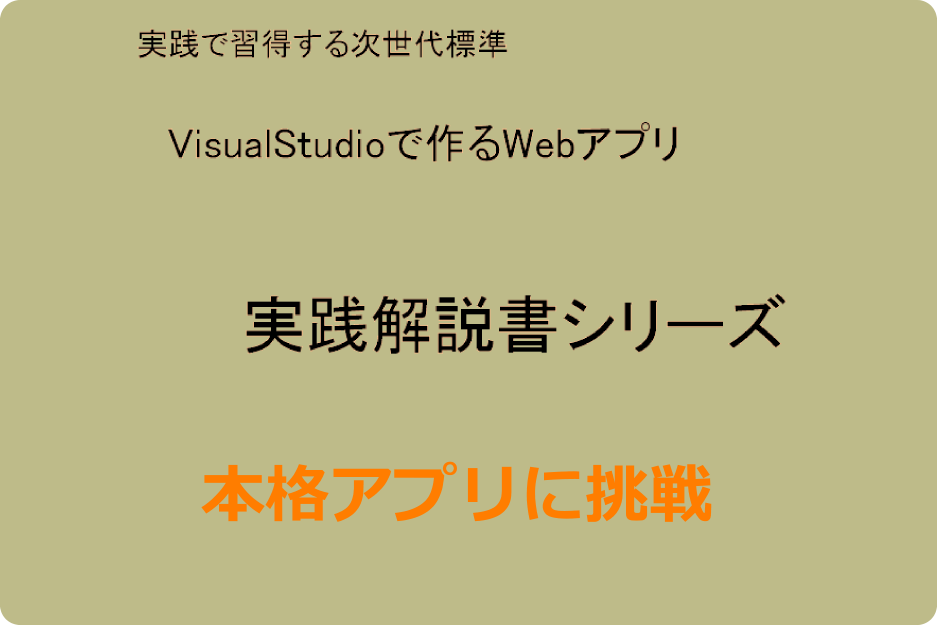
|  |
| --- |
| 本格的なアプリ開発を経験 |
| データベースとVSのサーバーコントロールを使用 |
| いよいよ、本科的なWebアプリを開発します。この章を終えればもう立派なプログラマーになっています。この章を足掛かりにして更にスキルアップ、プロフェッショナルなWebサイト構築の世界に入ります。 |
|  |



目次

[本格的なアプリ開発を経験 0](#_Toc6241228)

[初めに 6](#_Toc6241229)

[準備編 7](#_Toc6241230)

[この解説書で学ぶ事 7](#_Toc6241231)

[VSのドキュメントウインドウとは？ 9](#_Toc6241232)

[基本の基本、３つのキーワード 12](#_Toc6241233)

[オブジェクトって何だろう？ 13](#_Toc6241234)

[オブジェクトとプロパティの関係 13](#_Toc6241235)

[データ型とは 14](#_Toc6241236)

[変数とは？を深く理解する 16](#_Toc6241237)

[変数を宣言してデータを格納（Ｄｉｍステートメント） 16](#_Toc6241238)

[新規にWebサイトを作成 18](#_Toc6241239)

[テストページで基礎を極める 22](#_Toc6241240)

[サイト内に新規にWebページを作成する 22](#_Toc6241241)

[変数のスコープ 35](#_Toc6241242)

[変数のページ内のスコープの実験 36](#_Toc6241243)

[ToString（）とは 39](#_Toc6241244)

[変数の使用法について 39](#_Toc6241245)

[1. 配列：複数の値を表す変数 40](#_Toc6241246)

[TestHairetsu.aspxページを作成 41](#_Toc6241247)

[実行画面での、文字列の表示方法 48](#_Toc6241248)

[2. プロシージャ（関数）とは？ 49](#_Toc6241249)

[htmlコードとプログラム言語の関係 51](#_Toc6241250)

[プログラマーなら“CSS”を理解しよう 52](#_Toc6241251)

[マスターページの構造 55](#_Toc6241252)

[メニュー表示部分 55](#_Toc6241253)

[ページの表示部分とフッター部分 56](#_Toc6241254)

[HTMLとCSSの関係 57](#_Toc6241255)

[スタイルの定義位置の検索の仕方 58](#_Toc6241256)

[HTMLの<a>~</a>要素を理解する 61](#_Toc6241257)

[データベースの重要性 62](#_Toc6241258)

[データベースを理解する 63](#_Toc6241259)

[表 63](#_Toc6241260)

[主キー 63](#_Toc6241261)

[サイトにデータベースを構築する 65](#_Toc6241262)

[データベースのデータ型 69](#_Toc6241263)

[テーブルの作成方法 71](#_Toc6241264)

[テーブルのプロパティの設定 74](#_Toc6241265)

[マスターページの編集 78](#_Toc6241266)

[マスターページのメニューに項目を追加 78](#_Toc6241267)

[マスターページのフッターを編集 79](#_Toc6241268)

[アプリケーション名の編集 80](#_Toc6241269)

[本物のWebアプリケーション 81](#_Toc6241270)

[サイト内に新しいWebページ（フォーム）を作成する 81](#_Toc6241271)

[マスターページとの親子関係 83](#_Toc6241272)

[ページつくりの基礎の基礎 85](#_Toc6241273)

[<div の背景色をVS側で変更する 87](#_Toc6241274)

[Webサーバーコントロールの配置 91](#_Toc6241275)

[コントロール間に隙間を空ける 91](#_Toc6241276)

[同一ページ内で列（カラム）数を変える 93](#_Toc6241277)

[ページの列構成を理解する 97](#_Toc6241278)

[複数のコントロールはテーブルで配置 98](#_Toc6241279)

[テーブルのプロパティ設定でこんなに変わる 111](#_Toc6241280)

[プロフェッショナルなプログラムする 115](#_Toc6241281)

[パネルコントロールでユーザーを導く 115](#_Toc6241282)

[カレンダーコントロールの日付処理 118](#_Toc6241283)

[データベースを実用化する 121](#_Toc6241284)

[Dataコントロールを使う 124](#_Toc6241285)

[Dataコントロールとは 124](#_Toc6241286)

[DropDownListBoxにデータソースを関連つける 124](#_Toc6241287)

[DropDownListコントロールに“選択して下さい”を表示する 132](#_Toc6241288)

[ListBoxにデータソースを関連つける 134](#_Toc6241289)

[新しいデータベースのテーブルの作成 134](#_Toc6241290)

[ListBoxコントロールにデータソースを関連つける 141](#_Toc6241291)

[データベーステーブルに手入力でデータを入れる 142](#_Toc6241292)

[RadioButtonListのItem設定 143](#_Toc6241293)

[入力必須の処理、検証コントロール 145](#_Toc6241294)

[PanelDate内の検証コントロールのプロパティ表 148](#_Toc6241295)

[CausesValidationとは？ 151](#_Toc6241296)

[CompareValidatorのTypeプロパティ 153](#_Toc6241297)

[PanelStatus内の検証コントロール 159](#_Toc6241298)

[もう一度!!コンピュータープログラムって難しい？ 164](#_Toc6241299)

[「登録」ボタンクリックイベントのプログラム（１） 166](#_Toc6241300)

[スケジュール（予定表）を保存する 169](#_Toc6241301)

[スケジュールデータ登録用のデータベーステーブルの作成 170](#_Toc6241302)

[ページを追加デザインする 171](#_Toc6241303)

[データソースの構成 174](#_Toc6241304)

[データコントロール（GridView）とデータソースを関連つける 178](#_Toc6241305)

[GridViewの日付表示を日本式の表示にする 182](#_Toc6241306)

[テーブル内のデータの操作 183](#_Toc6241307)

[行の参照：Selectステートメント 183](#_Toc6241308)

[行の挿入：INSERTステートメント 185](#_Toc6241309)

[行の更新：UPDATEステートメント 186](#_Toc6241310)

[行の削除：DELETEステートメント 187](#_Toc6241311)

[データソース、SqlDataSourceコントロールの中身 188](#_Toc6241312)

[Selectコマンド（Select文） 188](#_Toc6241313)

[Insertコマンド 189](#_Toc6241314)

[Deleteコマンド 190](#_Toc6241315)

[Updateコマンド 191](#_Toc6241316)

[「登録」ボタンのイベントの処理（データを登録） 193](#_Toc6241317)

[「コマンド及びパラメータのエディター」で編集 195](#_Toc6241318)

[「コマンド及びパラメータのエディター」で値を設定 197](#_Toc6241319)

[「登録」ボタン、クリックイベントに、データ登録のコードを追加 200](#_Toc6241320)

[簡単なメニュー機能を作成 203](#_Toc6241321)

[Bootstrap-のボタンのデザインを利用する 204](#_Toc6241322)

[イベントハンドラーの作成方法 209](#_Toc6241323)

[データベーステーブルにデータ挿入時のイベント 209](#_Toc6241324)

[登録したデータを変更する 212](#_Toc6241325)

[GridViewのデータを編集する 212](#_Toc6241326)

[デザインビューで、各テンプレートを編集する 217](#_Toc6241327)

[編集インターフェイスでの代替 Web コントロールの使用 221](#_Toc6241328)

[データベーステーブルのデータ行を削除する 222](#_Toc6241329)

[GridViewの編集機能をテスト表示する 222](#_Toc6241330)

[日付のデータ列の編集インターフェースをカレンダーコントロールにする 223](#_Toc6241331)

[状態の列編集インターフェースをドロップダウンリストにする 232](#_Toc6241332)

[優先度列の編集インターフェースをRatioButtonListにする 233](#_Toc6241333)

[区別列の編集インターフェースをListBoxにする 235](#_Toc6241334)

[メモ列の編集インターフェース「TextBox」プロパティの変更 236](#_Toc6241335)

[登録したデータを削除する。 238](#_Toc6241336)

[スケジュール作成アプリのユーザビリティをアップする 244](#_Toc6241337)

[付録 247](#_Toc6241338)

[**共通する設定** 247](#_Toc6241339)

[VisualStudioのテーブルとその配置と使用法 248](#_Toc6241340)

[VisualStudio のテーブルとは、下図の位置のメニューにある項目の事を指します。注：デザインビューでの作業中に表示されます。 248](#_Toc6241341)

[あとがき 261](#_Toc6241342)

# 初めに

㈱スカイオーシャンの無料の解説書では、全くの初心者の方がスムーズにコンピュータプログラミングの世界に入れるように、開発ツールであるVisualStudio２０１７の使い方に慣れることに力点を置いて、解説してありました。全くの初心者の方が㈱スカイオーシャンの無料でダウンロードした、実践解説書を複数回実践されて開発ドキュメント画面に慣れ、イベントハンドラーの意味を掌握されていることを前提に本チュートリアルを進めていきます。

本チュートリアルの前半の「準備編」をしっかりと実践されることを願っています。

何かを記憶しなければならないことはありません、経験するだけです。Htmlコード及びプログラムコードは、解説書内のコードをコピーアンドペーストしてプログラムを実行するように構成してあります。この方法により読者様（実践者）は、Webアプリケーションのプログラム知識の習得をストレスなしで継続する事が出来て、結果的に短期間でWebプログラムのノウハウを習得することが出来ます。

使用するプログラム言語は、**VisualBasic**　です。開発ツールは最新のASP.NETのFramework機能を最大限利用できるVisualStudio Coomunity 2017を無料でダウンロードした開発ツールを使用します。

尚、この解説書のレイアウトについては、この解説書をPCのモニターで表示しながら実践される事を想定していますので空白部分が多かったり、不揃いの図形が並んだりしますが、プログラムコードなどをコピペする便利さを想定していますので、ご容赦下さい。またこの解説書をWordファイルまたはｐｄｆファイルで提供させていただいたのは、読者様自身が追加等の記事を追加、保存等も自由に出来るのではないかと思った故です、どうぞ便利に使用してください。

# 準備編

## この解説書で学ぶ事

プログラマー必須の基礎知識を早く身につけて、どんどんステップアップのスピードをあげて行きましょう。

これからしばらくは、Webプログラマーとして必須の基礎知識を身につけていきます。

下記の内容を解説します。

1. VisuaStudioの開発環境のドキュメントウインドウの配置
2. オブジェクトとプロパティ
3. データ型とは？
4. 変数、そして変数のスコープ
5. 新規にWebサイト作成

変数、そして変数のスコープ、データ型等の実践ページ作成

1. データの内容を表示する“ToStirng”
2. 変数の配列

配列を理解するための実践ページ作成

1. 関数（メソッド）
2. Htmlコードとプログラムの関係
3. HTMLとCSS
4. データベースを理解する

プロジェクト（Webサイト）内にデータベースを構築する

1. マスターページと各ページの関係（親子関係）を理解する

上記の内容を解説、実践します。その経験を基礎として本物のWebアプリを作成します。

新たに、数多くのことを憶えることではありません。経験をするつもりで臨んで下さい。

そして、あなたのプログラムスキルが上がっても、何度も開発途中でこのチュートリアルを参照して下さい。

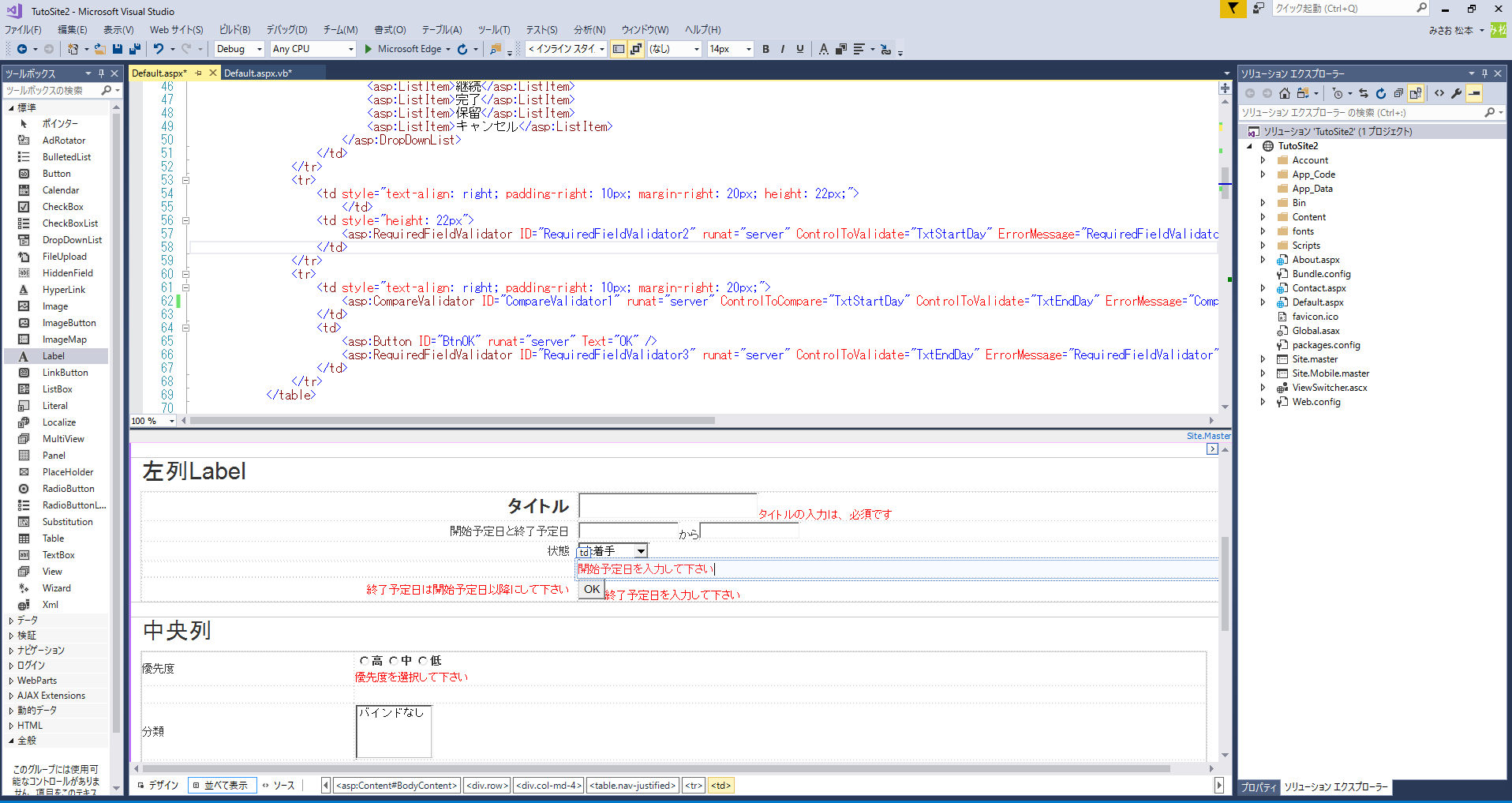
わが社の無料でダウンロードした、「VisualStudioに慣れる(1),又はVisualStudioに慣れる(2)　の内容を理解されている方を対象に本解説を進めて行きます、VisualStudio２０１７を立ち上げた状態でこれ以後は本解説を進めて下さい。

