsgBox()

エラー警告、注意などを表示するのに適しています。

MsgBox()は関数です。

用法　MsgBox(表示テキスト,ボタン,タイトル)

ボタンの種類

MsgBoxStyle.OKOnly	「OK」を表示
MsgBoxStyle.OKCancel	「OK」「キャンセル」を表示
MsgBoxStyle.YesNoCancel	「はい」「いいえ」「キャンセル」を表示
MsgBoxStyle.YesNo	「はい」「いいえ」を表示

アイコンの種類

MsgBoxStyle.Critical	警告アイコン
MsgBoxStyle.Question	問い合わせアイコン
MsgBoxStyle.Exclamation	注意アイコン
MsgBoxStyle.Information	情報アイコン

※　ボタンとアイコンを組み合わせるには「Or」でつなぎます。

使用例　MsgBox(″OKを押してください″)
　　MsgBox(″はい、いいえのどちら？″,YesNo)
　　MsgBox(″どちら？″,MsgBoxStyle。YesNo,″選択″)

MsgBox()は関数なので、押したキーの番号を返します。
　1:OK、2:Cancel、6:Yes(はい)、7:No(いいえ)

② MessageBox

MsgBox()と用法はほぼ一緒です。押したボタンは、戻り値によって判断します。

用法

　MessageBox.Show(表示テキスト,タイトル,ボタン,アイコン)

※　位置がMsgBox()と異なります。

③ InputBox()

テキストデータを入力してもらう場合に使用します。テキストは文字列型として取得されます。

用法

　InputBox(表示テキスト,タイトル,デフォルト文字列)

※　デフォルト文字列は入力推奨する文字を記入しますが、省略できます。

使用例

```
Dim Moji As String
Moji = InputBox(″意見を記入してください″)

Dim Suuti As Single
Suuti = Val( InputBox(″数値を入力してください″))　　Val関数で数値に変換
```

10　エラー処理

プログラムに誤りがあると実行が止まります。コードの誤りがなくとも、実行したときの場合によってエラーが発生する場合があります。発生したエラーに対して、適切な処理を行うのがエラー処理です。

エラー処理の構文ではTry文が使われます。

```
Try
    エラーの発生する可能性のあるプログラム
Catch
    エラー発生時の対処のプログラム
Filaliy
    例外の有無にかかわらず実行するプログラム
End Try
```

※　Filallyの部分は省略できます。

※　Try文の中にTry文を入れることもできます。

※　VB6までは「ON ERROR GOTO」を用いていましたが、Try文により、エラー処理の範囲と処理方法がよりわかりやすくなりました。

※　VB2005でも「ON ERROR GOTO」は使用可能ですが、推奨されていません。

11　外部プログラムの実行

外部プログラムを起動するには次のように記述します

System.Diagnostics.Process.Start(Webページアドレス)

これで、ブラウザが起動して、指定したページを表示します。

System.Diagnostics.Process.Start(アプリケーション)

アプリケーションが起動します。

System.Diagnostics.Process.Start(アプリケーション,ファイル名)

指定したファイルを読み込んで、アプリケーションが起動します。

起動しない場合の処理のため、Try文と一緒に用いるのがよいです。

(1)　Webページの表示

①　スタートページで「作成　プロジェクト」をクリックして、「新しいプロジェクト」のダイアログウィンドウを表示させます。

「Windowsアプリケーション」を選択し、プロジェクト名に「Webページの表示」と入力して、「ＯＫ」をクリックします。

②　デザイン画面でフォームを選択し､プロパティウインドウで次の値に設定します。

```
Size 300,300
Text Webページの表示
```

③　ツールボックスからリンクボタン(LinkButton)を選択して配置します。

プロパティウィンドウで次のように設定します。

```
Text　岩手県立総合教育センター
```

④　ツールボックスからボタン(Button)を選択して配置します。

プロパティウィンドウで次のように設定します

Button1 Text	Yahoo

⑤　コードを記入します

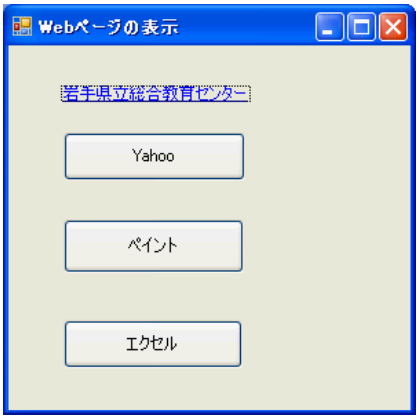
```
Public Class Form1

  Private Sub LinkLabel1_LinkClicked(ByVal sender As System.Object, _
    ByVal e As System.Windows.Forms.LinkLabelLinkClickedEventArgs) _
    Handles LinkLabel1.LinkClicked
      System.Diagnostics.Process.Start("http://www1.iwate-ed.jp/")
      LinkLabel1.LinkVisited = True
    End Sub

  Private Sub Button1_Click(ByVal sender As System.Object, _
    ByVal e As System.EventArgs) Handles Button1.Click
      System.Diagnostics.Process.Start("http://www.yahoo.co.jp/")
  End Sub
End Class
```

⑥　コードを入力し終えたら、開始ボタン  を押して作動を確認しましょう。

⑦　すべてを保存  をクリックして、ソリューションを保存しましょう。

(2)　マイ・ランチャーの作成

上記のプログラムにボタンを追加して自分用のプログラム起動アプリ「マイ・ランチャー」にします。



④　ツールボックスからボタン(Button)を選択して２つ追加します。

プロパティウィンドウで次のように設定します

Button2 Text	ペイント
Button3 Text	エクセル

⑧　コードを追加記入します

```
Private Sub Button2_Click(ByVal sender As System.Object, _
   ByVal e As System.EventArgs) Handles Button2.Click
      System.Diagnostics.Process.Start("mspaint.exe")
End Sub

Private Sub Button3_Click(ByVal sender As System.Object, _
   ByVal e As System.EventArgs) Handles Button3.Click
      System.Diagnostics.Process.Start("excel.exe")
End Sub
```

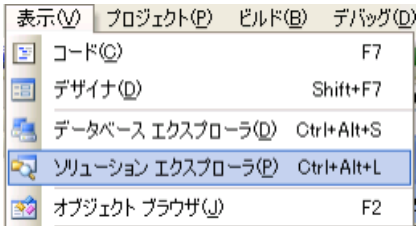
12　配布用プログラムの作成

(1)　アイコンの作成

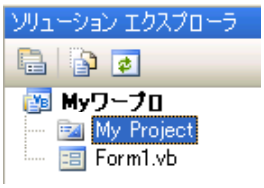
アプリケーションにアイコンを設定する場合にはフォームのプロパティの「アイコン」

この右側のボタンをクリックして、アイコンファイルを選択してください。アイコンファイルは拡張子icoの画像ファイルです。Windows付属のペイントでは作成することができません。作成するためのフリーソフト等がありますので、そちらをご利用ください。

(2)　バージョンの設定

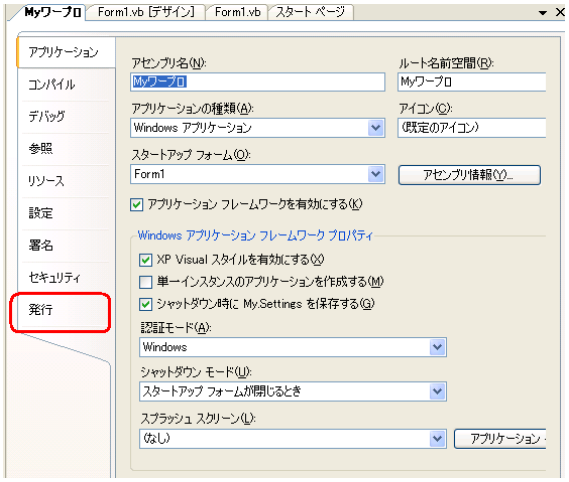


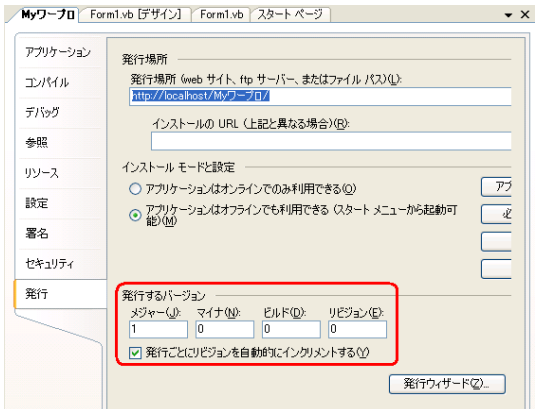
アプリケーションのバージョンは、メニューバーの「表示」→「ソリューションエクスプローラ」でソリューションエクスプローラを表示させます。



ソリューションエクスプローラのウィンドウ内の「My Project」をダブルクリックします。しばらくしますと、下図のような表示が出ます。

「発行」をクリックすると下右図のようになります。





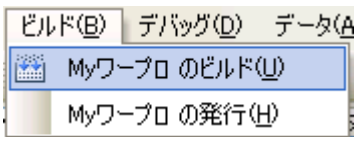
「発行するバージョン」に数値を入れます。

通常、プログラムの大きな改変時にはメジャー番号を増やします。バグをなおした場合などの小さな改変時にはマイナやビルド番号を増やします。

数値を入力し終えたら、このウィンドウの右上の×をクリックして閉じます。

(3)　配布用プログラムの作成

①　配布用のプログラムの作成を作成する場合には、メニューバーの「ビルド」→「○○○のビルド」をクリックします。



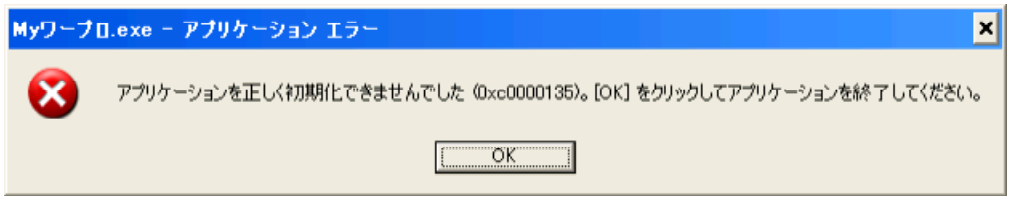
ウィンドウ下部に「ビルドを開始しました」  と表示されます。エラーがなければ「ビルド正常終了」と表示されます。