## VSのドキュメントウインドウとは？

編集中の、Webページや、ソースコード（htmlコード）を表示します。下図のような

各ウインドウの配置で、本チュートリアルを進めて行きます。（今の時点では、コードビューとデザインビューの位置を確認する程度にして下さい。）

解説を実践中に、その都度参照して下さい。



表示

**コードビュー**

**デザインビュー**

**３**

**２**

**１**

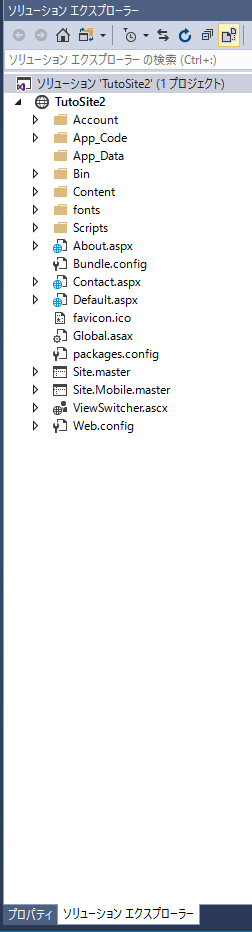
**ソリューションエクスプローラと**

**プロパティウインドウ**

**ツールボックス**r

1. “ソース、デザイン、並べて表示”等のタブが容易されています。
2. 左側に“ツールボックス”
3. 右側に“ソリューションエクスプローラ”と“プロパティウインドウ”
4. 中央の下が、“デザインビュー”、上が“コードビュー”です。

上図のドキュメントウインドウ内の“ツールボックス”、“ソリューションエクスプローラ”、“プロパティウインドウ”等は、ドキュメントウインドウの最上部のメニュー“表示”をクリックすると、各ウインドウの項目が表示されて、表示したいウインドウが選択できます。各ウインドウは、フローティングウインドウになっていて、マウスで自由に動かせますので、前図のように、配置して下さい。ソリューションエクスプローラとプロパティウインドウを配置した状態を次図に表示しておきます。



上の図の下部の赤い四角の部分は、次図のようにタブで選択できるようになっています。



両者共、フローティングウインドウですので、数回練習が必要です。私の場合、何回もメニューの“表示”からこのフローティングの表示を繰り返して、フローティングウインドウを使用する練習をした経験を思い出します。

## 基本の基本、３つのキーワード

この章では、いよいよ、コンピュータ―プログラマーとして最初に身に着けなればならない知識を少しづつ掲載していきます。

まず、次の３つのキーワードに慣れましょう。

1. オブジェクト
2. データ型
3. 変数

### オブジェクトって何だろう？

### オブジェクトとプロパティの関係

オブジェクトとは、物、目標物、対象という意味の英単語です。

例えば、“靴”という“物”を例にします、その靴をコンピュータープログラムの世界では「“靴”というオブジェクトとして取り扱います」、靴全体に、赤い色を付けて表示するためには、「“靴の全体のColor“を“赤”にします」、靴底の色を青色にして表示する場合は「“靴の底のColor”を“青”にします」、また靴ひもを付ける場合は「“靴ひも“を”あり“にします」。

上記のように、“靴の全体のColor”、“靴の底のColor”、“靴ひも”等のことを「プロパティ名」といいます、また、“赤”、“青”、”あり“等のことをプロパティの値と呼びます。

プログラマーの初心者を抜け出した時には、下記のような正しい解説が理解出来ます。

**オブジェクトとは**

ソフトウェアのユーザインターフェースなどでは、ユーザーの操作対象として列挙されたアイテム(指定されたフォルダに含まれるファイルなど)のことをオブジェクトとして、指すことが多い。

オブジェクト指向プログラミングにおいては、ソフトウェアが扱おうとしている現実世界に存在する物理的あるいは抽象的な実体を、属性(データ)と操作(メソッド)の集合としてモデル化し、コンピュータ上に再現したもの。オブジェクトを定義するモデルはクラスと呼ばれる。クラスに基づいて実際にコンピュータのメモリ上に展開されたオブジェクトのことをインスタンス(実体)と言うが、このインスタンスの意味で、オブジェクトと呼ぶ場合も多い。

### データ型とは

　ここで、次に進む前に抑えておきたい大事な事を眺めておきましょう。

データ型とは、データの種類とサイズのことです。たとえば、商品名などの文字列を扱うときはString型、小数点以下の数値で計算するときはSingle型など、そのデータの種類やサイズによって使用する変数（データを格納するための容量を持つ入れ物）のデータ型を宣言します。プログラムでは、様々な種類のデータを使用します。それらのデータを変数に記憶させて効率的にメモリを管理するために、VisualBasic .NETでは、データを変数に格納するときにあらかじめ用意されたデータ型も宣言します。

次表に一覧表を作成しました。

データ型の種類（参照用として、利用して下さい）

|  |  |  |
| --- | --- | --- |
| **データ型** | **サイズ** | **値の範囲** |
| 短整数型（Short） | ２バイト | -32,768～32,767 |
| 整数型（Integer） | 4バイト | -2,147,483,648～2,147,483,647 |
| 長整数型（Long） | 8バイト | -9,223,372,036,854,775,808  ～9,223,372,036,854,775,807 |
| 単精度浮動小数点数型（Single） | 4バイト | 負の値:  -3.4028235E+38～-1.401298E-45  正の値:  1.401298E-45～3.4028235E+38 |
| バイ精度浮動小数点数型（Double） | 8バイト | 負の値:  ･４.797693.134862315=7=OE÷308  ～-4.94065645841246544E-324  正の値:  ４,94065646841246544=Ｅ-324  ～1.79769313486231570E+308 |
| １０進数型（Decimal） | 16バイト | 整数の場合:  ･79､228,162,514,264,337,593,543,950,335  ~79,228,162,514,264,337,593､543,950,335  小数点以下28桁の場合:  -7.9228162514264337593543950335  ～7.9228162514264337593543950335  0を除いた場合の絶対値の最小値:  0.0000000000000000000000000001(IE-28) |
| バイト型（Byte） | 1バイト | O～255 (符号なし) |
| Char型（Char） | 2バイト | O～65535(符号なし) |
| 文字列型（String） | 格納サイズにより可変 | O～約20億個のUnicode文字 |
| ブール型（Bool） | 2バイト | True(真)またはFalse(偽) |
| 日付型（Date） | 8バイト | 0001年1月1日0：00：00～9999年12月31日 |
| オブジェクト型（Object） | 4バイト | 任意の型を格納出来る |
| ユーザー定義型（構造体） | 格納サイズにより可変 | 実装するプラットフォームに依存 |

難しいようなことが書いてありますが、眺めて置く程度で充分です。

私の場合上表を印刷して、参照用に壁に貼っていました。

## 変数とは？を深く理解する

### 変数を宣言してデータを格納（Ｄｉｍステートメント）

　変数とは、データを一時的に格納しておくための入れ物で、プログラム実行中に変数に値を代入したり、値を変更したりすることができます。　変数を利用するには、変数の名前を宣言する必要があります。変数の宣言は、下記のように記述します。

　Dim　変数名　Ａｓ　データ型

　１つのDimステートメントで、複数の変数を宣言できます。その場合、カンマ「､」で区切って入力します。変数を宣言したら、その変数に値を代入しますが、変数の宣言と同時に値の代入を行うこともできます。

データ型一覧表

プログラムコード例

プログラムコード例 1

|  |
| --- |
| ‘文字列型変数ａを宣言する  　　　　　Dim ａ As String  　　　　　　･整数型変数ｂ．ｃを宣言する  　　　　　　　　Dim b, c As Integer  　　　　　　　　‘長整数型変数ｆと、単精度浮動小数点型変数ｈを宣言する  　　　　　　　　Dim f As Long, h As Single  ・値の代入  a = "test"  b = 365  '文字列型変数ｇの宣言と値の代入を同時に行う  Dim g As String = “test2'” |

それでは実践を通して理解度を深めていきます。

# 新規にWebサイトを作成

それでは実際にVisualStudioを使用して見ましょう。

VisualStudioを起動します、下図

|  |
| --- |
|  |

下図は、上図の左上のメニュー「ファイル」をクリック、「新規作成」をクリックした時の画面です

|  |
| --- |
|  |

上図の右側のサブメニューの「プロジェクト」を選択します。次図が表示されます。

|  |
| --- |
| 1  3  2 |

上図が表示されたら、１の「WebSite」を選択します、次に、２の「ASP.NET Webフォームサイト」を選択して、３の名前の入力欄に「TimeTable」と入力して、画面右下の「OK」ボタンをクリックします。

次図のように「Default.aspx」ページのhtmlコードビューと右側にはソリューションエクスプローラが表示されます。

|  |
| --- |
|  |

## 最新のVisualStudio２０１７でのプロジェクト作成手順

VisualStudio２０１７のメニュー「新規作成」－＞「プロジェクト」と進みます、下図

|  |
| --- |
|  |

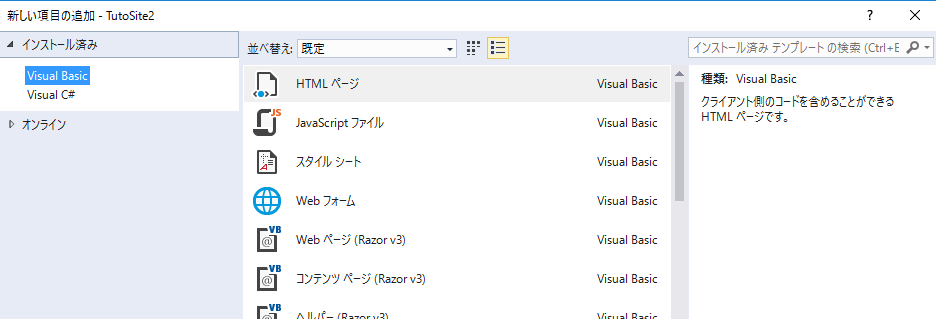
上図の「プロジェクト」を選択すると、下図が表示されます、

|  |  |
| --- | --- |
|  | 旧バージョンのVisualStudioと少し違うところは、赤い四角で囲んだ部分の“以前のバージョン」という項目が追加されています。  本解説書を実践する場合は、この“以前のバージョン”を選択して、次に右の欄の「ASP.NET　Webフォームサイト」を選択します。  その後の、「名前」、「ソリューション名」等の設定の仕方は、旧バージョンと同じ要領で設定して下さい。  尚、旧、新バージョン共、開発に関する機能は特別変わりはないようです。 |

## テストページで基礎を極める

## サイト内に新規にWebページを作成する

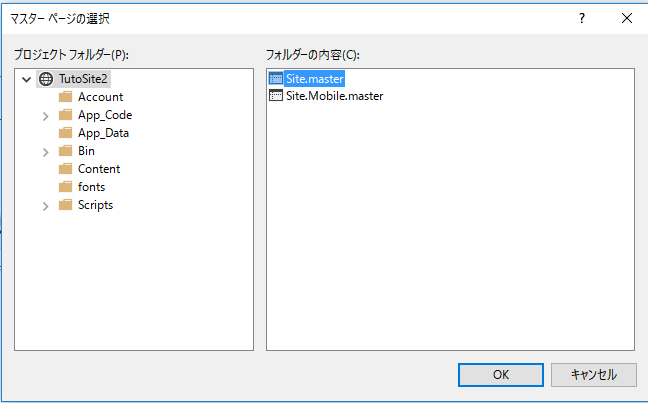
ソリューションエクスプローラの“サイト名”を右クリックします。表示されたドロップダウンメニューの“追加”という項目にマウスを載せます、続いて表示されるメニューの一番上の“新しい項目の追加”というメニューをクリックします。下図が表示されます。



上図の左列の“VisualBasic”を選択して、上図の中央付近の”Webフォーム”を選択して、下図の“名前”の入力欄に“TestHensu”と入力して、その入力欄の右側に“別のファイルにコードを書き込む”と“マスターページの選択”両方にチェックを入れて、“追加”ボタンをクリックします。下図



次に表示される、“マスターページの選択”のダイアログウインドウで、右側の欄の“Sitemaster”を選択します。次図



“OK”ボタンをクリックします。

すると、ソリューションエクスプローラの主ルートに“TestHensu.aspx”ファイルが追加されています。下図

|  |
| --- |
|  |

“TestHensu.aspx”の左の右向き矢印をクリック（展開）すると、下図のように”TextHensu.aspx.vb”ファイルも自動で追加されています。

|  |
| --- |
|  |

ドキュメントウインドウでは、コードビューが表示されていると思います。そのコードビュー内の下記のコードは“Content”タグで括られています。このContentタグのhtmlコードの内容が、デザインビューに反映（表示）されるのです。

下記のように、このページは、２列に表示するようにBootstrapのコードを入力します。

赤色のコードを“Content1”内にコピペして下さい。

|  |
| --- |
| <asp:Content ID="Content1" ContentPlaceHolderID="MainContent" Runat="Server">  　　　<div class=”row”>  <div class=”col-md-6”>  <h2>左列</h2>  </div>  <div class=”col-md-6”>  <h2>右列</h2>  </div>  </div>  </asp:Content> |

Htmlコードビューの任意の位置で、マウスの右クリックして、メニューの「ブラウザーで表示」をクリックして、実行して下さい。下図のようになっています。



実行画面を閉じてください。

次に、コードビュー及びデザインビューを表示して、どちらかの任意の場所を右クリックして、“コードの表示”を選択して、プログラムコード画面上部の中央列の“（TestHensu イベント）”をクリックします。

|  |
| --- |
|  |

次に、右側の“Disposed”と表示されている、ドロップダウンを開きます、下図

|  |
| --- |
|  |

次に上図中央付近の「Load」というイベント名を選択して下さい。

選択と同時にプログラムコード内に下表のようなイベントハンドラーが表示されます。

|  |
| --- |
|  |

このイベントは「TestHensu.aspx」ページが表示される時に発生するイベント名です。

作成した、“TestHensu\_Load”プロシージャ内に下記のコードを入力します。

赤いコード部分コピペ可能

|  |
| --- |
| Private Sub TestHensu\_Load(sender As Object, e As EventArgs) Handles Me.Load  '文字列型変数ａを宣言する  Dim a As String  '整数型変数ｂ.ｃを宣言する  Dim b, c As Integer  '長整数型変数ｆと、単精度浮動小数点型変数ｈを宣言する  Dim f As Long, h As Single  '値の代入  a = "test"  b = 365  c = 400  f = 1.5555  h = 2.22  '文字列型変数ｇの宣言と値の代入を同時に行う  Dim g As String = “test2”  End Sub |