② 配布用ファイルのコピー

配布用の実行ファイルは、binフォルダの中の**Releaseフォルダ**内に作成されます。「**○○○.exe**」が実行ファイルです。

他のコンピュータで作成したアプリケーションを実行したい場合には、このファイルをコピーしてダブルクリックで動きます。

もし、下図のようなエラーが出る場合には、.NET Framework 2.0をインストールしてください



③ セットアッププログラムの作成について

配布用プログラムの作成には「セットアッププログラム」を用いることもできます。しかし、VB2005の場合、「プロジェクト」→「参照の追加」でファイルを追加していない場合には、Releaseフォルダ内のexeファイルのコピーだけで動くので、ここでは省略します。

(4) .NET Framework 2.0のインストール

VB2005で作成したアプリケーションを動かすためには.NET Framework 2.0のインストールが必要です。次の２つのファイルをインストールしてください。

- dotnetfx.exe
- langpack.exe　（日本語に対応させるためのプログラム）

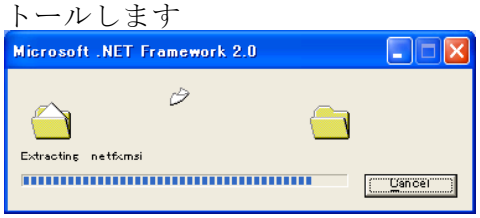
次に、インストールの手順について説明します。

① dotnetfx.exeとlangpack.exeを入手します

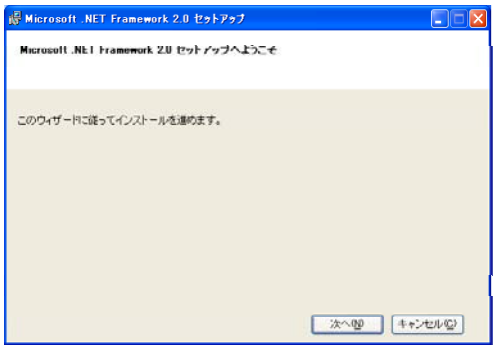
MicrosoftのWebサイト内でダウンロードできるようになっていますので、検索してファイルを入手してください。または、雑誌等の付録CDにも入っているのでそちらを利用しても良いです。

② dotnetfx.exe　をダブルクリックしてインストールします

ア　ファイルのコピーの画面が出ます



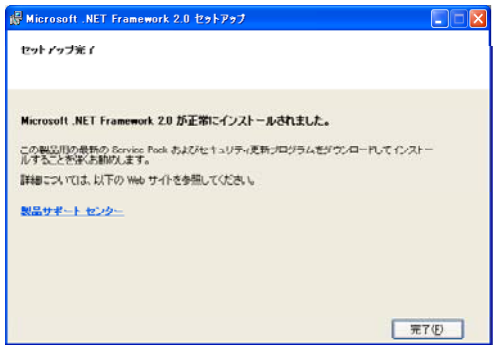
イ　次へをクリックします。



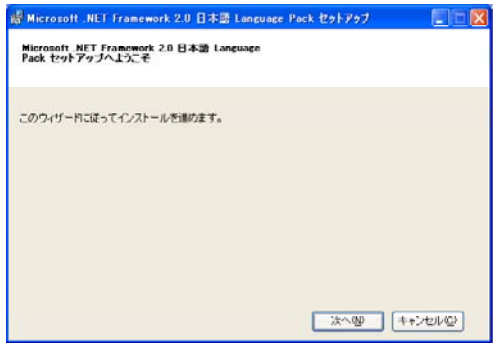
ウ　「同意する」チェックを入れて、
「インストール」をクリックします。

エ　コンポーネントがインストールされます。しばらく時間がかかります。

オ　セットアップ終了の画面が出たら、
「完了」をクリックしてください。



③　次にlangpack.exeをインストールします。
手順は上記と同様です。



《コラム》　プログラムの名前の選定

作成したアプリケーションには名前をぜひ付けてください。名前もないままではかわいそうです。名前を付けるときには、「用途や内容がわかる」「インパクトがある」名前を考えてください。Webで考えた名前を検索して、同じ名前のソフトや商品がないかを確認してください。

《コラム》　ドキュメントファイルの作成

作成したアプリケーションのソフトの内容や使い方のドキュメントファイル（説明文）を書きましょう。説明文を書くことにより、そのアプリケーションの内容や用途を見直すことができます。説明文はテキストファイルでReadme.txtまたはReadme.docに記入します。

次のような項目で記入をしてください。

==========

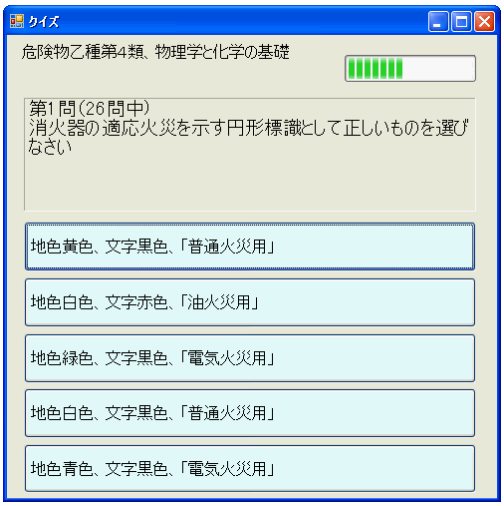
【ソ フ ト名】　Myワープロ
【 対象機種 】　Windows XP, 2000, Me, 98
【 開発言語 】　Visual Basic 2005 Express Edition
【 動作環境 】　.NET Framework 2.0がインストールされている環境
【実行ﾌｧｲﾙ名】　Myワープロ.exe
【 連 絡 先 】

==========

《ソフト紹介》
《ソフトの特徴》
《使用上の注意》
《作動環境について》
《インストールについて》
　インストールの必要はありません。プログラムをダブルクリックするだけで使うことができます。
《プログラムの使い方》
《授業での活用》

第４章　教材作成例

１　クイズ（テキストファイル読み込み）



・問題のテキストファイルを読み込んで表示します。
・回答制限時間は10秒で、プログレスバーで表示します。
・正答の場合には押したボタンが赤色になり、回答時間に応じた得点を加算します。
・誤答の場合は正答のボタンをピンク色で２秒間表示します。
・全問題を終了すると、総得点が表示されます。

①　スタートページで「作成　プロジェクト」をクリックして、「新しいプロジェクト」のダイアログウィンドウを表示させます。
　「Windowsアプリケーション」を選択し、プロジェクト名に「クイズ」と入力して、「ＯＫ」をクリックします。

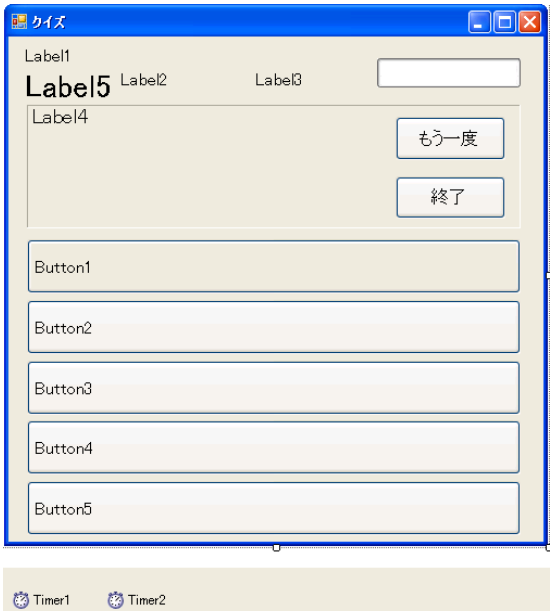
②　デザイン画面でフォームを選択し、プロパティウインドウで次の値に設定します。

```
Size 500,500
Text クイズ
```

③　ツールボックスからラベル(Label)を選択して５つ配置します。

```
Lable1 Font Size  12
Lable2 Font Size  12
Lable3 Font Size  12
Lable4 Font Size  14
Lable5 Font Size  22
```

④　ツールボックスからボタン(Button)を選択して７つボタンを配置します。プロパティウィンドウで次のように設定します




```
Button1 Font Size 12
Button2 Font Size 12
Button3 Font Size 12
Button4 Font Size 12
Button5 Font Size 12
Button6 Font Size 12
Button6 Text もう一度
Button7 Font Size 12
Button7 Text 終了
```