次に、htmlコードビュー及びデザインビューを表示して、デザインビューの“左列”の直下に、テーブル（８行、２列）を挿入します。

（テーブルの挿入の手順はこの解説書の「付録」で解説してあります）

Htmlコードビューの<h2>左列</h2>のコードの下側に“kk”という文字を手入力（キーボードから）して下さい。下図

|  |
| --- |
|  |

次に、コードビューとデザインビューの間に、下図のように黄色の「デザインビューとソースビューと同期していません“という表示がでます。

|  |
| --- |
|  |

上図の黄色の部分をクリックして、同期して下さい。下図のように、追加で入力したコードがデザインビューに反映されます。

|  |
| --- |
|  |

上図の“ｋｋ”という文字の右側にマウスのポインタを置くと、VisualStudioの作業画面の最上部のメニューに「テーブル」というメニュー項目が自動で追加されます。次図

|  |
| --- |
|  |

上図の最上部のメニュー項目の「テーブル」をクリックします。下図のようなサブメニューが表示されます。

|  |
| --- |
|  |

上図の「テーブルの挿入」を選択します。下図のような「テーブルの挿入」ダイアログ（設定画面）が表示されます。

自動で設定されている以外は、次図のように設定します。

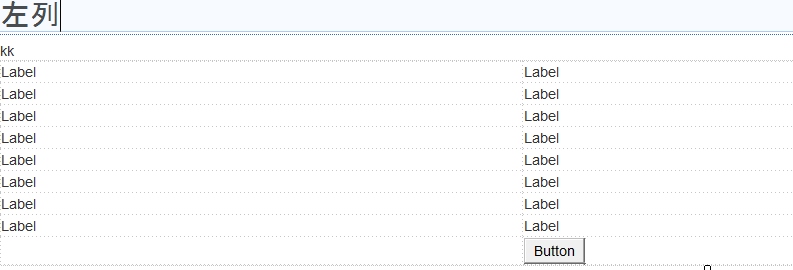
|  |
| --- |
|  |

「OK」ボタンをクリックします。

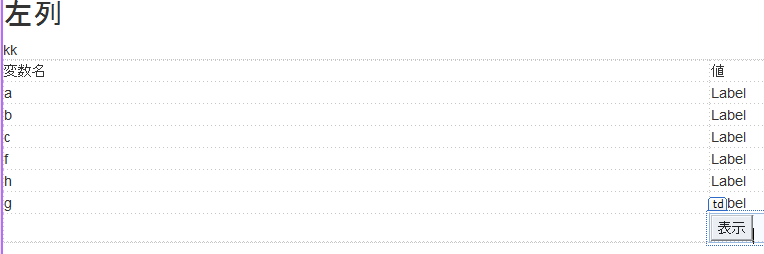
デザインビューは次図のようになります。

|  |
| --- |
|  |

上図のテーブルの各列、各行に下図のようにツールボックスからWebサーバーコントロール（以下コントロールと呼びます）をドラッグアンドドロップして下さい。



次に、テーブル内の各LabelコントロールのTextとButtonコントロールのTextを下図のようにします。



次にテーブルの右列の各Labelコントロールの「IDプロパティ」を下図のようにして下さい。



次に、プログラムコードに移って、下記のようにコードを追加します。網掛け部分を追加します。

|  |
| --- |
| '文字列型変数ｇの宣言と値の代入を同時に行う  Dim g As String = “test2'”  LblA.Text = a  LblB.Text = b.ToString()  LblC.Text = c.ToString()  LblF.Text = f.ToString()  LblH.Text = h.ToString()  LblG.Text = g  End Sub |

ブラウザーで実行して下さい。下図のようになりましたか？

|  |
| --- |
|  |

前図の左列の直下の“ｋｋ”は、テーブルを挿入する時に使用した目印です。

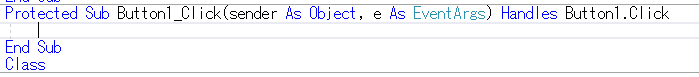
前図の、結果表示をプログラムしているのは、Loadイベントハンドラーの下記の部分です。

|  |
| --- |
| LblA.Text = a  LblB.Text = b.ToString()  LblC.Text = c.ToString()  LblF.Text = f.ToString()  LblH.Text = h.ToString()  LblG.Text = g |

次に、TestHensu\_Loadイベントハンドラー内の上記の網掛け部分を削除します。

### 変数のスコープ

次に、デザインビューの“表示”ボタンをダブルクリックして、Button1のClickイベントを作成して下さい。下図



このボタンコントロールには、特別にID名を設定していなかったから、自動で割り振られたID名をそのままに、Clickイベントが作成されました、状況に応じて、他のコントロールについても、特別にID名を付ける必要がないときは、自動で作成されるID名を使用して下さい。

その中に“LblA.Text = a”と入力して下さい。

|  |
| --- |
|  |

”a“の下に赤の波型のアンダーラインが表示されて、マウスを近づけると、”aは宣言されていません……”という表示が現れます、この意味は、“a”という変数をこのClickイベント内では、知らない状態なのです。“a”という変数は、Loadイベントハンドラー内で宣言されていますので、Loadイベントハンドラー内でだけ有効なのです、そのような規則になっています。したがってボタンのClickイベントは、“a”の変数を知らないのです。この現象を、見えない状態として、“スコープ外”と表現します。

“スコープ”の範囲は、色々あります。参考に大きく分けて次のようなスコープ範囲があります。

1. 変数のページ内のスコープ
2. 変数のサイト内のスコープ
3. 変数のクラス内のスコープ

等です。

### 変数のページ内のスコープの実験

プログラムコードのLoadイベントハンドラー内の下記の網掛け部分のコードを選択して、

|  |
| --- |
|  |
| '文字列型変数ｇの宣言と値の代入を同時に行う  Dim g As String = “test2” |

右クリックメニューから、“切り取り“を選択して切り取り、下記の部分に貼り付けて下さい。

|  |
| --- |
|  |

すると、Button1\_Clickイベント内の“a”の下の赤い波線が消えました。という事は“a”という変数がButton1\_Clickイベントから、見えたという事です。（かしこいですね～）

このことは、変数を“ページクラス内で宣言”したから、という事です。

上記の一行目の“Partial Class TestHensu”というコードで、なんとなく解るように、この

”TestHensu.aspx“というページは、“Class TestHensu”というクラスによって作成（インスタンス化）されていると、認識して下さい。（難しいことは、後回しにして….今はコードを実行して、コードの命令通りにコンピューターが動く事で満足しましょう）

それでは、Button1\_Clickイベントに、下記のコードを追加しましょう。赤のコード部分コピペ可能

|  |
| --- |
| Protected Sub Button1\_Click(sender As Object, e As EventArgs) Handles Button1.Click  LblA.Text = a 'aの型はString型の文字列だから、ToString()関数は使用しない  LblB.Text = b.ToString()　　'bはInteger（整数）型だからToString()関数で文字として表示する  LblC.Text = c.ToString()　　'同上  LblF.Text = f.ToString()　　'同上  LblH.Text = h.ToString()　　'同上  LblG.Text = g　　　　　　　'gもaと同じ  End Sub |

コードビュー及びデザインビューを表示して、ブラウザーで表示します。「表示」ボタンをクリックします。次図のように表示されます。

|  |
| --- |
|  |

これで、変数がページ内で有効であることを確認しました。

実行画面を閉じます。

ここまでの実験では、ページのロード時に、各、変数に値を代入していましたが、Laodイベントを空にして、Button1\_Clickイベント内に、各、変数に値を代入するコードを追加して、右列のLabelコントロールに表示する、プログラムをして見て下さい。

更に読者様の理解が深まります。

先程のButton1\_Clickイベント内のコードで、“ToString()”というコードがありました。

### ToString（）とは

ToString関数と呼びます、

キーワード「“Object.ToString メソッド ()”」です、検索して確認して下さい。

（メソッドという、意味は“値を引き出す式（コード）”と理解して下さい）

VSのテクニカルヘルプで検索した結果、下記のような情報を確認しました。

Object.ToString メソッド ()

Returns a string that represents the current object.

Public Overridable Function ToString As String

2行目を直訳しますと、「現在のオブジェクトの値を“文字（型）”で表示（出力）する。

3行目の意味を、簡単に訳しますと、パブリックなオーバーライド可能な、ToStringという名前の“Functionプロシージャ”であり、String型のデータを返す。

### 変数の使用法について

例えば、変数aの値がString型で、値が“こんにちは”である場合、このような場合は、ToString()関数を使用して、String型に変換する必要はありません。aの値を表示する、LabelXコントロールに代入する場合のコードは、“LabelX.Text = a”となります。

変数fの値が、単精度浮動小数点数型（Single）で“3.14”という値であるときは、ｆの値の型がString型ではないので、ToString（）関数を使用して、LabelXのTextプロパティには、ToString関数を使用して、Single型の値‘3.14’ をString型に変換して、代入する必要があります。その場合の表示（出力）された型は、文字の”3.14“です。

また、変数ｄというDate型の変数があって、日付の値が格納されている場合は、ToString関数によって、“2025/04/15”のような日付を表す文字列に変換して表示します。

これが、ToString関数の役割です。

## 配列：複数の値を表す変数

変数とは、コンピュータープログラムにおいて、データを格納するのに、文字とか数字等のデータをそれぞれのデータの型に応じて、格納する領域を名前を付けて宣言した物（オブジェクト）の事である事を今まで解説し、実践してきました。

プログラマーとしてもう一つレベルアップするために、“変数の配列”という知識を習得する必要があります。

前章の「Dim TodoType As String() = {"仕事", "個人", "行事", "その他"} '大かっこ”{ }”で囲まれた中の文字列をString型の配列に初期化している」というこの部分です。

配列と呼ばれる別の種類の変数は、同じ型の複数の値を格納するのに便利です。

前章の付録の中で詳しく説明してありますので、重複する説明、解説は控えます、実践を通して、さらに理解を深めて、読者様のレベルを上げて下さい。

### TestHairetsu.aspxページを作成

ソリューションエクスプローラのサイト名「TimeTable」を右クリックして、下図のように「追加」メニューにマウスを載せます。

|  |  |  |
| --- | --- | --- |
|  | 表示された左側のサブメニューの「Webフォーム（マスターあり）」を選択します。下図が表示されます。   |  | | --- | |  |   上図のように、項目名の入力欄に“TestHairetsu”と入力します。  次に、「OK」ボタンをクリックします。 |

次に、下図の「マスターページの選択」画面が表示されます。

|  |  |
| --- | --- |
|  | “Site.master”を選択して、「OK」ボタンをクリックします。 |

次に、表示されている、“TextHairetsu.aspx”のコードビューに下記のコードを追加します。赤のコード部分コピペ可能

|  |
| --- |
| <asp:Content ID="Content1" ContentPlaceHolderID="MainContent" Runat="Server">  　　　　<div class=”row”>  <div class=”col-md-12>  　　　　　　　　<h2>一列</h2>  </div>  </div>  </asp:Content> |

次に、プログラムコードファイルに移って。

TestHairetsu ページのLoadイベントハンドラーを作成して下さい。

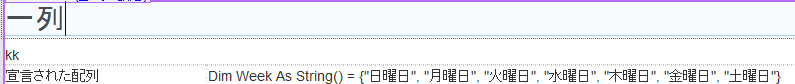
TestHairetsu.aspxが最初にLoadされた時に実行される、配列の初期化を行うコードを“TestHairetsu”のLoadイベントハンドラーに追加します。下記コード、赤のコード部分コピペ可能

|  |
| --- |
| If Not IsPostBack Then　'このページが最初にLoadされた時に実行する  Dim Week As String() = {"日曜日", "月曜日", "火曜日", "水曜日", "木曜日", "金曜日", "土曜日"}  '上記は、配列名”Week”を宣言して、日曜日から土曜日までの、曜日名を配列のデータとして代入しています、  '宣言と同時にデータを代入する事を,"初期化"といいます。  End If |

次に、

TestHairetsu.aspxのコードビュー及びデザインビューを表示して、”一列”の文字の直下に“ｋｋ”という文字を入力して目印にします。その“ｋｋ”の文字の右側にマウスのカーソルをおいて、テーブル（2行、2列）を挿入して下さ。1行目の左列にLabelコントロールを配置して、Textプロパティを“宣言された配列”にします。次に、右列にLabelコントロールを配置して、Textプロパティに

「'Dim Week As String() = {"日曜日", "月曜日", "火曜日", "水曜日", "木曜日", "金曜日", "土曜日"}'」の赤い部分をコピペして、貼り付けます。下図のようになります。

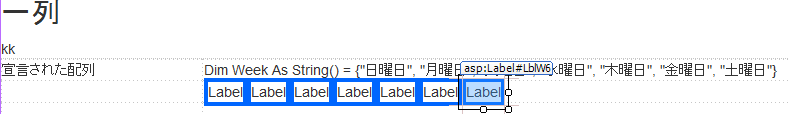


次に、テーブルの2行目の右列にLabelコントロールを連続して、7個配置して、各IDを左から“LblW0”,”LblW1”,“LblW2”,”LblW3”,“LblW4”,”LblW5”、“LblW6”にします。

そして、その7個のLabelコントロールのプロパティを下図のように設定します。

|  |
| --- |
| 青色 |

上図のBackColorは「#6666FF」“青色”（赤色部分コピペ可能）です。BorderStyleは“枠線”の事です。下図のようになります。



次に、htmlコードビューのの２行目の２列目のLabelコントロールが配置された、htmlコード（下記の部分）を表示して、手入力で<br />タグを挿入します。この場合<br　とタイプして、>をタイプすれば自動で“/”の記号は自動で挿入されます。

|  |
| --- |
|  |

デザイン画面は、下図のようにLabelコントロールが縦に並びます。

|  |
| --- |
|  |

次に、プログラムコードファイルに移ります。

TextHairetsu.aspxのLoadイベントに下記のコード、赤のコード部分を追加します。コピペ可能

|  |
| --- |
| Private Sub TestHairetsu\_Load(sender As Object, e As EventArgs) Handles Me.Load  Dim Week As String() = {"日曜日", "月曜日", "火曜日", "水曜日", "木曜日", "金曜日", "土曜日"}  '上記は、配列名”Week”を宣言して、日曜日から土曜日までの、曜日名を配列のデータとして代入しています、  '宣言と同時にデータを代入する事を,"初期化"といいます。  Dim len As Integer = Week.Length - 1 '配列の要素数から１を引く  Dim i As Integer  For i = 0 To len  If i = 0 Then  Me.LblW0.Text = "Week(" & i.ToString() & ")" & " =" & Week(i) 'iの値と日曜日を表示　Week(0)の値です  ElseIf i = 1 Then  Me.LblW1.Text = "Week(" & i.ToString() & ")" & " =" & Week(i) 'iの値と月曜日を表示　Week(0)の値です  ElseIf i = 2 Then  Me.LblW2.Text = "Week(" & i.ToString() & ")" & " =" & Week(i) 'iの値と火曜日を表示　Week(0)の値です  ElseIf i = 3 Then  Me.LblW3.Text = "Week(" & i.ToString() & ")" & " =" & Week(i) 'iの値と水曜日を表示　Week(0)の値です  ElseIf i = 4 Then  Me.LblW4.Text = "Week(" & i.ToString() & ")" & " =" & Week(i) 'iの値と木曜日を表示　Week(0)の値です  ElseIf i = 5 Then  Me.LblW5.Text = "Week(" & i.ToString() & ")" & " =" & Week(i) 'iの値と金曜日を表示　Week(0)の値です  ElseIf i = 6 Then  Me.LblW6.Text = "Week(" & i.ToString() & ")" & " =" & Week(i) 'iの値と土曜日を表示　Week(0)の値です  End If  Next  End Sub |

要素数（データの数）は、日曜日から土曜日まで数えれば、データの数は、７個です、この場合コンピューターに配列として保存されている場所は、Weekという配列名の“Week(0～６)”の場所にあります。コンピューターの数字は、0から始まりますので、Weekの1番目の要素（データ）の場所はWeek(0)の場所（位置）です。したがって、Week配列の値を取得するには、日曜日の場合、Week(0)の値を参照（取得）すればいいのです。

上記のプログラムコードの変数“i”の値（数字）を０から６まで（7回）に変化させてWeek内の要素（データ）をとり出している事を是非理解して下さい。（理解できるまでこのコードを眺めるのも一つの方法です）

コードビュー及びデザインビューを表示して、ブラウザーで表示、実行して下さい。

次図が実行結果です

|  |
| --- |
|  |

目からウロコでしたか？

実行画面を閉じます。

### 実行画面での、文字列の表示方法

先程の実行結果を表示するのに、全てのコンピュータプログラム言語に共通する、文字列表示のコードを説明します。

先程のプログラムコードの例

Me.LblW0.Text = "Week(" & i.ToString() & ")" & " =" & Week(i) を解説します。

Me.LblW0.Text　これはLabelコントロールのID、「LblW0」のTextプロパティ

“Week(“ このダブルクオーテーション「“」マークで囲んだ中の部分をそのままの文字で表示すします。ダブルクオーテーション「“」マークで囲むと、囲まれたデータはString（文字）ですよ、と宣言したことと同等です。

「＆」記号は　文字列と文字列をつなぐという役割です、「＆」記号の代わりに「＋」記号を使用しても、同じです。

i.ToString() は、変数iの値をString型に変換して、表示しています。このようなi等のデータの値を表示する時は、「“」で囲みません。「”」で囲みますと、iをデータとして認識しないで、文字として表示してしまいます。