⑤　ツールボックスからプログレスバー(ProgressBar)を１つ配置します。

⑥　ツールボックスからタイマーコントロールを２つ配置します。

⑦　コードを記入します

```
Public Class Form1

 Public Mondai(50, 5) As String                    '問題と答えの変数
 Public Syutudai1(4) As Single                     '答えの順番の変数
 Public Syutudai2(4) As Single                     '答えの順番を変える変数
 Public MondaiZyun1(50) As Single                  '出題順の変数
 Public MondaiZyun2(50) As Single                  '出題順を変える変数
 Public Bangou As Integer                          '出題数のカウント
 Public Tokuten As Integer                         '現在の問題の得点
 Public SouTokuten As Integer                      '総得点
 Public MondaiMax As Integer                       '問題数
 Public Kotae As Integer                           '正答の番号

 Private Sub Form1_Load(ByVal sender As System.Object, _
   ByVal e As System.EventArgs) Handles MyBase.Load
     Dim i As Integer
     Label2.Text = ""
     Label3.Text = ""
     Label4.Text = ""
     Label5.Text = ""
     Try
         Dim Fr As System.IO.TextReader            '問題ファイルの読み込み
         Dim Tx As String
         Fr = My.Computer.FileSystem.OpenTextFileReader("data.txt", _
              System.Text.Encoding.Default)
         Tx = Fr.ReadLine
         Label1.Text = Tx
         Tx = Fr.ReadLine
         MondaiMax = Val(Tx)
         Bangou = 1
         Do
             Mondai(Bangou, 0) = Fr.ReadLine
             Mondai(Bangou, 1) = Fr.ReadLine
             Mondai(Bangou, 2) = Fr.ReadLine
             Mondai(Bangou, 3) = Fr.ReadLine
             Mondai(Bangou, 4) = Fr.ReadLine
             Mondai(Bangou, 5) = Fr.ReadLine
             Bangou = Bangou + 1
```

共通の変数を定義します

```
        Loop While (Bangou <= MondaiMax)
    Catch
    End Try
    ReDim MondaiZyun1(MondaiMax - 1)
    ReDim MondaiZyun2(MondaiMax - 1)
    Randomize()
    For i = 0 To (MondaiMax - 1)
        MondaiZyun1(i) = Rnd()
    Next
    Array.Copy(MondaiZyun1, MondaiZyun2, MondaiMax)
    Array.Sort(MondaiZyun2)
    Bangou = 1
    SyutudaiSub()
    Timer1.Interval = 1000
    Timer2.Interval = 2000
End Sub

Private Sub SyutudaiSub()
    Dim i As Integer
    Dim Jun As Integer
    Dim MJun As Integer
    Label5.Text = ""
    Button1.BackColor = Color.LightCyan
    Button2.BackColor = Color.LightCyan
    Button3.BackColor = Color.LightCyan
    Button4.BackColor = Color.LightCyan
    Button5.BackColor = Color.LightCyan
    Button6.Visible = False
    Button7.Visible = False
    For i = 0 To 4
        Syutudai1(i) = Rnd()
    Next
    Array.Copy(Syutudai1, Syutudai2, 5)
    Array.Sort(Syutudai2)
    MJun = Array.IndexOf(MondaiZyun1, MondaiZyun2(Bangou - 1))
    Label4.Text = "第" + Bangou.ToString + "問 (" + MondaiMax.ToString _
                + "問中)" + vbCrLf + Mondai(MJun + 1, 0)
    Jun = Array.IndexOf(Syutudai2, Syutudai1(0))
    If Jun = 0 Then Kotae = 1
    Button1.Text = Mondai(MJun + 1, Jun + 1)
```

出題（問題提示）

```
    Jun = Array.IndexOf(Syutudai2, Syutudai1(1))
    If Jun = 0 Then Kotae = 2
    Button2.Text = Mondai(MJun + 1, Jun + 1)
    Jun = Array.IndexOf(Syutudai2, Syutudai1(2))
    If Jun = 0 Then Kotae = 3
    Button3.Text = Mondai(MJun + 1, Jun + 1)
    Jun = Array.IndexOf(Syutudai2, Syutudai1(3))
    If Jun = 0 Then Kotae = 4
    Button4.Text = Mondai(MJun + 1, Jun + 1)
    Jun = Array.IndexOf(Syutudai2, Syutudai1(4))
    If Jun = 0 Then Kotae = 5
    Button5.Text = Mondai(MJun + 1, Jun + 1)
    Tokuten = 100
    ProgressBar1.Value = 0
    Timer1.Start()
End Sub

Private Sub Timer1_Tick(ByVal sender As System.Object, _
  ByVal e As System.EventArgs) Handles Timer1.Tick
    ProgressBar1.Value = 100 - Tokuten
    Tokuten = Tokuten - 10
    If Tokuten < 0 Then
        Tokuten = 0
        Label5.ForeColor = Color.Blue
        Label5.Text = "×"
        Hazure()
        Timer1.Stop()
        Timer2.Start()
    End If
End Sub

Private Sub Timer2_Tick(ByVal sender As System.Object, _
  ByVal e As System.EventArgs) Handles Timer2.Tick
    Timer1.Stop()
    Timer2.Stop()
    Label2.Text = ""
    Bangou = Bangou + 1
    SyutudaiSub()
    If Bangou <= MondaiMax Then
        SyutudaiSub()
```

回答時間制限用タイマー

時間切れの場合

正答提示用タイマー

全問題の出題終了時

```
    Else
        Button1.Visible = False
        Button2.Visible = False
        Button3.Visible = False
        Button4.Visible = False
        Button5.Visible = False
        Button6.Visible = True
        Button7.Visible = True
        Label4.Text = "今回の学習結果" + vbCrLf _
                    + "総得点＝" + SouTokuten.ToString + vbCrLf
    End If
End Sub
Private Sub Atari()
    Label5.ForeColor = Color.Red
    Label5.Text = "○"
    SouTokuten = SouTokuten + Tokuten + 10
    Label2.Text = (Tokuten + 10).ToString + "点Ｇｅｔ"
    Label3.Text = "総合点=" + SouTokuten.ToString
End Sub

Private Sub Hazure()
    Select Case Kotae
        Case 1
            Button1.BackColor = Color.Pink
        Case 2
            Button2.BackColor = Color.Pink
        Case 3
            Button3.BackColor = Color.Pink
        Case 4
            Button4.BackColor = Color.Pink
        Case 5
            Button5.BackColor = Color.Pink
    End Select
    Label5.ForeColor = Color.Blue
    Label5.Text = "×"
End Sub

Private Sub Button1_Click(ByVal sender As System.Object, _
  ByVal e As System.EventArgs) Handles Button1.Click
    Timer1.Stop()
```

```
    If Kotae = 1 Then
        Button1.BackColor = Color.Magenta
        Atari()
    Else
        Button1.BackColor = Color.DarkCyan
        Hazure()
    End If
    Timer2.Start()
End Sub

Private Sub Button2_Click(ByVal sender As System.Object, _
  ByVal e As System.EventArgs) Handles Button2.Click
    Timer1.Stop()
    If Kotae = 2 Then
        Button2.BackColor = Color.Magenta
        Atari()
    Else
        Button2.BackColor = Color.DarkCyan
        Hazure()
    End If
    Timer2.Start()
End Sub

Private Sub Button3_Click(ByVal sender As System.Object, _
  ByVal e As System.EventArgs) Handles Button3.Click
    Timer1.Stop()
    If Kotae = 3 Then
        Button3.BackColor = Color.Magenta
        Atari()
    Else
        Button3.BackColor = Color.DarkCyan
        Hazure()
    End If
    Timer2.Start()
End Sub

Private Sub Button4_Click(ByVal sender As System.Object, _
  ByVal e As System.EventArgs) Handles Button4.Click
    Timer1.Stop()
    If Kotae = 4 Then
```

```
        Button4.BackColor = Color.Magenta
        Atari()
    Else
        Button4.BackColor = Color.DarkCyan
        Hazure()
    End If
    Timer2.Start()
End Sub

Private Sub Button5_Click(ByVal sender As System.Object, _
  ByVal e As System.EventArgs) Handles Button5.Click
    Timer1.Stop()
    If Kotae = 5 Then
        Button5.BackColor = Color.Magenta
        Atari()
    Else
        Button5.BackColor = Color.DarkCyan
        Hazure()
    End If
    Timer2.Start()
End Sub

Private Sub Button6_Click(ByVal sender As System.Object, _
  ByVal e As System.EventArgs) Handles Button6.Click
    Dim i As Integer
    Bangou = 1
    Button1.Visible = True
    Button2.Visible = True
    Button3.Visible = True
    Button4.Visible = True
    Button5.Visible = True
    Button6.Visible = False
    Button7.Visible = False
    ReDim MondaiZyun1(MondaiMax - 1)
    ReDim MondaiZyun2(MondaiMax - 1)
    Randomize()
    For i = 0 To (MondaiMax - 1)
        MondaiZyun1(i) = Rnd()
    Next
    Array.Copy(MondaiZyun1, MondaiZyun2, MondaiMax)
```

```
        Array.Sort(MondaiZyun2)
        Bangou = 1
        SyutudaiSub()
    End Sub

    Private Sub Button7_Click(ByVal sender As System.Object, _
      ByVal e As System.EventArgs) Handles Button7.Click
        Me.Close()
    End Sub

End Class
```

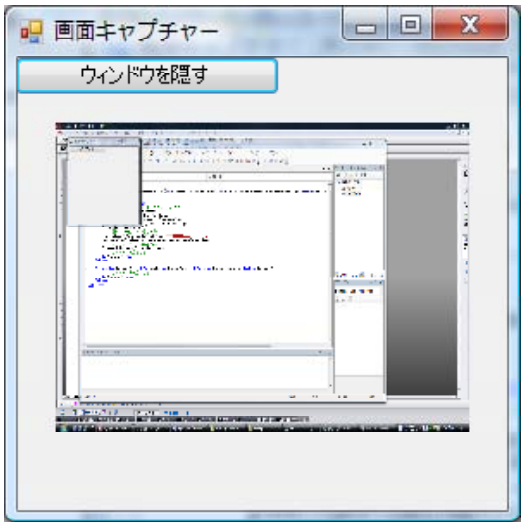
⑧　コードを入力し終えたら、開始ボタン  を押して作動を確認しましょう。

⑨　すべてを保存  をクリックして、ソリューションを保存しましょう。

※　発展問題

- 回答時間を２０秒にしてみましょう。
- 正答数をカウントして、最後に表示してみましょう。

2　画面キャプチャー



タクストレイ（右下のバー）に常駐したアイコンをダブルクリックすると起動します。

表示されているコンピュータの画面を

・　ピクチャーボックスに表示

・　ファイルに保存

・　クリップボードに画像をコピー

します。

保存するファイル形式はjpegです。

①　スタートページで「作成　プロジェクト」をクリックして、「新しいプロジェクト」のダイアログウィンドウを表示させます。

「Windowsアプリケーション」を選択し、プロジェクト名に「画面キャプチャー」と入力して、「ＯＫ」をクリックします。

②　デザイン画面でフォームを選択し、プロパティウインドウで次の値に設定します。

```
Text　画面キャプチャー
```

③　ツールボックスからボタン(Button)を選択して１つ配置します。

```
Text　ウィンドウを隠す
```

④　ツールボックスからピクチャーボックス(PictureBox)を１つ配置します。

```
Size　　240,180
SizeMode　StretchImage
```

⑤　ツールボックスからノティフィアイコン(NotifyIcon)

を１つ配置します。

プロパティウィンドウのIcon(アイコン)をクリックして,「キャプチャ.ico」を選び出して選択します。

⑥　配置したボタン(Button1)をダブルクリックしてコードを入力します。

```
    Me.Visible = False
```

⑦　フォームをダブルクリックしてコードを入力します。

```
    Me.Visible = False
```

⑧　ノティフィアイコン(NotifyIcon)をダブルクリックしてコードを入力します。

```
Private Sub NotifyIcon1_MouseDoubleClick(ByVal sender As System.Object, _
ByVal e As System.Windows.Forms.MouseEventArgs) _
Handles NotifyIcon1.MouseDoubleClick
    Dim imgC As Image
    Dim rct As Rectangle
    Dim jpgFileName As String
```

```
        '       画面をキャプチャーしてピクチャーボックスに表示
        rct = My.Computer.Screen.Bounds
        imgC = New Bitmap(rct.Width, rct.Height)
        Dim grphC As Graphics = Graphics.FromImage(imgC)
        grphC.CopyFromScreen(rct.X, rct.Y, 0, 0, rct.Size)
        PictureBox1.Image = imgC
        '       画像をjpegファイルとして保存
        jpgFileName = DateTime.Now.ToString("yyyyMMddHHmmss") + ".jpg"
        imgC.Save(jpgFileName, System.Drawing.Imaging.ImageFormat.Jpeg)
        '       クリップボードに画像を転送
        Clipboard.SetImage(PictureBox1.Image)
        '       フォームを表示させる
        Me.Visible = True
    End Sub
```

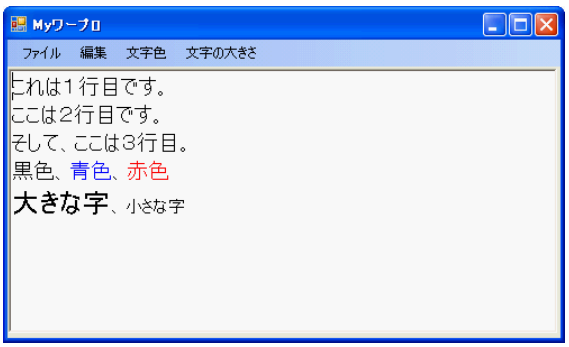
⑨　コードを入力し終えたら、開始ボタン  を押して作動を確認しましょう。

⑩　すべてを保存  をクリックして、ソリューションを保存しましょう。

※　発展問題

- ファイルとして保存しない場合には，どうすればよいでしょうか？
- 一定時間ごとに画像をファイルに保存する場合にはどうすればよいでしょうか？

3　Myワープロの作成



①　スタートページで「作成　プロジェクト」をクリックして、「新しいプロジェクト」のダイアログウィンドウを表示させます。

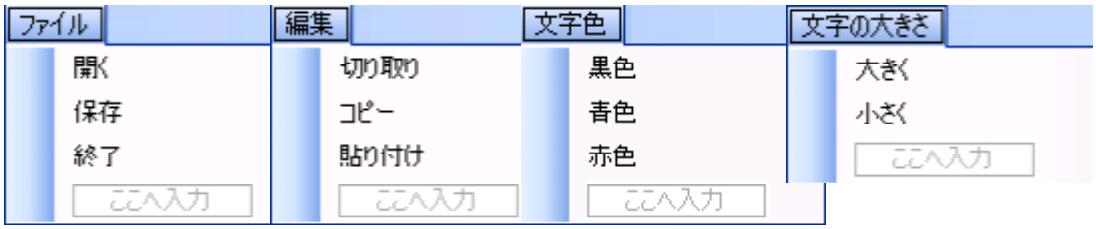
「Windowsアプリケーション」を選択し、プロジェクト名に「Myワープロ」と入力して、「ＯＫ」をクリックします。

②　デザイン画面でフォームを選択し、プロパティウインドウで次の値に設定します。

```
Size 500,300
Text Myワープロ
```

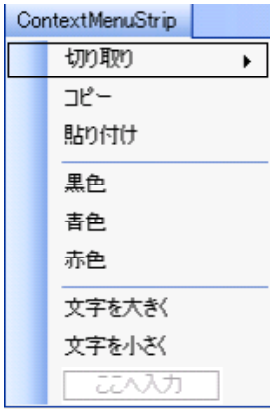
③　ツールボックスからメインメニュー(MenuStrip)を選択して配置します。

下図のように「ここへ入力」に文字を入力します



④　ツールボックスからコンテキストメニュー(ContextMenuStrip)を配置します。

右図のように「ここへ入力」に文字を入力します。



⑤　オープンファイルダイアログ(OpenFileDialog)を配置します。

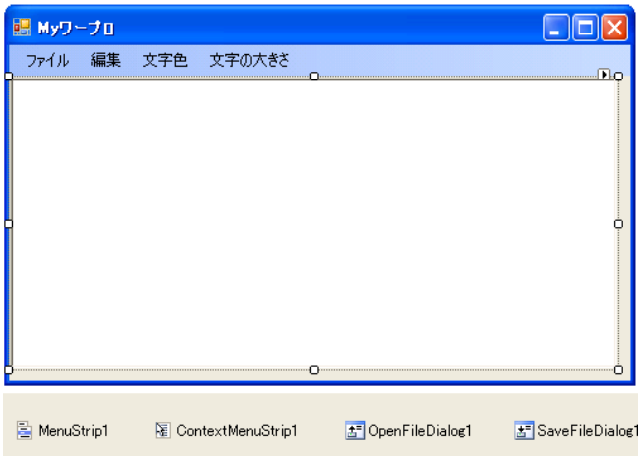
⑥　セーブファイルダイアログ(SaveFileDialog)を配置します。

⑦　リッチテキストボックス(RichTextBox)を配置します。

プロパティを次のように設定します。

```
Size　480,230
```

右図のように配置します。



⑧　コードを記入します

```
Public Class Form1

  Private Sub 開くToolStripMenuItem_Click(ByVal sender As System.Object, _
    ByVal e As System.EventArgs) Handles 開くToolStripMenuItem.Click
      Dim Fn As String
      If OpenFileDialog1.ShowDialog = Windows.Forms.DialogResult.OK Then
          OpenFileDialog1.Filter = "ﾘｯﾁﾃｷｽﾄ(*.rtf)|*.rtf|全てのﾌｧｲﾙ(*.*)|*.*"
          Fn = OpenFileDialog1.FileName
          RichTextBox1.LoadFile(Fn, RichTextBoxStreamType.RichText)
      End If
  End Sub

  Private Sub 保存ToolStripMenuItem_Click(ByVal sender As System.Object, _
    ByVal e As System.EventArgs) Handles 保存ToolStripMenuItem.Click
      Dim Fn As String
      If SaveFileDialog1.ShowDialog = Windows.Forms.DialogResult.OK Then
          SaveFileDialog1.Filter = "ﾘｯﾁﾃｷｽﾄ(*.rtf)|*.rtf|全てのﾌｧｲﾙ(*.*)|*.*"
          Fn = SaveFileDialog1.FileName
           If Fn.IndexOf(".rtf") = -1 Then
              Fn = Fn + ".rtf"
          End If
          RichTextBox1.SaveFile(Fn, RichTextBoxStreamType.RichText)
      End If
  End Sub

  Private Sub 終了ToolStripMenuItem_Click(ByVal sender As System.Object, _
    ByVal e As System.EventArgs) Handles 終了ToolStripMenuItem.Click
      Me.Close()
  End Sub

  Private Sub 切り取りToolStripMenuItem_Click(ByVal sender As _
    System.Object, ByVal e As System.EventArgs) _
    Handles 切り取りToolStripMenuItem.Click, _
            切り取りToolStripMenuItem1.Click
      RichTextBox1.Cut()
      RichTextBox1.Focus()
  End Sub

  Private Sub コピーToolStripMenuItem_Click(ByVal sender As _
```

```
    System.Object, ByVal e As System.EventArgs) _
    Handles コピーToolStripMenuItem.Click, _
            コピーToolStripMenuItem1.Click
        RichTextBox1.Copy()
        RichTextBox1.Focus()
End Sub

Private Sub 貼り付けToolStripMenuItem_Click(ByVal sender As _
    System.Object, ByVal e As System.EventArgs) _
    Handles 貼り付けToolStripMenuItem.Click, _
            貼り付けToolStripMenuItem1.Click
        RichTextBox1.Paste()
        RichTextBox1.Focus()
End Sub

Private Sub 黒色ToolStripMenuItem_Click(ByVal sender As _
    System.Object, ByVal e As System.EventArgs) _
    Handles 黒色ToolStripMenuItem.Click, _
            黒色ToolStripMenuItem1.Click
        RichTextBox1.SelectionColor = Color.Black
        RichTextBox1.Focus()
End Sub

Private Sub 青色ToolStripMenuItem_Click(ByVal sender As _
    System.Object, ByVal e As System.EventArgs) _
    Handles 青色ToolStripMenuItem.Click, _
            青色ToolStripMenuItem1.Click
        RichTextBox1.SelectionColor = Color.Blue
        RichTextBox1.Focus()
End Sub

Private Sub 赤色ToolStripMenuItem_Click(ByVal sender As _
    System.Object, ByVal e As System.EventArgs) _
    Handles 赤色ToolStripMenuItem.Click, _
            赤色ToolStripMenuItem1.Click
        RichTextBox1.SelectionColor = Color.Red
        RichTextBox1.Focus()
End Sub

Private Sub 大きくToolStripMenuItem_Click(ByVal sender As _
```

```
    System.Object, ByVal e As System.EventArgs) _
    Handles 大きくToolStripMenuItem.Click, _
            文字を大きくToolStripMenuItem.Click
      Dim FSize As Single
      FSize = RichTextBox1.SelectionFont.Size + 1
      RichTextBox1.SelectionFont = _
          New Font(RichTextBox1.SelectionFont.Name, _
          FSize, RichTextBox1.SelectionFont.Style)
      RichTextBox1.Focus()
  End Sub

  Private Sub 小さくToolStripMenuItem_Click(ByVal sender As _
    System.Object, ByVal e As System.EventArgs) _
    Handles 小さくToolStripMenuItem.Click, _
            文字を小さくToolStripMenuItem.Click
      Dim FSize As Single
      FSize = RichTextBox1.SelectionFont.Size - 1
      RichTextBox1.SelectionFont = _
         New Font(RichTextBox1.SelectionFont.Name, _
         FSize, RichTextBox1.SelectionFont.Style)
      RichTextBox1.Focus()
  End Sub

End Class
```