このような、記法をマスターするのに、筆者の場合、「LabelX.Text = “ｘｘ” + “)” & i.ToString()」等というように、何度もキーボードから手入力して、身に着けたことを思い出します。

プログラマー必須の定型文

## プロシージャ（関数）とは？

コンピュータ―プログラムの世界では、“メソッド”、“プロシージャ”または“関数”という言葉が頻繁に出てきます。“プロシージャ”、“関数”双方とも“メソッド”と呼びます、今までに出てきた“Button1\_Click”や“PageLoadイベントハンドラー”もメソッドです。

“メソッド”を直訳しますと、“順序、筋道、秩序”です。

“プロシージャ（procedure）”を直訳しますと、“（進行上の）手続き、手順、順序”等の意味があります。

技術情報：[基本的なメソッド定義と呼び出し](https://msdn.microsoft.com/ja-jp/library/dd314348.aspx)　のリンクを参照して詳細を読んで理解を深めて下さい。

日本語で“関数”と呼ぶのは、数学の関数から来ているように思われますが。これから解説する、“ファンクションプロシージャ”を理解すれば、“関数”の語源が理解できるでしょう。

ここから以後“プロシージャ”の事を、日本語の“関数”または、“メソッド”、“サブ関数”、“ファンクション関数”等と呼んで、解説を進めます。

関数、メソッドには大きく分けて、次の2つに分かれます。

1. 値を返さない“サブ（Sub） 関数”
2. 値を返す“ファンクション（Function）関数”
3. のサブ関数は、イベントハンドラーに象徴されるような、関数名の前に“Sub”という語をつけて、”値を返さないメソッド“であると宣言します。
4. のファンクション関数は、関数名の前に“Function”という語をつけて、”値を返すメソッド“であると宣言します。読者様にとってはまだ見たこともない関数ですが、これから実践する“TestProcedure.aspx”のページで体験していただきます。

それでは、この章の、「サイト内に新規にWebページを作成する」項を参考にして、「Webフォームマスターあり」

項目名”の入力欄に“TestProcedure”と入力して、“OK”ボタンをクリックします。マスターページは「Site.master」を選択します。

TestProcedure.aspxページのコードビューが表示されましたら、「並べて表示」をクリックして、コードビュー及びデザインビューを表示して下さい。

コードビューのhtmlコードでページを3列に分割して表示するようにします。赤のコード部分コピペ可能

|  |
| --- |
| <asp:Content ID="Content1" ContentPlaceHolderID="MainContent" Runat="Server">  　　　　<div class=”row”>  <div class=”col-md-4”>  <h4>左列</h4>  </div>  <div class=”col-md-4”>  <h4>中央列</h4>  </div>  <div class=”col-md-4”>  <h4>右列</h4>  </div>  </div>  </asp:Content> |

コードビュー及びデザインビューを表示した状態で、ブラウザーで実行します。下図



## htmlコードとプログラム言語の関係

技術情報：[ASP.NET Webページの概要](https://msdn.microsoft.com/ja-jp/library/428509ah(v=vs.100).aspx)　（日本語に翻訳するボタンが出現します）

特に、上記のヘルプドキュメントの“ASP.NET　Webページを使用して出来ること”の項に目を通しておいてください。

HTML（ハイバー.テキスト..マークアップ.ランゲージ）の知識を得るにはインターネットに溢れるほどの情報がありますので、是非検索して知識を習得して下さい。

前章でも、解説済ですが、Webページそのものは、htmlコードによって表示されています、解り易く言えば、VisualStudioという開発ツール（多言語の）を使用して、プログラムされたWebアプリは最終的には、HTMLコードにコンパイルされて、Webブラウザーによって表示されます。そのプログラムの過程においてプログラマーの困難な部分、例えば入力用のWebサーバーコントロール等、マイクロソフトのASP.NETフレームワークが、プログラマーが膨大な労力を要するWeb用の資産（VSのツールボックス内のツール）を既に作成してくれているのです、そのことは、VSのツールボックスから、コードビュー及びデザインビューにWebサーバーコントロールをドラッグアンドドロップした後の、コードビューを確認すると、<asp:Label ID="Label1" runat="server" Font-Bold="True" Font-Size="1.2em" Text="タイトル"></asp:Label> のようにhtmlコードでマークアップされます。

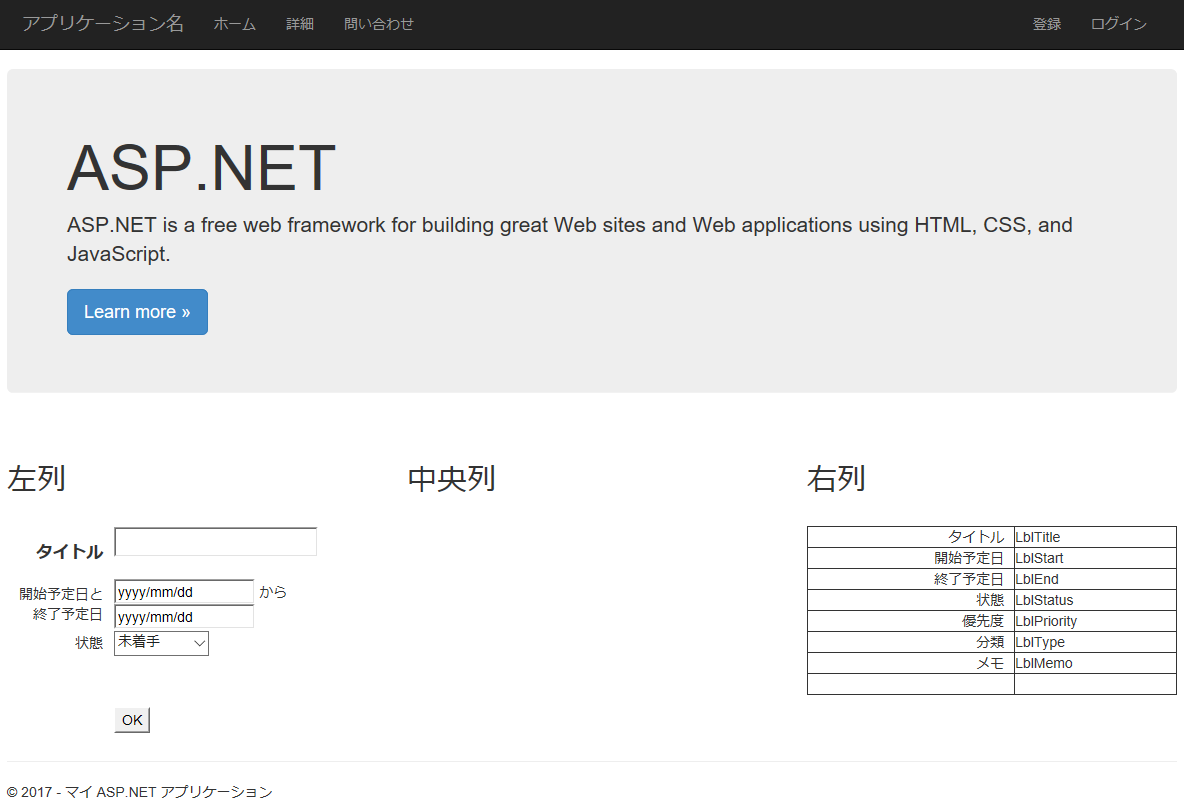
また、使用するプログラム言語とプログラムコードを参照するために、VSの各ページの1行目に下記のような記述があります。赤色の部分

<%@ Page Title="Home Page" Language="VB" MasterPageFile="~/Site.Master" AutoEventWireup="true" CodeFile="Default.aspx.vb" Inherits="\_Default" %>

使用するプログラム言語が“C#”の場合は、”Languege=”C#”“になります。

## プログラマーなら“CSS”を理解しよう

VisualStudioでWebサイトを作成して、マスターページを親とする子のページを表示した場合の画面をざっくりとプログラマー目線で見ると、下図のように解剖できます。



コンテンツの中の <div class=”row”>内

コンテンツの中の　ヘッダー部

コンテンツ部（各ページ）

フッター部

メニューバー

前図の「メニューバー」と「フッター部」はマスターページ内で構成されています。

コンテンツ部は、各ページでプログラムしたものが表示されています。

次に、マスターページと各ページの関係を図にしました。下図

マスターページ

メニューバー

各ページのコンテンツ

ID=“MainContent”

フッター

前図のように、「メニューバー」、「各ページのコンテンツ」、「フッター」は、マスターページ内に、あります。

サイト内に“新規にWebページを作成”する時に、“Webフォーム（マスターあり）”のメニューを選択した場合に作成したページは、前図の「各ページのコンテンツ」部に表示されます。

## マスターページの構造

マスターページも他のWebページと同様、htmlコードで作成されて表示されます。

ソリューションエクスプローラウインドウから、“Site.master”をクリックして、Site.masterのコードビューを開いてください。

メニュー表示部分が下記のコードで表示されています。

### メニュー表示部分

|  |
| --- |
|  |

43行目の<div…はメニューを表示す範囲の背景色、等を定義しています。

45行目から50行目までは、モバイル用にメニューを表示するための各種の設定をしています。５４行目から、58行目までがパソコン、タブレット用のメニュー表示のコードです。

### ページの表示部分とフッター部分

「ページの表示部分とフッター部分」を担っているのが、下記の部分です。

|  |
| --- |
|  |

「<asp:ContentPlaceHolder>～</asp:ContentPlaceHolder>の中に、各ページが表示されます。<footer></footer>も後で、変更して見ましょう。

その他の、htmlコードについては、わが社の実践解説書の高度化に合わせて、解説しています。

ざっくり（大雑把）に理解して下さい。コンピュータ―プログラムの世界は膨大な量の知識が必要です、優秀なプログラマーは、どの情報がどの辺にあったかを知っている（経験している）ことなんです。

## HTMLとCSSの関係

Webページの本体がHTML（ハイパーターミナルマークアップランゲージ）です。以前はこれだけで作るのが主流でしたが、現在は、CSS（カスケードスタイルシート）を絡めて作るのが多くなってきています。下欄はヘルプの記事からの引用です。

|  |
| --- |
| カスケード スタイル シート (CSS: Cascading Style Sheet) には、Web ページの要素に適用されるスタイル ルールが格納されます。CSS スタイルは、ページでの要素の表示方法と配置を定義します。ページの各要素に属性を個別に割り当てる代わりに、要素のインスタンスや、特定のスタイル クラスに割り当てられた要素が Web ブラウザで見つかるたびに属性を適用する一般ルールを作成します。 |

１、CSSは文章に直接styleで書き込む事もできる。

２、HTMLの上部（head部）に指定する事もできる。

３、外部にCSSファイルを置きlinkする事もできる。

１，２、３のどれかの方法です（混合も可能）

多くのＨＰを作る場合、「３」の一つの外部CSSにlinkさせると一つのCSSだけで全部のＨＰのHTMLを同様に指定できます。

例えば　「赤い大きい文字」が１０箇所あってそこの部分だけを「黄色の小さい文字」に変えたい時HTMLだけで作ってある場合は、その１０箇所を一箇所づつ変えなければなりませんが、「２」「３」の様にCSSで指定していた場合は

CSSの一箇所だけを変えれば１０箇所全部が変更可能になるのです。

また、HTMLだけでもＨＰは作れます。CSSを使うことでプログラム効率も上がると言う事です。

当然必要最低限だけCSSを使う事もできます。

（これらの情報は、インターネットで検索すれば詳しく解説してあります。）

CSS,スタイルシートの詳細については市販の解説書をご用意して下さい。本解説書はプログラミングについての解説ですので、CSS,スタイルシートについては、必要最小限にとどめています。

### スタイルの定義位置の検索の仕方

まずメニュー表示部分のコードを見てみます。下記

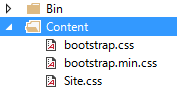
|  |
| --- |
| <ul class="nav navbar-nav">  <li><a runat="server" href="~/">ホーム</a></li>  <li><a runat="server" href="~/TestHairetsu.aspx">配列テスト</a></li>  <li><a runat="server" href="~/About">詳細</a></li>  <li><a runat="server" href="~/Contact">問い合わせ</a></li>  </ul> |

<ul> <li> </ul>によって、箇条書きを指定しています、<ul class=”……..”>によって箇条書きのスタイルを指定しています。この場合<li></li>の間の項目を横並びに指定しています。

そのスタイルとは、前項のマスターページでのCSSファイルのリンクにより“bootstrap.css”ファイル内に定義されています。その定義されている場所の見つけ方を説明します。

まず、<ul class=”nav navbar-nav”>の“navbar-nav”という文字だけをコピー状態にします、

次に、ソリューションエクスプローラのツリー構造の中の、Site.masterでリンク先をしていた、“~/Content/css”を探します。下図の位置にありました。



上図の“bootstrap.css”を選択して、CSSファイルの中身を表示します。

次に、VSのメニューバーの「編集」メニューをクリックして、表示されたドロップダウンメニューの上から二番目の“検索と置換”にマウスを乗せて一番上の“クイック検索”をクリックします、下図のように“navbar-nav”と貼り付けか、または入力して、右矢印をクリックして、検索を開始します、下図



すると次図のように、その文字が使用されているのコードの位置にジャンプします。

|  |
| --- |
|  |

上記のオレンジ色に選択されたブロック（｛～｝の間）の内容が、下記の、マスターページ内のhtmコード　<ul　のスタイルが定義されている位置になります。

|  |
| --- |
| <div class="navbar-collapse collapse">  <ul class="nav navbar-nav">  <li><a runat="server" href="~/">ホーム</a></li>  <li><a runat="server" href="~/About">詳細</a></li>  <li><a runat="server" href="~/Contact">問い合わせ</a></li>  </ul> |

ここまでの説明で、

1. 外部CSSのリンクの記述位置
2. 外部CSSのソリューションエクスプローラ内のフォルダの位置とファイルの位置
3. CSSファイルの開き方
4. CSSファイル内のスタイルの定義位置の検索の仕方

等を実践しました。尚CSSの詳細（スタイルの定義）等は、インターネットの情報などで、知識を習得して下さい。図書も本屋さんで、数多く出版されています。

### HTMLの<a>~</a>要素を理解する

Webページのhtmlコードを見ると、コードビューの１行目から<>で囲まれた文の連続で構成されています。例えば<head runat=”server”></head>,<body></body>等という記法で記述されています。<body>~</body>の内にも複数の<>~</>の記述があります、この<>~</>の事を要素と呼びます。<body>~</body>の呼び名を“<body>要素”または”<body>~</body>”要素という呼び方をします。これ以後チュートリアルを進めるのに”<a>要素“だけはここで理解しておきましょう。Site.masterのメニュー表示のマークアップの中で次のような行があります。

<li><a runat="server" href="~/About">詳細</a></li>　を直訳しますと、

「箇条書きとして表示される“詳細”という文字は主ルートの“About.aspx”のページへ遷移する」という意味になります。

[runat=”server”]　はサーバー側で実行、「href」は遷移先、「About」はaspxという拡張子を省略したWebページ名です。この<a>要素は頻繁にでてきますが、記法を暗記する必要はありません、様々なシーンで使い分ける必要があります。要は「<a …..href=”…”>」の形によって、ページ遷移を実行できる（ハイバーリンク）だと記憶しておいて下さい。

## データベースの重要性

一昔前は、データベースの構築といえば、膨大な知識と時間をかけなければできないことでした、がVisualStudioの機能として簡単に世界一優秀な“MicrosoftのSQLサーバー”によるデータベースを作成することができます。このデータベースの仕組みを理解して、使いこなすことにより、読者の皆様はコンピュータープログラマーとして一層の自信になるでしょう。最近は、コンピューターアプリケーション（Web、ウインドウズ）全体のアプリ開発において、データベースの活用は欠かせないどころか、Webアプリにおいてはデータベースの活用シーンは、全ページに関係する程です。

まず本チュートリアルの実践を通してデータベースの基礎を身につけましょう。

## データベースを理解する

データべースは身近に存在しています。例えば携帯電話、スマホの中に電話番号と名前の関係を結びつけてあるデータが、そもそもデータベースという概念です。

### 表

データベースの中には、いくつもの表が格納されています。表は列と行から構成されています。下図を見て下さい

列名

|  |  |  |
| --- | --- | --- |
| 商品コード | 商品名 | 単価 |
| 1 | アジ | 200  行 |
| 2 | サバ | 200 |
| 3 | マグロ | 500 |
| 4 | タイ | 800 |