⑨　コードを入力し終えたら、開始ボタン  を押して作動を確認しましょう。

⑩　すべてを保存  をクリックして、ソリューションを保存しましょう。

※　発展問題

下記のプロシージャを加えることにより、フォームの大きさを変更したとき、リッチテキストボックスの大きさも変わるようにできます。

```
  Private Sub Form1_Resize(ByVal sender As Object, _
    ByVal e As System.EventArgs) Handles Me.Resize
      RichTextBox1.Width = Me.Width - 12
      RichTextBox1.Height = Me.Height - 64
  End Sub
```

※　保存した文書ファイルは、ワードや一太郎から読み込むこともできます。もちろん、色や字の大きさなども変えずに読むことができます。

第５章　リファレンス編

１　関数編

(1)　算術関数、メソッド

Fix関数　　数値の整数値を返します

例　数値 = Fix(1.5)　　数値は 1

数値 = Fix(-1.5)　　数値は-1

Int関数　　数値を超えない値の整数値を返す

例　数値 = Int(1.5)　　数値は 1

数値 = Int(-1.5)　　数値は-2

Logメソッド　対数を返します

例　数値 = Math.Log(1)　数値は 1

Rnd関数　　乱数を返す

例　数値 = Rnd()　　数値は０以上、１未満の間の乱数

用法　Randomize　　乱数表の再生成

数値 = Int(Rnd()*10+1)　　数値は 1～10の乱数

Roundメソッド　四捨五入します

例　数値 = Math.Round(1.4142 , 1)　　数値は1.4

Sinメソッド　サイン値を返します（引数の単位はラジアン）

例　数値 = Math.Sin(30 *Math.PI /180)　　数値は0.5

Cosメソッド　コサイン値を返します（引数の単位はラジアン）

例　数値 = Math.Cos(60 *Math.PI /180)　　数値は0.5

Tanメソッド　タンジェント値を返します（引数の単位はラジアン）

例　数値 = Math.Tan(45 *Math.PI /180)　　数値は 1

Sqrtメソッド　平方根を返します

例　数値 = Math.Sqrt(9)　　数値は 3

(2)　配列の関数

Filter関数　　文字配列の各要素から、条件にあった要素を抜き出す

例　配列B = Filter(配列A, ″愛″)　　配列Bに「愛」をふくむ要素

Split関数　　文字列を指定の文字で分割して、配列化します

例　配列B = Split(文字列A , vbCrLf)　　改行コードで分割

(3)　入出力の関数

InputBox関数　入力ダイアログを表示します

例　　文字列 = InputBox("説明文", "タイトル", "デフォルト文字列")

デフォルト文字列は省略可能

MsgBox関数　　メッセージボックスを表示します

例　　数値 = MsgBox("説明文", MsgBoxStyle.YesNo, "タイトル")

ボタンの例　MsgBoxStyle.YesNo　　「はい」「いいえ」

MsgBoxStyle.YesNoCansel「はい」「いいえ」「キャンセル」

戻り値　　MsgBoxResult.Yes　　「はい」を選択

MsgBoxResult.No　　「いいえ」を選択

MsgBoxResult.Cansel　「キャンセル」を選択

(4)　型変換の関数

CDate関数　　日付型に変換します(ありえない日付を指定するとエラー)

例　　日付型 = CDate("05/8/31")　　日付型は2005/8/31

CDbl関数　　倍精度浮動小数点型に変換します

CSng関数　　単精度浮動小数点型に変換します

CInt関数　　整数型に変換します

CStr関数　　文字列に変換します

例　　文字列A = CStr(3.14)　　文字列Aは「3.14」

Asc関数　　文字コードに変換します

例　　整数値 = Asc("a")　　整数値は97

Chr関数　　文字コードを文字に変換します

例　　文字列 = Chr(97)　　文字列は「a」

Hex関数　　数値を16進数に変換します

例　　文字列 = Hex(15)　　文字列は「F」

Oct関数　　数値を８進数に変換します

例　　文字列 = Oct(10)　　文字列は「12」

Val関数　　文字列を数値に変換します

例　　数値 = Val("123.4")　　数値は123.4

(5)　文字列操作の関数

Mid関数　　文字列内の指定位置から文字列を取り出します

例　文字列 = Mid("abcdefghi", 2 ,3)　文字列は「bcd」

Left関数　　文字列の左から指定位置までの文字列を取り出します

例　文字列 = Microsoft.VisualBasic.Left("abcdef",3)

文字列は「abc」

Right関数　　文字列の右から指定位置までの文字列を取り出します

例　文字列 = Microsoft.VisualBasic.Right("abcdef",3)

文字列は「def」

Trim関数　　文字列の左右にあるスペースを取り除きます

例　文字列 = Trim("　abc　")　　文字列は「abc」

LSet関数　　左から指定した長さに文字列をそろえます

例　文字列 = LSet("abcdefg", 4)　　文字列は「abcd」

RSet関数　　右から指定した長さに文字列をそろえます

例　文字列 = RSet("abc", 6)　　文字列は「　abc」

Len関数　　文字列の文字数を数値で返します

例　数値 = Len("abcあいうえお")　　数値は8

InStr関数　　文字列から文字を検索してその位置を数値で返します

例　数値 = InStr("abcdefg","bc")　　数値は2

InStrRev関数　文字列から文字を検索して後ろからの位置を返します

例　数値 = InStr("abc_abc","bc")　　数値は6

RePlace関数　文字列中の文字を置き換えます

例　文字列 = RePlace("abcdef","c","X")　文字列は「abXdef」

UCase関数　　小文字を大文字にします

例　文字列 = UCase("abcあいう")　　文字列は「ABCあいう」

LCase関数　　大文字を小文字にします

例　文字列 = LCase("ABCあいう")　　文字列は「abcあいう」

StrConv関数　文字列を指定した形式に変換します

例　文字列= StrConv(″あい″,vbStrConv.Katakana)　文字列は「アイ」

例　文字列= StrConv(″アイ″,vbStrConv.Hiragana)　文字列は「あい」

例　文字列= StrConv(″ｱｲｳ″,vbStrConv.Wide)　文字列は「アイウ」

Space関数　指定した数のスペース文字列を返します

例　文字列 = Space(4)　文字列は「　　」

StrDup関数　指定した数の文字列を返します

例　文字列 = StrDup(5, ″★″)　文字列は「★★★★★」

(6)　日時の関数

FormatDateTime関数　日付型を指定の日付や時刻の書式にします

例　文字列 = FormatDateTime(Now, DateFormat.ShortTime)

文字列は今の時刻を「hh:mm」形式ので表す

Format関数　指定した書式に変換します

例　文字列 = Format(1980, ″##,###″)　文字列は「1,980」

Now プロパティ　現在の日時を返します

例　日付型 = Now　日付型に現在の日時が入る

Todayプロパティ　今日の日付を返します

例　日付型 = Today　日付型に今日の日付が入る

TimeOfDay プロパティ　現在の時刻だけを返します

例　日付型 = TimeOfDay　日付型に現在の時刻だけが入る

Year関数　日付型から年だけを整数値で返します

例　整数値 = Year(Now)　整数値に現在の年が入ります

Month関数　日付型から月だけを整数値で返します

Day関数　日付型から日だけを整数値で返します

Hour関数　日付型から時だけを整数値で返します

Minute関数　日付型から分だけを整数値で返します

Second関数　日付型から秒だけを整数値で返します

WeekDay関数　指定した日付型から曜日を１～７の整数値で返します

例　整数値 = WeekDay(Now)　整数値に今日の曜日の数値が入ります
日曜日が１、月曜日が２、・・・・土曜日が７になります

WeekDayName関数　１～７の数値を曜日名に変換します

例　文字列 = WeekDayName(3)　文字列は「火曜日」

DateValue関数　文字列を日付型に変換します

例　日付型 = DateValue("2006/9/1")　日付型は2006/9/1

TimeValue関数　文字列を時刻を表す日付型に変換します

例　日付型 = TimeValue("20:15:10")　日付型は20:15:10

DateSerial関数　整数値を日付型に変換します

例　日付型 = DateSerial(2006, 9, 1)　日付型は2006/9/1

TimeSerial関数　整数値を時刻を表す日付型に変換します

例　日付型 = TimeSerial(20, 15, 10)　日付型は20:15:10

DateAdd関数　指定した日付に日数や時間を加算します

例　日付型= DateAdd(DateInterval.Day,10, Now)　今日から10日後
日付型= DateAdd(DateInterval.Minute,20, Now)　今から20分後