値

列

上図は商品の価格表を示しています。この表は、「商品コード」、「商品名」、「単価」という３つの列と４つの行から構成されています。各行の各列には、値が格納されています。尚、列の名前の事を“列名”と呼びます。表の事を“テーブル”、行の事を“ロウ”、列の事を“カラム”と呼ぶ場合もあります。

### 主キー

表の中の１つの行を特定するために使う列の事を“主キー”と呼びます、下図を見て下さい。

単価「200」では行を特定することが出来ない

|  |  |  |
| --- | --- | --- |
| 商品コード | 商品名 | 単価 |
| 1 | アジ | 200 |
| 2 | サバ | 200  主キー：商品コード「２」で行を特定することが出来る。 |
| 3 | マグロ | 500 |
| 4 | タイ | 800 |

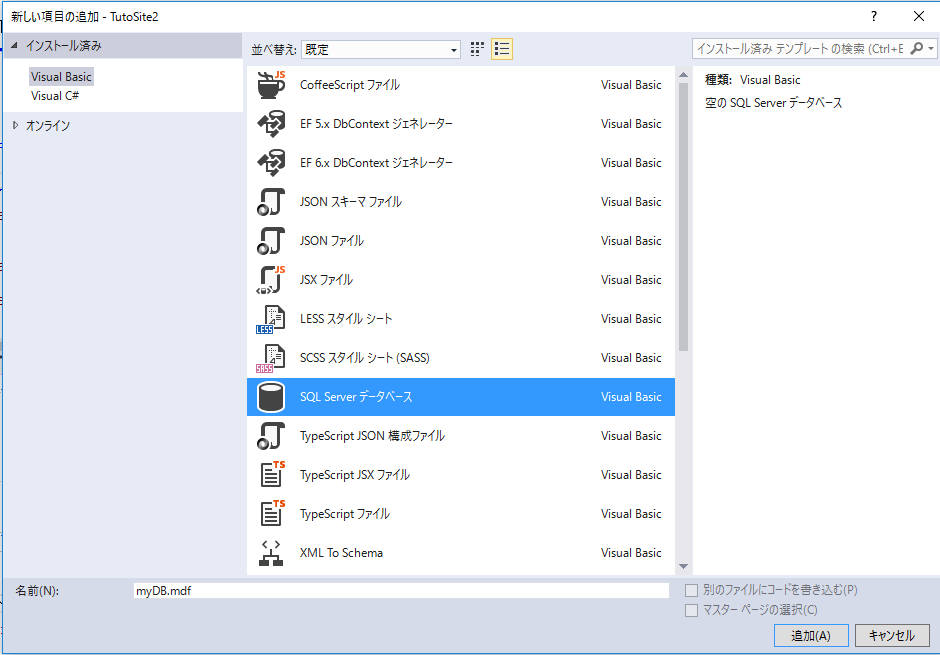
ここでは、「商品コード」列の値によってそれぞれの商品、つまり表の中の１つの行を特定する事が出来ます。したがって「商品コード」列がこの表（テーブル）の“主キー”になります。主キーには表の中で行を特定することが出来る値を設定しなければなりません。例えばこの表(テーブル)には「単価」列に“200”という値を持つ行が２つあります、そのため「単価」の列の値によって行を特定することが出来ません。

この後のチュートリアルでデータベースシステムを実践していきます。“行と列”の構成をしっかり理解しておいてください。

## サイトにデータベースを構築する

1. ソリューションエクスプローラの「App\_Data」フォルダを右クリック。
2. 表示されたドロップダウンメニューの「追加」の上にマウスを乗せる
3. 続いて表示されるメニューの一番上の「新しい項目の追加」をクリックします。
4. 表示された「新しい項目の追加」一覧表の上から９行目の「SQLサーバーデータベース」を選択します。

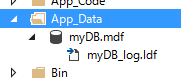
下図のように、名前の入力欄に“ｍｙDB.mdf”（任意の名前）と入力します。



「追加」ボタンをクリックします。

しばらく時間がかかります。（コンピュータが「myDB.mdf」を作成しています。）

ソリューションエクスプローラの「App\_Date」フォルダの配下に下図のような「myDB.mdf」というデータベースが作成されました。



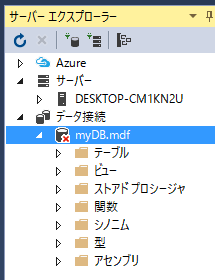
同時にサーバーエクスプローラ（表示していない場合は、ドキュメントウインドウのメニュー「表示」を選択します。

|  |
| --- |
|  |

下図のように、サブメニューが表示されます。

|  |
| --- |
|  |

上から３行目に「サーバーエクスプローラ」をクリックして表示して下さい。）に次図のように自動で、「myDB.mdf」が作成されています。



上図のように、「myDB.mdf」データベース内には「テーブルフォルダ」～「アセンブルフォルダ」まで、７つのフォルダがあります、それぞれのフォルダが「myDB.mdf」の配下にあると認識しておいてください。

次に、下図の部分の“ドロップダウンリスト”コントロールに表示する“Item”のデータをデータベース（テーブル）から取得するようにプログラムします。



上図のドロップダウンリストに表示する“Item）（項目）”はプロパティウインドウから設定することも出来ますが、今回はその”Item”をデータベース（テーブル）に登録しておいて、そこからドロップダウンにデータ（“Item”）を表示するようにします。そのための”テーブル“を作成します。なぜこのような手法を使用するのか、これ以後の実践を通して理解して下さい。高レベルなコンピュータープログラムの入り口です。

### データベースのデータ型

ここまで、データベース“myDB.mdf”を作成しました。このことを別の言い方をしますと、データベース“myDB.mdf」”ファイルを作成（定義）しました、とも言います。

そのデータベースのファイルの中（配下）にテーブル（表）が含まれることも説明しました。ここではその準備段階としてテーブルに格納するデータの種類について触れておきます。

表のそれぞれの列には、同じ種類のデータを格納します。ある列には“数値”を、ある列には“文字”を格納します、同じ列の行ごとに異なる種類のデータを格納することはできません。例えば、行によって、“数値”を格納したり、“文字”を格納したりすることは出来ません。このように列に格納するデータの種類の事を“データ型“と呼びます。

テーブルを作成する場合は、列毎にデータ型を指定し、それぞれの列にどのような種類のデータを格納するかを定義しておきます。Microsoft　SQLの規格では、文字列型や数値型など、いくつかのデータ型が定義されています。一般に文字列型には、英数字（「A」「8」）、半角カタカナ（「ｱ」）、漢字（「松」）、ひらがな（「あ」）、その他の全角文字（「A」「8」「ア」）などを格納することが出来ます。また数値型の列には「123」「1.25」等を格納することが出来ます。

下表は、使用頻度の高いデータ型の説明です（初心者用）

|  |  |  |  |
| --- | --- | --- | --- |
|  | 型名 | 説明 | 使用例 |
| 数値型 | int | 整数（１～200000位まで） |  |
| float | 浮動小数点数 | 小数点の数字を格納 |
| 文字列型 | nchar() | 固定長文字列 | nchar(10) 2バイト文字を10文字以内で格納 |
| nvarchar() | 可変長文字列 | nvarchar(50) 2バイト文字を50文字以内で格納 |
| ビット列型 | bit | ON/OFF | ０（False）または　１（True）を格納 |
| 日付型 | date | 日付と時刻 | 2018/12/30 00:00:00 という形式で格納 |
|  |  |  |  |
| バイナリ型 | image | イメージデータをバイナリで格納 | 写真などのイメージデータをバイナリデータで格納 |

前項の「データベースを理解する」で説明しましたテーブル（表）を作成します。

### テーブルの作成方法

作成するテーブルは下図のようなテーブル（データ登録後）を作成します。

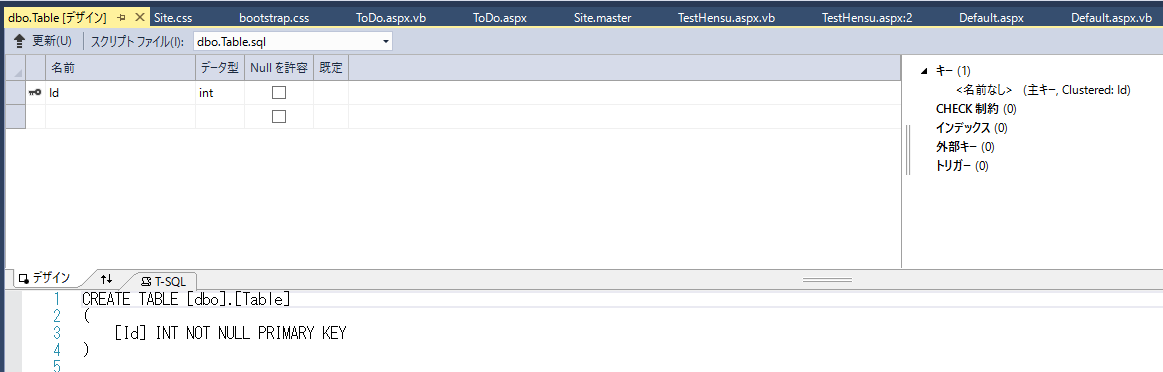
テーブル名：“Status”

|  |  |
| --- | --- |
| **ID** | **Item** |
| 1 | 未着手 |
| 2 | 開始 |
| 3 | 継続 |
| 4 | 完了 |
| 5 | 保留 |
| 6 | キャンセル |

1. サーバーエクスプローラで、「myDB.mdf」の配下にある「テーブル」フォルダを右クリックします
2. 表示されたドロップダウンメニューの「新しいテーブルの追加」をクリックします。

下図のようなテーブルのデザイン画面が表示されます。ユーザーが編集し易いように汎用的なひな形が表示されています。

テーブルのデザイン画面



テーブル名をここで指定します

表示されるコードはT-SQL文と言って、テーブルを作成したり、操作するためのSQL言語のコードです。コードは列名追加の欄と連動（リンク）しています。

列名を追加して、データ型を指定します

表示された、デザイン欄（上の欄）に下図のように変更、入力します



すると、下のT-SQL欄は、自動でCREATE文が作成されます、次図

|  |
| --- |
|  |

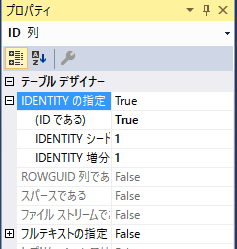
このように、上のデザイン欄に必要な項目を設定すると、下の、T-SQL文は自動で変更されるように両者共に連結されています。逆にT-SQLを変更すると、上のデザイン欄も変更されます。

### テーブルのプロパティの設定

次に、デザイン欄のIDの行の任意の位置をマウスで右クリックします。作業画面の右側のソリューションエクスプローラ内のプロパティウインドウが自動で表示されます。下図

|  |
| --- |
|  |

表示されたプロパティ項目の一番上の“IDENTITYの指定”の＋記号をクリックして、下図のように設定して下さい。



自動で設定される

ここをTrueに設定

上図、IDENTITYの指定の意味は、ID列に対して、データベースシステムが自動で連番を割り振るようにするために“IDENTITYの指定”プロパティを“True”にして、データが追加される毎に、最初のデータのID値は（IDENTITYシード=１プロパティにより）1にして、、2行目以降のID値を前行のIDの値に、（IDENTITY増分= 1　プロパティにより）

”1“を加算します。

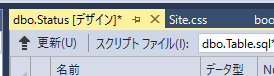
上図、プロパティ設定終了と同時に、下図のようにT-SQL文が自動で変更されています。

|  |
| --- |
|  |

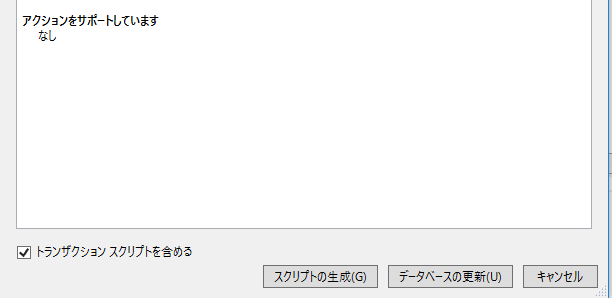
次に、T-SQL欄の、[dbo].[Table]の[Table]を[Status]に変更します。下図

|  |
| --- |
| CREATE TABLE [dbo].[Status]  (  [ID] INT NOT NULL PRIMARY KEY IDENTITY,  　　[Item] NCHAR(10) NULL  ) |

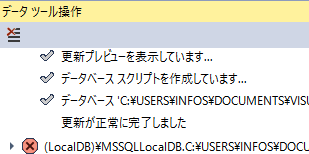
次に、デザイン画面の左上の「更新」ボタンをクリックします。



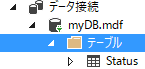
次に、「データベース更新のプレビュー」画面が表示され、少し時間を置いて、下図の「データベースの更新」をクリックします。



すると、ドキュメントウインドウの最下段に「データツール操作」というウインドウが表示されます。下図



上図の“更新が正常に完了したました。”という表示を確認して、ソリューションエクスプローラ内のサーバーエクスプローラの「テーブル」フォルダを右クリックして、表示されたメニューの「最新の情報に更新」をクリックします。そして、テーブルフォルダを開いて、下図のようにテーブルが作成されていることを確認します。



ここまでで、「myDB.mdf」ファイルの中に“Status”テーブルを作成して、次に進む準備が出来ました。

## マスターページの編集

### マスターページのメニューに項目を追加

Site.masterのhtmlコードの、パソコン、タブレット用のメニュー表示、記述コード(マークアップ)部に下記のコード（選択部分）を追加します。既に作成してある“TestHairetsu.aspx”ページをメニュー項目に追加します。

|  |
| --- |
|  |

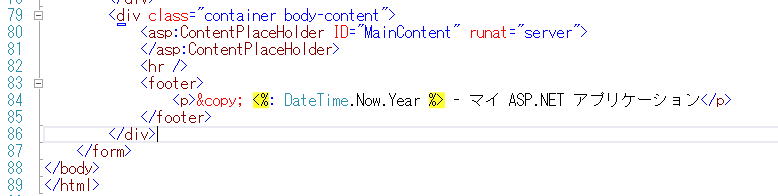
ソリューションエクスプローラのサイト名（TimeTable）の右クリックから、ブラウザーで表示、を実行してメニューに“配列のテスト”が表示されていることを確認して下さい。

|  |
| --- |
|  |

実行画面を閉じます。

### マスターページのフッターを編集

次にSite.masterのhtmlコードの、下記の部分に注目します。



上記の四角で囲んだ部分が、“ContentPlaceHolder”と言って、IDが“MainContent”になっています。このMainContent内に各ページが表示されるようになっています。

また、赤線の楕円形の部分が“フッター”部です、下記のようにフッター部分のコードを変更します。網掛け部分

|  |
| --- |
| <footer>  <p>&copy; <%: DateTime.Now.Year %> - 行動予定 アプリケーション</p>  </footer> |

ソリューションエクスプローラのサイト名の右クリックから、ブラウザーで表示、を実行して下さい。変更部分の確認をして下さい。

実行画面を閉じます。

### アプリケーション名の編集

次に、メニューバーの、アプリケーション名のところの、htmlコードを変更します。行番号は参考にして下さい。

変更前

|  |
| --- |
| または、ApplicationName |

変更後、<a>要素は取り除き<div>要素にします。

|  |
| --- |
|  |

ソリューションエクスプローラのサイト名の右クリックから、ブラウザーで表示、を実行して下さい。変更部分の確認をして下さい。

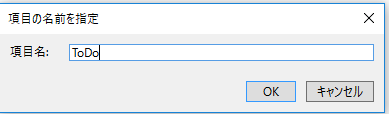
|  |
| --- |
|  |

実行画面を閉じます。

# 本物のWebアプリケーション

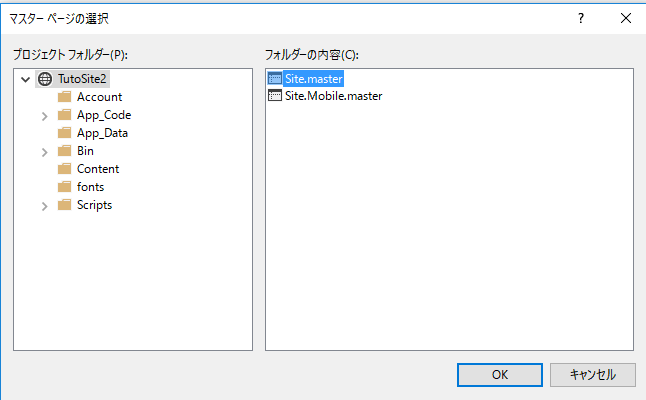
## サイト内に新しいWebページ（フォーム）を作成する

1. ソリューションエクスプローラのサイト名（TimeTable）を右クリック
2. 表示されたドロップダウンメニューから“追加”にマウスを乗せる
3. 表示されたドロップダウンメニューの一番下の“Webフォーム（マスターあり）”をクリック
4. 次に表示された“項目の名前を指定”の項目名の入力欄に“ToDo”と入力する、下図



「OK」ボタンをクリックします。

下図の“マスターページの選択”ダイアログウインドウで“Site.master”を選択して、「OK」ボタンをクリックします。



これでソリューションエクスプローラの主ルートに“ToDo.aspx”というページ(フォーム)が出来ました。確認して下さい。

## マスターページとの親子関係

VSの作業画面は、“ToDo.aspx”のコードビューが表示されていま、下図

|  |
| --- |
|  |

上図の3~5行目<asp:Content ID=”Content1” ContentPlaceHolderID=”MainContent” Runat＝Server>

</asp:Content>

に注目して下さい、マスターページのhtmlコードの80行目の下記のコードのIDと一致しています。行番号は参考です。

|  |
| --- |
|  |

つまり各ページの作成時にマスターページを指定して作成した場合、各ページとマスターページは、親（マスターページ）と子（各ページ）の関係になります。

子のページにおいて、ページの内容を表示するための“Content”この場合ID名は“Content1”です。“ContentPlaceHolderID=MainContent”という記述は、マスターページの”ContentPlaceHolderの ID名“MainContent”を参照するように指定しています。マスターページに”MainContent”というContentPlaceHolderがあれば、親子関係が確立します。

（子のページのLoad時に、マスターページの”ContentPlaceHolder”のIDを参照します、ID名が一致しないとエラーになります）

試しに“ToDo.aspx”のhtmlコードの下記の部分を変更して、ブラウザーで実行して見て下さい。

|  |
| --- |
| <asp:Content ID="Content1" ContentPlaceHolderID="MainContent" Runat="Server">  MainCont  </asp:Content> |

エラーの内容を確認出来れば、このページがマスターページを参照していることがわかるでしょう。

実行画面を閉じて、正しいID名に直してください。

これでマスターページの“ContentPlaceHolder”内に、子となる各ページが抱合されることが実証されました。

## ページつくりの基礎の基礎

Webページ作成においてBootstrapという、CSSのフレームワークの知識は見逃すことは出来ません。

ASP.NETでも、WebFormテンプレートでは、標準で**Bootstrap**を採用しています。現在読者様がVisualStudioを起動してこの実践をしている環境では、特別な準備をしなくても利用できます。

BootstrapのCSSでは

1. <div>~</div>によって、範囲を決める（divには、詳細なスタイルが設定出来ます）
2. <div>~</div>の範囲内に、任意の数のカラム（列）を作成することが出来ます
3. 任意の（カラム）内に、更に任意のカラム（列）も作成出来ます。この状態をネストと呼びます

Bootstrapについては、インターネット上でたくさんの情報があります。

Bootstrap日本語リファレンス　　<http://bootstrap3.cyberlab.info/>

例：col-md-4 等の数値４については、上のリンクのBootstrap日本語リファレンスを参照して理解して下さい。

是非、読者様も上記のアドレスをインターネットブラウザーの“お気に入り”に加えて重宝して下さい。

VisualStudioでは、Bootstrapは既にバンドルされていて、htmlコード内に直接記述できます。

Todo.aspxのコードビューの“Content”内に下記の赤のコードを追加します。赤のコード部分コピペ可能

|  |
| --- |
| <div class="jumbotron">  <h3>マイ スケジュール</h3>  <div class="row">  </div>  </div>  <div class="row"> <%--二つ目の<div class="row"--%>  <div class="col-lg-4 col-md-4 col-sm-4">  </div>  <div class="col-lg-4 col-md-4 col-sm-4">  </div>  <div class="col-lg-4 col-md-4 col-sm-4">  </div> <%--二つ目の<div class="row" の終わり--%>  </div> |

“Jumbotron”はBootstrapのCSSで定義されています。下記が定義部分（前項の“スタイルの定義位置の検索の仕方”で検索して下さい）

.jumbotron {

padding: 30px;

margin-bottom: 30px;

font-size: 21px;

font-weight: 200;

line-height: 2.1428571435;

color: inherit;

background-color: #eeeeee;

}

上記の網掛け部分に注目、背景がライトグレーに設定されています。

そのjumbotronの中を<div class=”row”で class rowの定義を利用します。

デザインビューは下図のようになっています。

|  |
| --- |
|  |

ページのhtmlコードとデザインの対比によって自分の意図する表示に慣れて下さい。

## <div の背景色をVS側で変更する

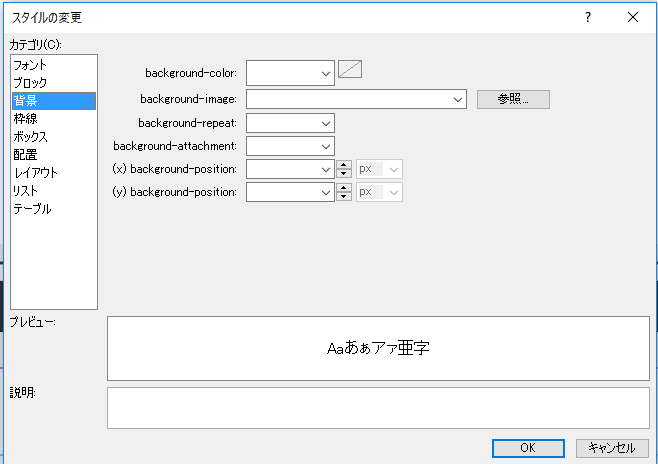
デザインビューで、下図のの列をマウスでクリックします、左上に<divの要素名が表示され、そのタブをクリックして、選択状態にします。すると同時にコードビューに該当する要素も選択されます。下図

|  |
| --- |
|  |

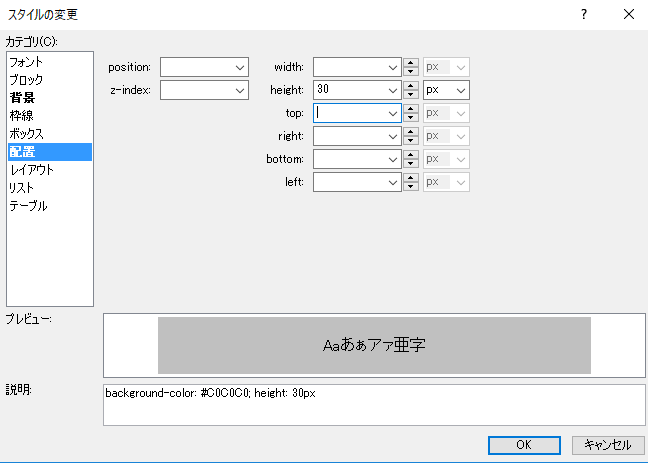
この状態で、ソリューションエクスプローラのプロパティウインドウを表示します、プロパティウインドウには、選択された<div要素のプロパティが表示されています、styleプロパティを展開します。

|  |
| --- |
|  |

下図のように“背景”を選択します。



上図の一番上の“background-color”のドロップダウンを開きます。カラーピッカーが開きますので、下段の薄い灰色を選択します、するとプレビュー欄に選択したカラーと文字が表示されます。次に”配置“という項目をクリックして、下図のように”height”を”30px”にします。



「OK」ボタンをクリックします。

<div要素全体の背景色が薄い灰色になりました。続いて下図のように、一つ列をおいて、3番目の列を選択して、同じ色の背景色にして、“配置”の“height”を”30px”にします。

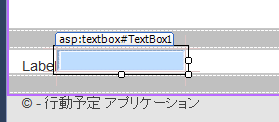
|  |
| --- |
|  |

「OK」ボタンをクリックして、終了します。

このようなCSSの要素をVisualStudio側で変更しても、CSSファイルの当該要素のコードまでを変更することはありません。

## Webサーバーコントロールの配置

続いて、デザインビューで、背景色を設定していない、真ん中の列にツールボックスから、LabelコントロールとTextBoxコントロールを配置します。下図



### コントロール間に隙間を空ける

Labelコントロールの“Textプロパティ”を“タイトル”にして、プロパティウインドウからFontプロパティのドロップダウンを開き“Font-Sizeプロパティをキーボードから”1.2em”にします。デザイン画面ではLabelコントロールが選択状態になっている筈です、この状態でキーボードの右矢印キーを押下しますLabelコントロールの選択状態が解除されてマウスのキャレットがTextBoxとの中間に点滅します、この状態でキーボードのスペースキーを5回押下します（両コントロールの間のスペースを空ける為です）。

コードビューには、下記のように自動でコードが追加されています。赤色の部分

|  |
| --- |
| <div class="col-md-4">  <asp:Label ID="Label1" runat="server" Font-Size="1.2em" Text="タイトル"></asp:Label>  &nbsp;&nbsp;&nbsp;&nbsp;  <asp:TextBox ID="TextBox1" runat="server"></asp:TextBox>  </div>  <div class="col-md-4" style="background-color: #C0C0C0">  </div> |

次にTextBoxコントロールのIDを“TxtTitle”にして下さい。

次に、TxtTitleの右隣に“Buttonコントロールを配置して、IDを”BtnTitle”にして、Textプロパティを“次へ”と入力して下さい。そして、スペースを（半角スペースで）5回分空けます。

実行画面下図



### 同一ページ内で列（カラム）数を変える

次に、新たな領域を確保します、下図の位置に<div>要素を追加します。下記の“Panelの終わり”というコメントの直上のｋｋの位置です。

|  |
| --- |
|  |

<div>のclass＝“row”要素の>< の部分にマウスカーソルを置いて、リターンキーを押下して、下図のようにしてください。

|  |
| --- |
|  |

次図のように<div class=”row” でページ幅を確保してから、新たな列の構成をします。選択部分

本実践解説の場合は次図のようなページのデザインを予定しています。

|  |
| --- |
|  |

そして前図のデザイン構成を図にすると、下図のようになります。

|  |
| --- |
| **A**  **B**ootstrap  **３**  **2**5日  **１**  <div class=”row”を２列に分割  <div class=”row” を３列に分割  <div class=”jumbotron” |

上図の３番目の領域は、AとBに、２列に分割してあります、その理由は、このページが初めて表示された時には、AもBも非表示の状態にして、２のタイトルをユーザーが入力し、２の「次へ」ボタンをクリックしたら、Aを表示し、Aの必須のデータが入力されて、Aの「次へ」ボタンがクリックされたら、Bを表示するというプログラムを予定しているからです。

それでは、早速上記のプログラムをしていきます。

今現在は、３の部分では、下図のように、<div class=”row”> ～　</div>　が出来ています。

|  |
| --- |
|  |

上図の　<div class=”row”> ～　</div>　を２分割するためのhtmlコードを、その中に追加します。

下記のhtmlコードで２分割します。<div class=”row”> ～　</div>　の間に下記の赤色のコードを追加して下さい。コピペ可能

|  |
| --- |
| <div class="row">  <div class="col-lg-6 col-md-6 col-sm-6">  </div>  <div class="col-lg-6 col-md-6 col-sm-6">  </div>  </div> |

次に、分割したそれぞれの領域、A及びBを表示、非表示がプログラムから制御できるように、Panelコントロールを使用します。

下図のように、Panelコントロールを配置して下さい。ツールボックスから２つ、続けてドラッグアンドドロップして下さい。

双方のPanelコントロールのIDは、下表のコードのようにします。赤色のコード部分、コピペ可能です。

|  |  |  |  |
| --- | --- | --- | --- |
| <div class="row">  <div class="col-lg-6 col-md-6 col-sm-6">  <asp:Panel ID="PanelDate" runat="server"></asp:Panel>  </div>  <div class="col-lg-6 col-md-6 col-sm-6">  <asp:Panel ID="PanelStatus" runat="server"></asp:Panel>  </div>  </div> | Panelコントロールを配置して、その中にコントロール、文字、その他のオブジェクトを配置する場合は、Panelを配置した直後の下記の状態から   |  | | --- | | <asp:PanelD="PanelState"runat="server"></asp:Panel> |   要素の終わりの<の前の位置（赤い｜）の部分にマウスをポイントして、キーボードのリターンキーを押下します、下図のようになります。   |  | | --- | |  |   上図の空白の部分にオブジェクトを配置します。現在の解説では、Webサーバーコントロールを要素として挿入しています。  または、デザインビューのPanelコントロールに、ツールボックスから、直接ドラッグアンドドロップして、配置出来ます。 |

<div class=”row”は、幅12/12までを抱合出来ますが、それ以上になる場合上記の選択されたコードの例のように、再度<div class=”row”のマークアップ（記述）によって新たな領域を確保して、列の構成を設定します。

次の“row”の定義部分と、概念図によって、ページ内の列の構成の理解を深めて下さい。

### ページの列構成を理解する

CSS　“row”の定義

|  |
| --- |
|  |

下図によって、列の構成（デザイン）を切り替える方法と理論が理解できるでしょう。

|  |
| --- |
| 上の3列のデザイン、続けて下に2列のデザインをする場合  同じ<div class=”row”の定義を引き継ぐことにより、画面上の左右の空き等のズレを防ぐ。（margin-right: -15px;  margin-left: -15px;）  <div class=”col-mg-6” ✕ 2 = 12  <div class=”row” 再度マークアップ  <div class=”col-mg-4” ✕ 3 = 12  <div class=”row” |

### 複数のコントロールはテーブルで配置

コードビューの下記の位置に下図をプログラムします。（青色の四角形の部分）

|  |
| --- |
|  |

これからプログラムする範囲は“PanelDate”の中である事を認識しておいてください。

下図は、PanelDateの中にコントロール配置後の実行画面です。

|  |
| --- |
|  |

上図の部分をプログラムします。

上図のような多数のコントロールを整然と配置するためには、“テーブル”を使用します。

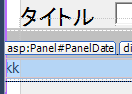
下記の、<div class=”row”の直上の位置に“<hr>”とタイプします、すると自動的に<hr />というコードが挿入されます。（水平線を挿入するためのコードです）

|  |
| --- |
|  |

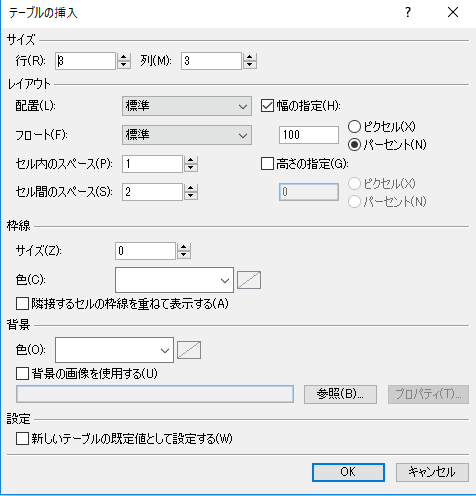
次に、テーブルを挿入するhtmlコードの位置に“ｋｋ”という文字を入力して、目印にします。下記コード

|  |
| --- |
|  |

デザイン画面では、“ｋｋ”部分が下図のように表示されます。



上図の“ｋｋ”の直後に、3行3列のテーブルを挿入します。次図のように“行と列”に入力します。



「OK」ボタンをクリックします。

テーブルの左列の1行目にLabelコントロールとTextBoxコントロールを配置します。

続けて、中央の列に同じようにLabelコントロールとTextBoxコントロールを配置します。

右列には、Buttonコントロールを配置します。テーブルの1行目は、下図のようになります。



次に、テーブルの2行目の左列と中央の列に“Calendar”コントロールを配置します。下図

|  |
| --- |
|  |

上図のテーブルの1行目の、1列目と2列目の、LabelとTextBoxの間に、前項の“コントロール間に隙間を空ける”の項を参考にして、適当に隙間を空けて下さい。各コントロールのIDプロパティの設定は後で行います。