DateDff関数　指定した２つの日時の差を任意の単位で計算します

例　長整数型 = DateDff(DateInterval.Day, Now, "2006/1/1")
長整数型は2006/1/1から今日までの日数
長整数型 = DateDff(DateInterval.Hour, Now, "2006/1/1")
長整数型は2006/1/1から今日までの経過時間

(7)　ファイル操作の関数

FileCopy関数　ファイルをコピーします

例　FileCopy("Text.txt", "Text2.txt")

FileDateTime関数　ファイルの更新日時を返します

例　日付型 = FileDateTime("Text.txt")

Kill関数　ファイルを削除します（使い方を誤ると危険な関数です）

例　Kill("*.txt")　ｶﾚﾝﾄﾌｫﾙﾀﾞの拡張子txtの全ﾌｧｲﾙを削除します

Rename関数　ファイルの名前を変更します

例　Rename("C:¥Test.txt","C:¥Test2.txt")　Test.txt→Test2.txt

Dir関数　フォルダやファイル名を順番に取得できます

例　文字列 = Dir("C:¥")　文字列にC:¥の１件目のファイル名

CurDir関数　現在のディレクトリを調べます

例　文字列 = CurDir()　文字列に現在のﾄﾞﾗｲﾌﾞのｶﾚﾝﾄﾃﾞｨﾚｸﾄﾘ

ChDir関数　現在のディレクトリを変更します

例　ChDir(C:¥My Document)

MkDir関数　フォルダを作成します

RmDir関数　フォルダを削除します

注意　削除フォルダ内にファイル等がある場合にはエラーになります

FileOpen関数　ファイルを指定したモードで開きます(VB6からの記述方法)

例　FileOpen(1, "Text.txt", OpenMode.Input)

FileClose関数　閉じるファイル番号を指定します

例　FileClose(1)　１を閉じる、　FileClose()　全てを閉じる

EOF関数　読み込んでいるファイルの最後になったとき知らせます

使用法　Do Until EOF(1)
　　文字列 = LineInput(1)
　Loop

Print関数　ファイルにデータを書き込みます

例　Print(1, "はい！")　ファイル番号１に「はい！」と書き込む

PrintLine関数　ファイルにデータを１行書き込みます

例　Print(1, "はい！")　ファイル番号１に「はい！」と書き込む

Write関数　ファイルにデータを書き込みます

例　Write(2, "へい！")　ファイル番号２に「へい！」と書き込む

WriteLine関数　ファイルにデータを１行書き込みます

例　Write(2, ″へい！″)　ファイル番号２に「へい！」と書き込む

Input関数　ファイルからデータを１つ読み込みます

例　Input(1, 文字列)　ファイル番号１から文字列に読み込む

LineInput関数　ファイルからデータを１行読み込みます

例　文字列 = LineInput(1)　ファイル番号１から文字列に読み込む

(8)　他のアプリケーションの起動

Shell関数　アプリケーションを起動して、番号を取得します

例　数値 = Shell(″mspaint.exe″)　ペイントを起動します

AppActivate関数　アプリケーションウィンドウをアクティブにします

例　AppActivate(数値)　数値で指定したﾌﾟﾛｸﾞﾗﾑをｱｸﾃｨﾌﾞ

２　クラスを利用した文字列処理、型変換、配列操作

(1)　クラスを利用した文字列処理、型変換

Stringクラスを利用した文字列処理では「文字列が０からはじまる」ので注意してください。(文字列関数は「文字列が１からはじまる」)

.IndexOf(検索する文字列)　　文字列を検索して位置を返します

例　文字列 = "abcdefg"
数値 = 文字列.IndexOf("cd")　　数値は 2
数値 = 文字列.IndexOf("xy")　　数値は-1

.Insert(挿入場所,挿入する文字)　　文字列に文字を挿入します

例　文字列A = "abcdefg"
文字列B = 文字列A.Insert(2,"XY")　　文字列Bは「abXYcdefg」

.Length()　　文字列の長さを返します

例　文字列 = "abcde"
数値 = 文字列.Length()　　数値は 5

.Replase(検索する文字、置換する文字)　　文字列の一部を置換

例　文字列A = "abcdefg"
文字列B = 文字列A.Replase("cd","うえ")　　文字列B「abうえefg」

.Substring(開始位置,取り出す文字数)　　文字列を取り出します

例　文字列A = "abcdefg"
文字列B = 文字列A.Substring(3,2)　　文字列Bは「de」

.ToLower()　　大文字を小文字に変換します
.ToUpper()　　小文字を大文字に変換します

例　文字列A = "abcABC"
文字列B = 文字列A.ToLower()　　文字列Bは「abcabc」
文字列C = 文字列A.ToLower()　　文字列Bは「ABCABC」

.ToString　　数値を文字列に変換します

例　数値 = 123
文字列 = 数値.ToString　　文字列は「123」

.Trim()　　文字列の前後の空白文字を削除します

例　文字列A = "　abc　"
文字列B = 文字列A.Trim()　　文字列Bは「abc」

(2)　クラスを利用した配列操作

Array.Clear()　配列の内容を削除します

例　Array.Clear(配列A, 0, 10)　配列Aの0番目から10項目を削除

Array.Copy()　配列の内容をコピーします

例　Array.Copy(配列A, 配列B, 10)

配列Aの10項目を配列Bにコピーします

Array.Sort()　配列の並べ替えをします

例　Array.Sort(配列A)　配列Aの項目が並べ替えされます

Array.IndexOf()　指定した内容と一致する項目の番号を返します

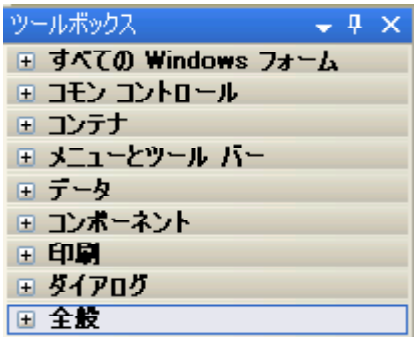
例　整数値 = Array.IndexOf(配列A, 内容)

整数値には一致する番号の数値が入ります

```
Dim A(10) As Integer
Dim B(10) As Integer
Dim D As Long
B = A.Clone()
Array.Clear(A, 0, 10)
Array.Copy(A, B, 10)
Array.Sort(A)
D = Array.IndexOf(A, |
```

▲1 / 3▼ IndexOf (array As System.Array, **value As Object**) As Integer
value: array 内で検索するオブジェクト。

３　コントロール編



コントロール類はツールボックスの中に項目ごとに整理されています。

すべてのWindowsフォームで全コントロールを表示することができますが、通常は、コモンコントロールを開いておけば事足りるでしょう。

ここでは、代表的なコントロールについて、主なプロパティやメソッド、イベントともに簡単に説明します。

(1) コモンコントロール

Button(ボタン)　 

機能　　ボタンです。クリックするとイベントプロシージャを実行します

主なプロパティ　BackColor　ボタンの色、
　　ForeColor　文字色
　　Font　　文字のフォント指定
　　Text　　表示する文字列

主なイベント　　Click　　ボタンをマウスでクリックしたときに発生

CheckBox(チェックボックス)　 

機能　　はい、いいえを表現します

主なプロパティ　Checked　オン・オフの状態を取得、設定します
　　Text　　表示する文字列を設定します

主なイベント　　CheckedChanged　オン・オフが変更されたときに発生

CheckedListBox(チェックドリストボックス)　 

機能　　リストボックスにチェックボックスの機能を追加したものです

主なプロパティ　CheckedItems　チェックされた項目の文字列を取得
　　CheckedItems.Count　チェックされた項目数を取得

主なメソッド　　Items.Add　　リストに項目を追加
　　Items.Clear　全項目を削除
　　SetItemChecked　チェックの有無を指定します

主なイベント　　SelectedIndexChenged　チェックが変更されたとき発生

ComboBox(コンボボックス)　 

機能　　リストからの選択が可能、テキストボックスへの文字の入力も可能

主なプロパティ　Text　　テキスト部分の文字列の取得、設定を行います

主なメソッド　　Items.Add　　リストに項目を追加
　　Items.Clear　全項目を削除

主なイベント　　SelectedIndexChanged　項目の選択が変更されたとき発生

DateTimePicker(デイトタイムピッカ)　 

機能　　日付や時刻を入力するためのボックス

主なプロパティ　Text　　ボックスの内容を文字列型で取得(設定不可)

Value　ボックスの内容を日付型で取得、設定します

主なイベント　　ValueChanged　日時が変更されたとき発生

Label(ラベル)　 

機能　　文字列を表示します（画像も表示できますが・・・）

主なプロパティ　AutoSize　Trueで幅自動調節

BackColor　背景色、Transparentで背景色を透明

ForeColor　文字色

Font　　文字のフォント指定

Text　　表示する文字列

LinkLabel(リンクラベル)　 

機能　　ハイパーリンクをつけた文字列を表示します（画像も・・・）

主なプロパティ　　Labelと同様です

特色的プロパティ　LinkVisited　Trueでリンク先参照済み表示色

主なイベント　　LinkClicked　マウスでクリックしたときに発生

ListBox(リストボックス)　 

機能　　複数の項目のリストから、項目を選択します

主なプロパティ　SelectedItem　選択している項目１つを取得します

SelectedItems　選択している複数項目を取得します

主なメソッド　　Items.Add　　リストに項目を追加します

Items.Clear　　全項目を削除します

SetSelected　　指定した項目を選択状態にします

主なイベント　　SelectedIndexChanged　項目の選択が変更されたとき発生

ListView(リストビュー)　 

機能　　項目をアイコン表示したり、詳細表示をできます

主なプロパティ　SelectedItems　選択している項目リストを取得します

View　「大きいｱｲｺﾝ」「小さいｱｲｺﾝ」「リスト」「詳細表示」の４つの表示形式を選択します

主なメソッド　　Items.Add　　リストに項目を追加します

Items.Clear　　全項目を削除します

主なイベント　　SelectedIndexChanged　項目の選択が変更されたとき発生

MonthCalender(マンスカレンダー)

機能　　カレンダー上で日付を選択できます

主なプロパティ　MaxDate　　選択可能な最大の日付を設定します

Selection.Range.Start　選択した日付の開始日

Selection.Range.End　　選択した日付の終了日

主なイベント　　DateChanged　選択した日付が変更されるたびに発生します

NotifyIcon(ノティファイアイコン)

機能　　タスクバーの通知領域（右側の部分）にアイコンを表示します

主なプロパティ　ContextMenu　右クリックで表示するメニューを指定

Icon　　表示するアイコンファイルの指定

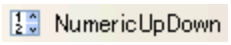
Text　　マウスポインタを置いたときに表示する文字列

Visible　Trueでアイコンを表示

主なイベント　　Click　　アイコンをクリックしたとき発生

DoubleClick　アイコンをダブルクリックしたとき発生

NumericUpDown(ヌメリックアップダウン)



機能　　数値を入力するためのボックスです

主なプロパティ　Increment　上下の矢印ボタンで増減する値の指定

Minimum　　入力可能な最小値（初期値　0）

Maximum　　入力可能な最大値（初期値100）

Value　　数値の取得、設定をします

PictureBox(ピクチャーボックス)

機能　　画像や図形を表示することができます

主なプロパティ　Image　表示する画像を指定します

Nothing　　画像を消します（指定なし）

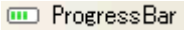
SizeMode　表示のしかたを指定します

Normal　　等倍表示

StrechImage　画像をｺﾝﾄﾛｰﾙのｻｲｽﾞに合わせます

AutoSize　　ｺﾝﾄﾛｰﾙを画像のｻｲｽﾞに合わせます

ProgressBar(プログレスバー)



機能　　処理の進行状況を水平グラフで表します

主なプロパティ　Maximum　　プログレスバーの最大値（初期値100）

Minimum　　プログレスバーの最小値（初期値　0）

Value　　プログレスバーの現在の値の指定

主なイベント　　Click　　プログレスバーをクリックしたときに発生

RadioButton(ラジオボタン)

機能　　複数の選択肢から１つを選ぶときに利用します

主なプロパティ　Appearance　Buttonにするとボタン状の表示になります

Checked　　項目が選択されているかどうかを取得、設定

Text　　表示する文字列を設定します

主なイベント　　CheckedChanged　ラジオボタンの状態が変更されたとき発生

RichTextBox(リッチテキストボックス)

機能　　フォントの色や大きさを指定できるテキストボックスです

主なプロパティ　TextBoxにさらに機能が加わります

SelectionColor　選択した文字列の色を指定します

SelectionFont　　選択した文字列のフォントを指定します

TextBox(テキストボックス)

機能　　文字列の入力が可能です

主なプロパティ　ImeMode　　IME(日本語入力)のモードを設定します

MultiLine　Trueで複数行入力可能、初期設定は１行だけ

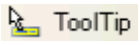
ScrollBars　複数行入力時のスクロールバーの表示

Text　　文字列の表示

WordWrap　　Trueで複数行入力時、右側で折り返し

主なイベント　　KeyPress　　キーを押したとき

ToolTip(ツールチップ)



機能　　マウスポインタを置いたときヒントを表示する機能を与えます

主なプロパティ　Active　　Trueでツールヒントを表示します

主なメソッド　　SetToolTip　コントロールにツールヒントを設定

TreeView(ツリービュー)

機能　　階層関係にある情報を表示します。展開表示、省略表示できます

主なプロパティ　Nodes　　ノードを指定します

ShowLines　　ノード間をつなぐ点線の表示の有無の指定

主なメソッド　　Nodes.Add　　ノードを追加します

Nodes.Clear　全ノードを削除します

Nodes.Insert　指定した位置にノードを挿入します

Nodes.RemoveAt　　指定したノードを削除します

主なイベント　　AfterSelect　ノードを選択し終えた時に発生します

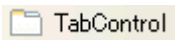
(2) コンテナ

GroupBox(グループボックス)　 

機能　コントロールをグループ化します。タイトル表示ができます
主なプロパティ　Enabled　Falseでコントロールの使用が不可になります
Text　タイトルの文字列を設定します

Panel(パネル)　 

機能　コントロールをグループ化します。タイトル表示は不可
主なプロパティ　Enabled　Falseでコントロールの使用が不可になります
AutoScroll　Trueで自動でスクロールバーを表示します

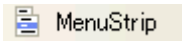
TabControl(タブコントロール)　 

機能　タブページを使って限られた面積で多くの情報を提示できます
主なプロパティ　TabPages　タブページの詳細を設定します
TabPageのText　タブに表示する文字列を設定します
※　TabPageコレクションエディタから詳細設定します

(3) メニューとツールバー

ContexMenuStrip(コンテキストメニュー)　 

機能　右クリックで表示されるメニューを設定します
主なプロパティ　MenuItems.Text　表示する文字列を設定します
MenuItems.Visible　Falseでメニューが非表示になります
主なメソッド　MenuItems.Add　メニューを追加します
MenuItems.Clear　全メニュー項目を削除します
MenuItems.RemoveAt　指定したメニュー項目を削除します
主なイベント　Popup　メニューが表示されたとき発生します

MenuStrip(メインメニュー)　 

機能　フォーム上のメニューバーを設定します
主なプロパティ　MenuItems.Enabled　Falseにすると選択不可になります
MenuItems.Text　表示する文字列を設定します
主なメソッド　MenuItems.Add　メニューを追加します
MenuItems.Clear　全メニュー項目を削除します
MenuItems.RemoveAt　指定したメニュー項目を削除します

(4) コンポーネント

Timer(タイマー)　 

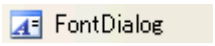
機能　一定時間ごとにイベントを発生します
主なイベント　Tick　一定時間ごとに発生するイベントです

主なプロパティ　Enabled　Trueでタイマー稼働、Falseで停止
Interval　ミリ秒単位でイベント発生間隔時間を設定
主なメソッド　Start　タイマー稼働、EnabledがTrueになります
Stop　タイマー停止、EnabledがFalseになります

(5) ダイアログ

ColorDialog(カラーダイアログ) 

機能　Windows標準のダイアログで色を選択できます
主なプロパティ　Color　ダイアログで選択した色を参照できます
主なメソッド　ShowDialog　ダイアログを表示します

FontDialog(フォントダイアログ) 

機能　フォントのスタイル、サイズ、色を選択できます
主なプロパティ　Font　フォントを設定、参照します
主なメソッド　ShowDialog　ダイアログを表示します
主なイベント　Apply　適用ボタンが押されたときに発生

OpneFileDialog(オープンファイルダイアログ) 

機能　Windows標準のファイルを開くダイアログを利用できます
主なプロパティ　FileName　選択したファイル名（パス付き）
FileNames　複数選択したときの全てのファイル情報
Filter　表示するファイルの種類の指定
MultiSelect　Trueで複数選択可
主なメソッド　ShowDialog　ダイアログを表示します
主なイベント　FileOk　開くボタンをクリックしたとき発生

SaveFileDialog(セーブファイルダイアログ) 

機能　Windows標準の名前を付けて保存ダイアログを利用できます
主なプロパティ　Filter　表示するファイルの種類の指定
OverwritePrompt　Trueで既存ファイルへの上書き確認
主なメソッド　ShowDialog　ダイアログを表示します
主なイベント　FileOk　保存ボタンをクリックしたとき発生

(6) すべてのWindowsフォーム

TrackBar(トラックバー) 

機能　スライダー（つまみ）の移動で数値の設定します
主なプロパティ　LargeChange　軸をクリックしたときの変化量
Maximum　最大値（規定値10）
Minimum　最小値（規定値 0）

	Orientation	Horizonalで横型、Verticalで縦型
	SmallChange	スライダーをドラッグしたときの変化量
	Value	現在の値の取得や設定
主なイベント	Scroll	スライダーが移動したとき発生

第６章　Visual Basic 2005 Express Editionのインストール方法

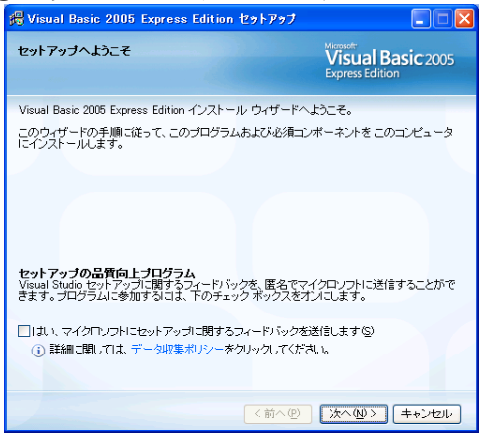
１　Visual Basic 2005 Express Editionの入手方法