この状態で、ブラウザーで表示、実行して下さい。

下図のようになっています。

|  |
| --- |
|  |

実行画面を閉じてください。

目印の“ｋｋ”を削除します。

メモ：このような細かい作業をするときは失敗を恐れず、思い切って作業をすることです、VisalStudioでは、回復機能があり、以前の状態に戻すには、作業画面の一番上のメニューバーの下に、ツールバーがあります。その中の下図のツールに注目して下さい。

|  |
| --- |
|  |

上図の赤の四角で囲んだツールをクリックすると、直前の操作に戻ります。複数回操作を戻すことが出来ます。是非試してください、この機能を信用して、失敗を恐れず思い切って様々な操作をして下さい。

また、各コントロールのプロパティを設定する場合、当該コントロールを選択したままで、ソリューションエクスプローラ内の、プロパティウインドウを表示して、プロパティを設定して下さい。

次に、今配置した、各コントロールのプロパティを下表に示します。

|  |  |  |
| --- | --- | --- |
| **コントロール名** | **プロパティ名** | **値** |
| 左列　Label | ID | LblStartDay |
| Text | “開始予定日” |
| 左列　TextBox | ID | TxtStartDay |
| 中列　Label | ID | LblEndDay |
| Text | “終了予定日” |
| 中列　TextBox | ID | TxtEndDay |
| 右列　Button | ID | BtnDateOK |
| Text | “次へ” |
| 左列　Calendar1 | ID | CalStart |
| SelectedDayStyle  　　　　BackColor |  |
| TodayDayStyle  　　　　BackColor | Gold |
| 右列　Calender2 | ID | CalEnd |
| SelectedDayStyle  　　　　BackColor | CalStartと同じ |
| TodayDayStyle  　　　　BackColor | CalStartと同じ |

次に、PanelStatusの下記の位置に“5行2列”のテーブルを挿入します、青色の四角部分、行番号は開発環境によって多少違います。

|  |
| --- |
|  |

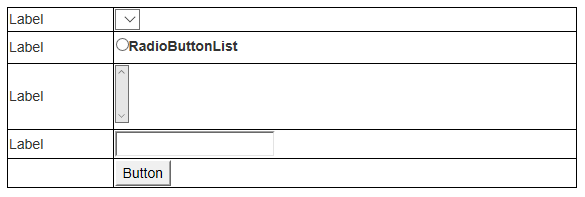
メモ：**テーブル内にコントロールを簡単に配置する方法**

下図のように３行２列のテーブルがあるとします。左列の１行目にコントロールを配置する場合下図のように、コントロールを配置するセルを選択状態にして、ツールボックスの目的のコントロールをダブルクリックします、選択したセルにそのコントロールが瞬時に配置されます。

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |

他の行、列に配置する場合も、配置したいセルを選択状態にして、ツールボックスのコントロールをダブルクリックします。

テーブルを配置したら、下図のようにコントロールを配置して下さい、解り易いように行と列に線を入れています。



TextBox

ListBox

DropDownList

上図のようにコントロールを配置したら次のプロパティ表のように各コントロールのプロパティを設定して下さい。

各コントロールのプロパティ表

|  |  |  |
| --- | --- | --- |
| **コントロール名** | **プロパティ名** | **値** |
| 左列　Label　（上から順） | Text | “状態” |
| 同上 | Text | “優先度” |
| 同上 | Text | “分類” |
| 同上 | Text | “メモ” |
| 右列　DropDownList | ID | DDLStatus |
| 右列　RadioButtonList | ID | RBLPriority |
| RepeatDirection | Horizontal |
| Items | メンバー名：　低、中、高  Valueもメンバー名と同様  この後“RadioButtonListのItem設定”の項で、説明しています。 |
| 右列　ListBox | ID | LBCategory |
| 右列　TextBox | ID | TxtMemo |
| width | 300px |
| height | 60px |
| TextMode | MultiLine |
|  | Htmlコードで日本語入力にする。  ２０１９年現在CSS5のバージョンでは、Ime-modeは使用できません  style="ime-mode:active;" |
| 右列　Button | ID | BtnRegi |
| Text | 登録 |

左列の各LabelのIDプロパティは設定しません。プログラミングにおいて、なんの影響もありません。

### テーブルのプロパティ設定でこんなに変わる

前項で、各コントロールのプロパティを設定したところで、ブラウザーで表示、実行して下さい。実行画面の“状態”、“優先度”等の表示がもう少し右にあれば見栄えも良くなるように思います。

実行画面をとじて下さい。

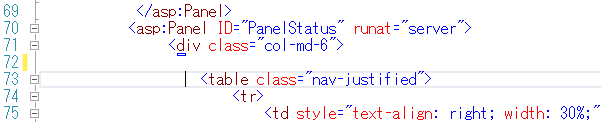
デザインビューで、先程のテーブルの左列の最上部にマウスを移動して、下向き矢印を表示して、styleプロパティを下図のように設定して下さい。

注　値列の％の単位名を間違えないようにして下さい。

テーブルのプロパティ表

|  |  |  |  |
| --- | --- | --- | --- |
| **列の位置** | **カテゴリ** | **プロパティ名** | **値** |
| 左列 | ブロック | text-align | right |
| 左列 | 配置 | width | 30% |
| **テーブルの右列のstyleプロパティ** | | | |
| 右列 | ボックス | padding left | 15% |
|  | | | |

上の表の通りプロパティを設定したら、コードビューの下記の位置のテーブルタグに移って、



下記のように変更して下さい。

<table class="table">　　（bootstrapのテーブル定義です）

ブラウザーで表示、実行して下さい。

変化に、気づかれたら、実行画面を閉じます。

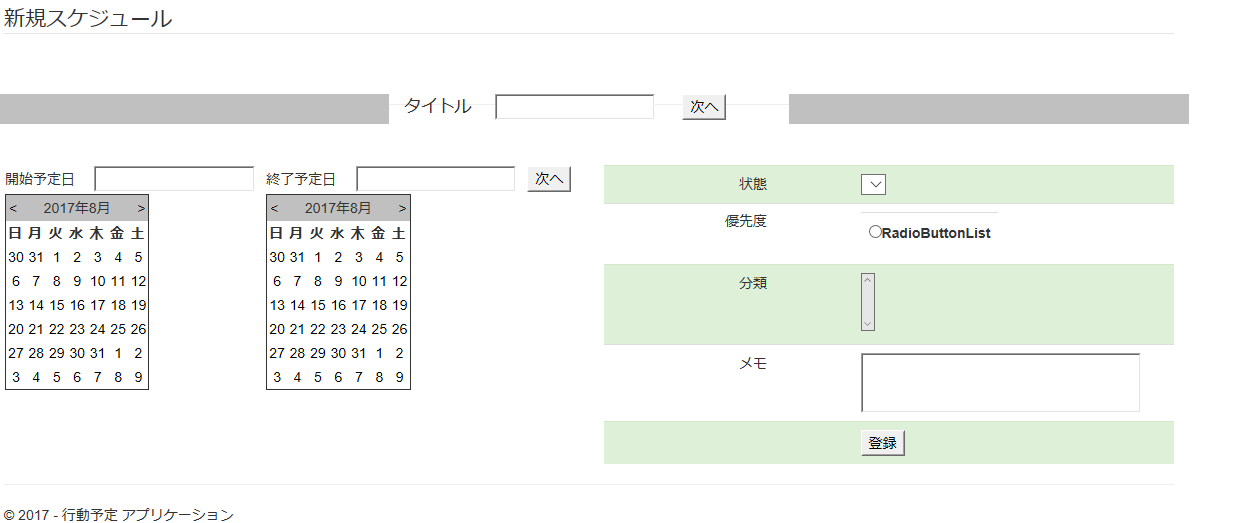
更に、テーブルのデザインをhtmlコードから、変更します。下記のようにコードを追加して下さい。赤い楕円の部分。行番号は開発環境によって、多少の違いがあります。

|  |
| --- |
|  |

上記のコードは、テーブルの行の背景色を1行おきに設定しています。“class=”success”はbootstrapのテーブルの行の背景色を指定する定義です。他に”class=”info”等の定義があります。

Bootstrapについてのサイト　http://bootstrap3.cyberlab.info/

ここまでの時点で、実行画面は下図のようになっています。



# プロフェッショナルなプログラムする

## パネルコントロールでユーザーを導く

これまでのプログラムで、コントロールの配置を主に行ってきました、そのことによりコードビューには、120行程のコードが自動で追加されたり、プログラマー自身が入力しました。その中で、Panelコントロールをhtmlコード内に挿入するように、追加しました。

その意味をこれから体験しましょう。下図をみてPanelの位置を把握して下さい。



PanelDate

PanelStatus

コードビューの“PanelDate”を選択して、Visibleプロパティを”False”にして下さい。

次に、“PanelStatus”も同じようにVisibleプロパティを”False”にして下さい。

ブラウザーで表示、実行して下さい。下図



上図のように、PanelDateとPanelStatusの中の各コントロールも表示されていません。

このような、Panelコントロールの使用法を是非マスターして下さい。

実行画面を閉じます。

次に、デザインビューの“タイトル”入力部分のボタンコントロール“BtnTitle”をダブルクリックして、プログラムコードファイルに、”BtnTitle＿Click”イベントハンドラーが作成されていることを確認して、その中に下記のコードを追加します。赤のコードコピペ可能

|  |
| --- |
| Protected Sub BtnTitle\_Click(sender As Object, e As EventArgs) Handles BtnTitle.Click  Me.PanelDate.Visible = True  End Sub |

追加したら、コードビューまたはデザインビューから、ブラウザーで表示して、実行して下さい。

TxtTitleに“テスト”と入力して、「次へ」ボタンをクリックして、PanelDateが表示されて、その中の各コントロールも表示されることを確認して下さい。

実行画面を閉じてください。

## カレンダーコントロールの日付処理

まず、Calendarコントロールの日付をクリックした時のイベントハンドラーを作成します。

CalStart及びCalEnd　双方をダブルクリックして、イベントハンドラーを作成して下さい。

下記コードをそれぞれの、イベントハンドラーに追加します。赤色のコード、コピペ可能

|  |
| --- |
| Protected Sub CalStart\_SelectionChanged(sender As Object, e As EventArgs) Handles CalStart.SelectionChanged  Dim StartDay As Date = DateValue(Me.CalStart.SelectedDate)  　　　　　　‘カレンダーの日付がクリックされ、その日付を変数“startDay”に格納  Me.TxtStartDay.Text = Format(startDay, "D")  　　　　　　‘変数をFormat関数によって、日本語表記で表示する  End Sub  Protected Sub CalEnd\_SelectionChanged(sender As Object, e As EventArgs) Handles CalEnd.SelectionChanged  Dim EndDay As Date = DateValue(CalEnd.SelectedDate)  　　　　　　　‘カレンダーの日付がクリックされ、その日付を変数“EndtDay”に格納  Me.TxtEndDay.Text = Format(EndDay, "D")  　　　　　　　　　‘変数をFormat関数によって、日本語表記で表示する  End Sub |

技術情報：　[定義済みの日付/時刻書式 (Format 関数)](https://docs.microsoft.com/ja-jp/office/client-developer/access/custom-date-and-time-formats-for-the-format-function-access-custom-web-app)

参考記事

|  |
| --- |
| 解説  書式指定文字列では、大文字小文字を区別します。大文字と小文字を使い分けることで、別の書式を取得できます。たとえば、文字列 "D" で日付の書式設定を行うと、(現在のロケールに応じた) 長い形式の日付が取得されます。ただし、"d" に変更すると短い形式の日付が取得されます。また、定義された書式文字列の大文字と小文字に書式が一致しない場合は、予期しない結果またはエラーが発生することがあります。 |

Me.TxtEndDay.Text = Format(EndDay, "D") は、“長い日付形式”（2019年4月15日）という形式で表示します。

Me.TxtStartDay.Text = Format(StartDay, "d")は、“短い日付形式”（2019/04/15）という形式で表示します。

ここで、ブラウザーで表示、実行して、それぞれのカレンダーの日付をクリックして確認して下さい。

実行画面を閉じます。

次に、BtnDateOKをダブルクリックして、“BtnDateOK＿Clck”のイベントハンドラーを作成して下さい。下記のように”PanelStatus”のVisibleをTrueにします。赤色のコード、コピペ可能

|  |
| --- |
| Protected Sub BtnDateOK\_Click(sender As Object, e As EventArgs) Handles BtnDateOK.Click  Me.PanelStatus.Visible = True  End Sub |

ブラウザーで表示、実行して下さい。下図のように表示されます。

|  |
| --- |
|  |

だいぶ、アプリケーションらしくなりました。

実行画面をとじます。

# データベースを実用化する

本章の「サイトにデータベースを構築する」項の中で、既にテーブルは作成してあります、下図、サーバーエクスプローラのテーブルフォルダをオープン

|  |
| --- |
|  |

テーブル“Status”のデータをDropDownListコントロールに表示して、選択できるようにプログラムします。

まず、上図の“Status”テーブルをマウスで右クリックして、メニューから、「テーブルデータの表示」を選択します。表示された、Statusテーブルのデータを編集する表が表示されますので、”Item”列の一番上に“未着手”と入力して下さい、次に同列の2行目に、“開始”と入力して、一行下の“NULL”の部分をクリックして下さい。ID列に自動でID番号が割り振られたことを確認して下さい。下図

|  |
| --- |
|  |

IDに自動で連番を割り振ることについて、理解を深めるために、このテーブルを作成した時のデザイン画面をもう一度表示ます。一番下の行が下図のように変化しています、

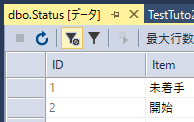
|  |
| --- |
|  |

上図の赤楕円で囲んだ部分のコードによって、IDの番号が自動で＋１づつ増やされてID列に挿入されています。このことは、“テーブルのプロパティの設定”で「データベースシステムが自動で連番を割り振るようにするために“IDENTITYの指定”プロパティを“True”にして、データが追加される毎に、最初のデータのID値は（IDENTITYシード=１プロパティにより）1にして、、2行目以降のID値を前行のIDの値に、（IDENTITY増分= 1　プロパティにより）”1“を加算します。」の説明の内容が反映されています。

ASCの意味は、**データを抽出する時に**、昇順（小から大へ）という意味です。反対にDESCは降順（大から小へ）という意味です。

続けて、下図のようになるようにテーブルにデータを挿入して下さい。

|  |  |
| --- | --- |
|  | 複数行入力する時は、文字の入力後リターンキーを押下すると、下の行に入力位置が移動します。 |



上図の黄色い表示の部分の✕印をクリックして、データの挿入を終了して下さい。

これでStatusテーブルにデータが登録され、それを利用することが出来ます、

## Dataコントロールを使う

### Dataコントロールとは

Dataコントロールとは、ツールボックスの「データ」タブにまとめられているWebサーバーコントロールです。Dataコントロールには、データソースとして、データベースなどからデータを取得する役割を持つコントロール（DataSourceコントロール）と、取得したデータを表示する表示系のコントロールがあります。

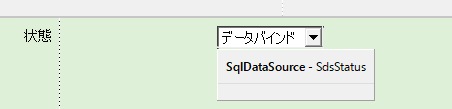
例えば、データソースコントロールの中のSqlDataSourceコントロールは定義したSQLステートメント（SQL文）に従って、データベースからデータの取得や変更等の操作を行います。また表示系のコントロールの内GridViewコントロールや、FormViewコントロール、DropDownListコントロール等は関連つけされたデータソースコントロールが取得したデータを表形式、リスト形式、カード形式等で表示したり、ユーザーからの変更内容をデータソースコントロールへ渡したりします。

とにかく、実践を通して理解しましょう。

### DropDownListBoxにデータソースを関連つける

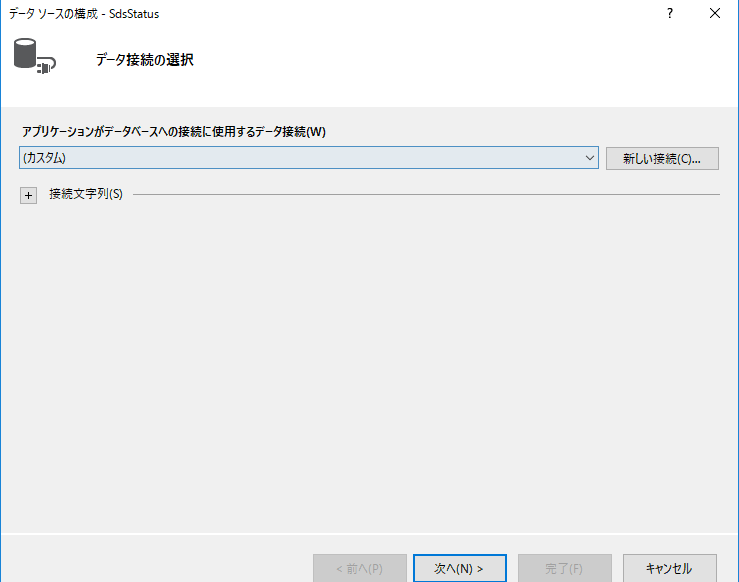
デザインビューの状態、優先度などが表示されている、テーブルのDDLStatusの右側にデータコントロールの“SqlDataSource”コントロールをドラッグアンドドロップして下さい。