MicrosoftのWebサイトからダウンロードしてください。

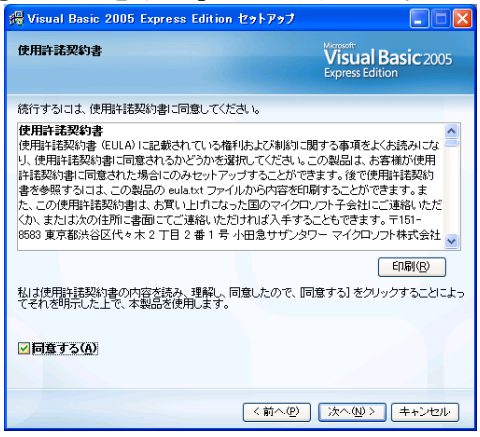
http://www.microsoft.com/japan/msdn/vstudio/express/

２　Visual Basic 2005 Express Editionのインストール方法

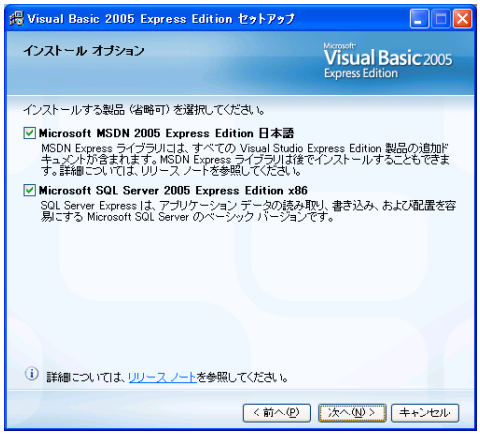
①　次へをクリックします



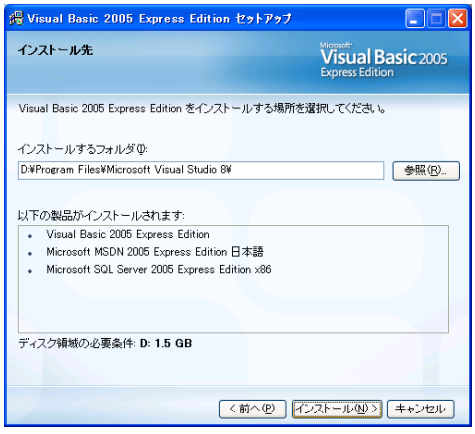
②　「同意する」にチェックし次へ



③　次へをクリックします

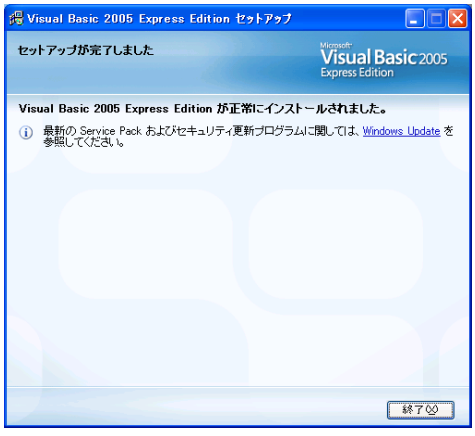


④　インストールが開始されます



⑤　20～30分かかります

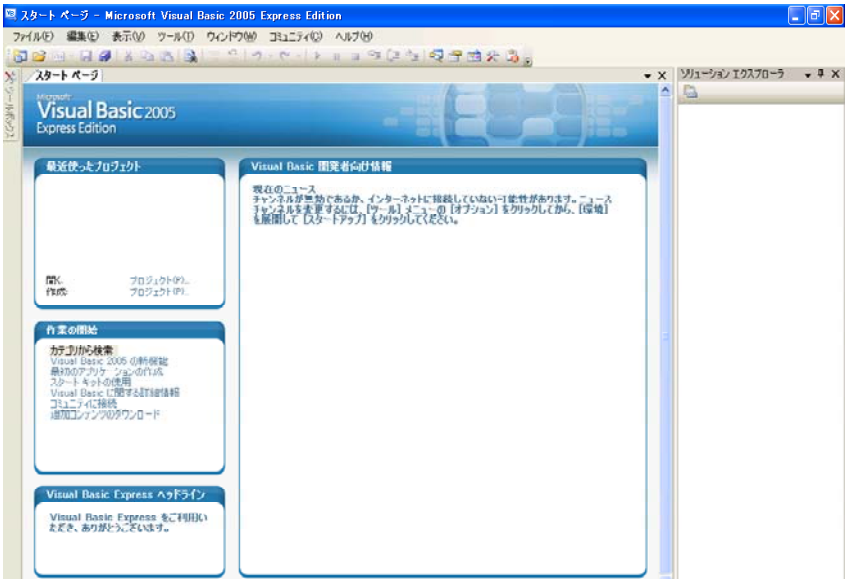
⑥　セットアップの終了です



⑦　スタートボタンから「すべてのプログラム」→「Microsoft Visual Basic 2005 Express Edition」を選択します

⑧　初めて起動したときだけ右図の画面が表示されます。

⑨　Microsoft Visual Basic 2005 Express Edition が起動します。



第７章　Visual Basicのプログラミングの参考になるWebサイト

１　Visual Basicについての最新の情報を入手できるWebサイト

(1)　MicroSoft Visual Studio 2005 Express Edition

(http://www.microsoft.com/japan/msdn/vstudio/express/)

マイクロソフト社のWebサイトです。VB2005の最新情については、ここからダウンロードすることができます。

２　Visual Basicを使った教材があるWebサイト

(1)　突ちゃんのHomePage(http://www.eonet.ne.jp/~y2kuda/)

.NETで作成した中学校理科の教材や、VBAで作成したExcelの中学校校務処理ソフトがあります。

(2)　Vector(http://www.vector.co.jp/vpack/filearea/win/edu/)

日本で一番、教材が豊富なWebサイトです。ドリル学習教材や、提示用教材、シミュレーション教材などたくさんあります。VB.NETやVB2005で作られた教材も増えてきています。

3　Visual Basicの基本について学ぶことができるWebサイト

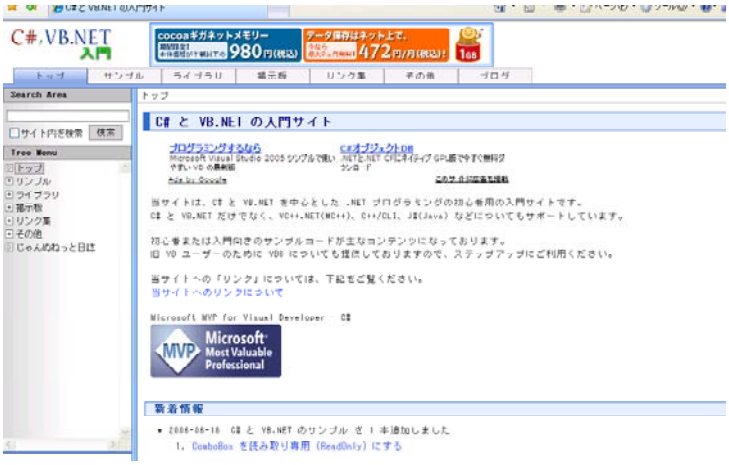
(1) Visual Basic 中学校 (http://homepage1.nifty.com/rucio/main/main.htm)

初心者や中学生、高校生向けにVisual Basicについて解説しているWebページです。VB2005についても対応しています。

(3) VB.NET初心者の館 (http://vbnet-iku2.hp.infoseek.co.jp/)

VB.NETについてのWebサイトです。「.NETとは」「プロパティについて」のなどのプログラムとVB.NETに初めてふれる本当の初心者向けに書かれています。残念ながら、VB.NET2005についてはふれられていません。

(2) C#,VB.NET入門 (http://jeanne.wankuma.com/)



題名には「入門」なっていますが、VBの使い方や、用語が分かっている人のためのWebサイトです。「これはどうすればいいのかな？」と迷ったときに参考になるページがあります。

4　Visual Basicについて、さらに高度な技術を学べるWebサイト

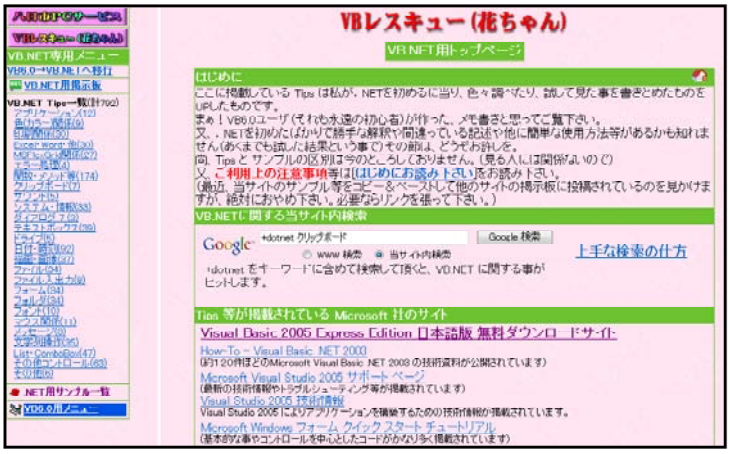
(1)　DOBON.NET（http://dobon.net/）

VBやC#など、.NETでどうすればよいか困ったときに参考になるWebサイトです。特に、画像の表示や、グラフィック関係について、詳しい説明と、VBとC#のサンプルプログラムが載っています。

Dobo

(2)　VBレスキュー（花ちゃん）　VB.NET用トップページ

（http://hanatyan.sakura.ne.jp/dotnet/index.html）



VBで作るとき，「どうすればいいのか？」と迷ったときのコードの書き方が載っています。

(3)　VB.NETで作る！（http://shinshu.fm/MHz/88.44/）

VB.NETのTips、フリーソフトが公開されています。