下図が配置後の画面



SqlDataSourceのIDを“SdsStatus”にします。

SdsStatusが選択状態の場合、右肩の＞印をクリックすると、タスクメニューが表示されます。「データソースの構成」をクリックします、下図が表示されます



myDB.mdf

「新しい接続」ボタンの左側のドロップダウンを開いて、“myDB.mdf”を選択します。

ダイアログウインドウの下の「次へ」ボタンをクリックします。

注：このWebプロジェクト（Webサイト）では、初めてのデータソースの構成作業ですので、ConnectionStringの命名をVisualStudio 側から催促しているのです。

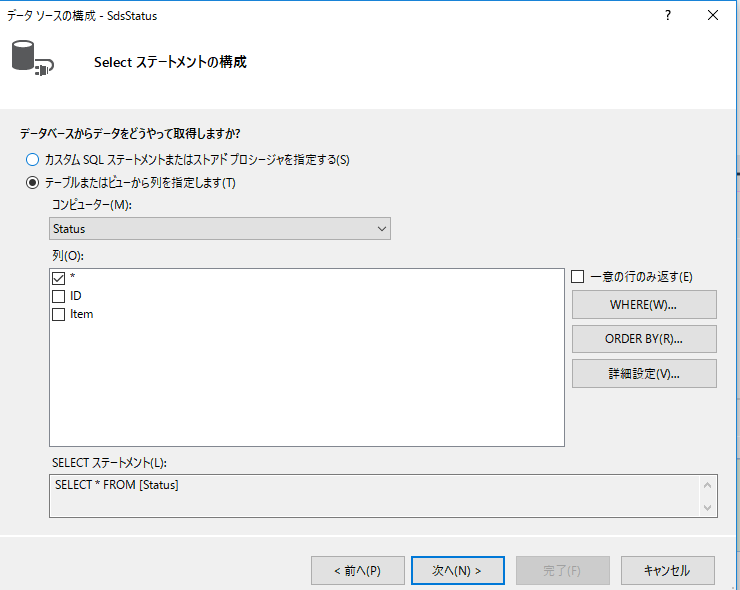
表示された、ダイアログウインドウの下図の部分“ConnectionString”を“SiteSqlString”と英字で入力します。



SiteSqlString

“SiteSqlString”と入力したら、「次へ」ボタンをクリックします。

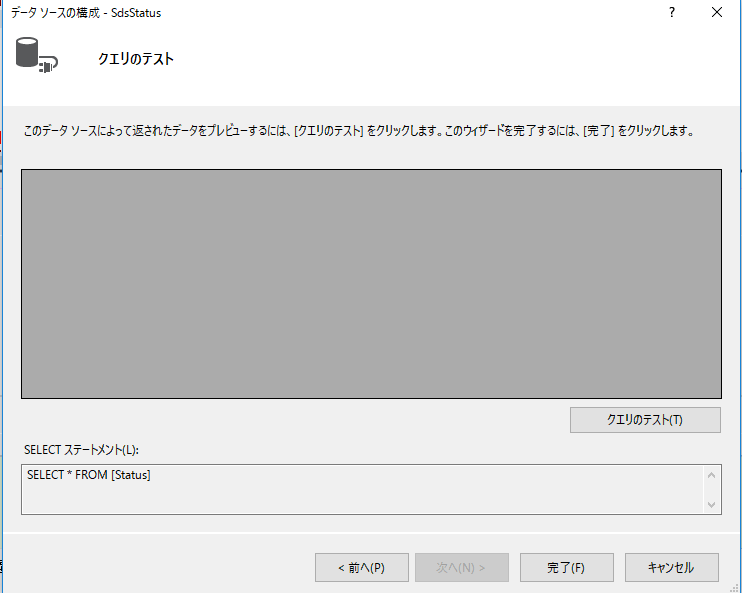
下図のダイアログウインドウが表示されます。



既に、全ての列を選択する、SQL文が作成されています。

このままの状態で、「次へ」ボタンをクリックします。

下図が表示されます。



「完了」ボタンをクリックします。

次に、DDLStatusを選択状態にして、右肩の＞印をクリックして、「データソースの選択」をクリックします、下図が表示されます。



「データソースの選択」のドロップダウンを開いて、「SdsStatus」を選択します、

下図のように設定して、

|  |
| --- |
|  |

データテーブルStatusのデータ“Item”の項目が表示されない場合、上図の“スキーマの更新”をクリックします。

「OK」ボタンをクリックします。

ここで、ブラウザーで表示、実行して、下図のように、DDLStatusに“Statusテーブル”の内容が表示されていることを確認します。

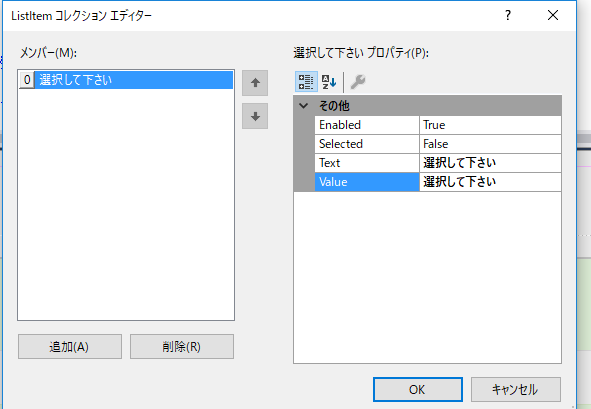


DDLStatusのドロップダウンを開いて、Statusテーブルの全てのデータが表示されることを確認して下さい。

実行画面を閉じます。

### DropDownListコントロールに“選択して下さい”を表示する

DDLStatusを選択状態にして、プロパティウインドウを表示して、「AppendDataBoundItems」プロパティをTrueにします。次に、「Items」プロパティの“コレクション”を開いて、「追加」ボタンをクリックして、下図のように設定します。



「OK」ボタンをクリックしてコレクションエディターを終了します。

ブラウザーで表示、実行して下さい。　下図のようにDDLStatusに“選択して下さい”が表示されます。

|  |
| --- |
|  |

この状態の場合、DDLStatusのValueは“選択して下さい”という、Stringであることを認識して下さい。

DDLStatusの現在のValue値が“選択して下さい”という文字列であれば、ユーザーはDDLStatusから何も選択していないことを意味しています。プログラミングにおいて、この状態をプログラムの条件式に利用することの手法を度々使用します。

実行画面を閉じます。

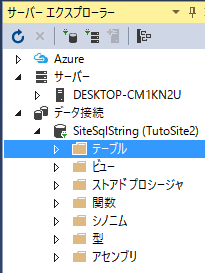
### ListBoxにデータソースを関連つける

次に、スケジュール予定が、仕事上なのか、個人的な事なのか、を選択するListBoxの項目もデータベースから設定するようにします。

分類の選択項目を表示する、LBCategoryに関連つけるデータベースのテーブルはまだ作成していませんので、これから作成します。

### 新しいデータベースのテーブルの作成

サーバーエクスプローラを表示して、下図のように“SiteSqlString”の配下のテーブルフォルダを表示します。



DDLStatusに関連つけるSqlDataSourceを作成した際にmyDB.mdfのConnectionStringを

“SiteSqlString”にしましたので、この表示に変わっています。

テーブルフォルダを右クリック、表示されたメニューの「新しいテーブルの追加」をクリックします、デザイン編集を下図のようにします。

|  |
| --- |
|  |

デザイン編集欄の下の、T-SQL編集欄のSQLステートメントは、自動で下図のようになっています。

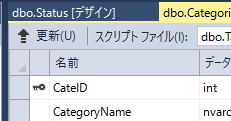
|  |
| --- |
|  |

T-SQL編集欄のSQLステートメントを次のように編集します。Statusテーブルの時のプロパティからの設定の方法と、このように直接コードの編集でもできます。赤のコードコピペ可能です

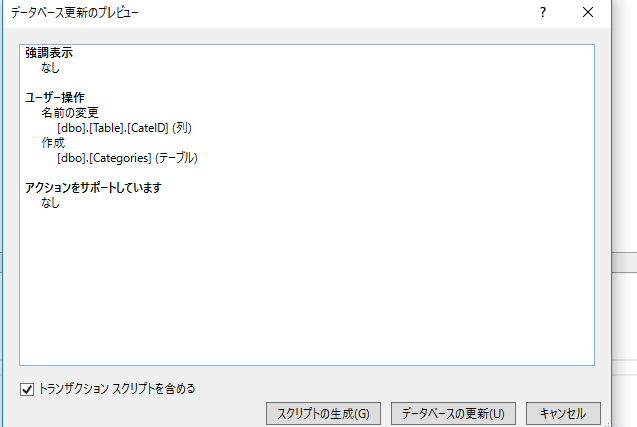
|  |
| --- |
| CREATE TABLE [dbo].[Categories]  デフォルトのT-SQLすべて削除して、新しいT-SQLをコピペする方法があります。  (  [CateID] INT IDENTITY (1, 1) NOT NULL,  [CategoryName] NVARCHAR (50) NOT NULL,  PRIMARY KEY CLUSTERED ([CateID] ASC)  ) |

ステートメントの意味は、テーブル名が“Categories”で、CateID列を主キーにして、データが登録される毎にCateIDの数を＋１づつ増やす。”NOT NULL” はCateID、CategoryName共にNull（空白）を許可しない。という意味です。

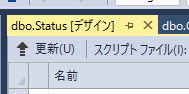
画面の左上の更新ボタンをクリックします、下図



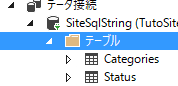
下図が表示されます。



「データベースの更新」ボタンをクリック、VSのドキュメントウインドウの最下部に「データツール操作」ウインドウが表示されます、“更新が正常に完了しました。”という表示を確認して、デザイン画面を閉じます。次図

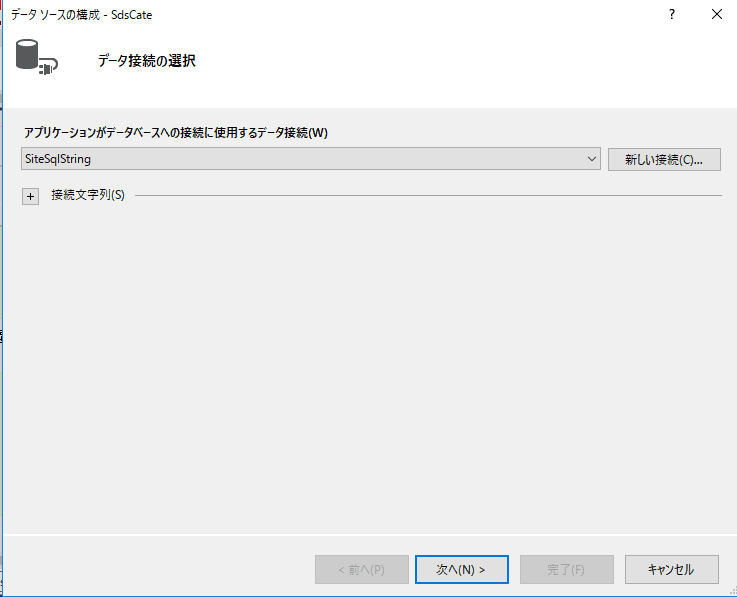


再び、サーバーエクスプローラの“SiteSqlString”の配下の「テーブル」フォルダを、右クリックして、表示されたメニューの上から3番目の「最新の情報に更新」をクリックします。下図のように、”Categories”テーブルが出来ています。



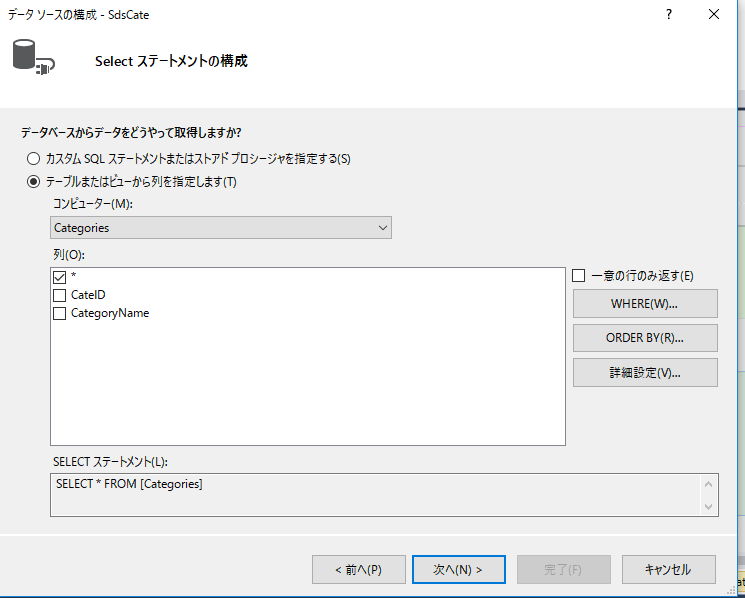
データソースとWebサーバーコントロールと関連つける方法において、前項のDropDownListコントロールに関連つけた方法と同じ手順で、今回は、ListBoxコントロールに関連つけましょう。

デザインビューのLBCategoryの右側に、ツールボックスの“データ”タブの中の“SqlDataSource”をドラッグアンドドロップして下さい。IDプロパティを”SdsCate”にします、SdsCateを選択状態にして、右肩の＞印をクリック、表示された”データソースの構成“をクリックします。下図のように「新しい接続。ボタンの左側のドロップダウンを開いて、”SiteSqlString”を選択します。



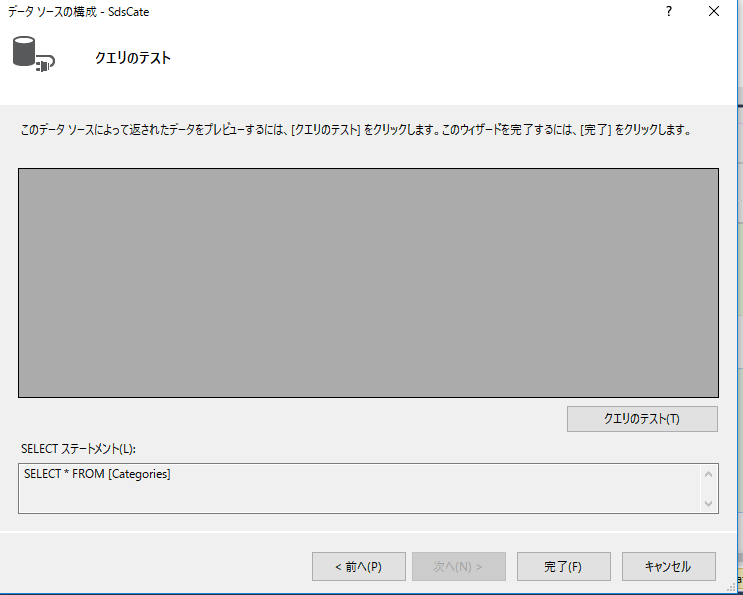
「次へ」ボタンをクリックします。

下図の状態で



「次へ」ボタンをクリックします。

下図が表示されます。



「完了」をクリックします。

### ListBoxコントロールにデータソースを関連つける

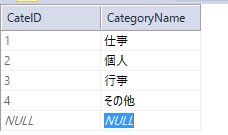
次に、LBCategoryを選択します、右肩の＞印をクリックして、「データソースの選択」をクリックして、下図のように設定します。

|  |
| --- |
|  |

「OK」ボタンをクリックします。

### データベーステーブルに手入力でデータを入れる

前項の「データベースを実用化する」で説明したように、今度は“Categories”テーブルのデータを表示して、下図のように、データを挿入してください。



“仕事”と入力してから、リターンキーを押下すると“個人”の行にマウスキャレットが移動して入力が楽に出来ます。CateIDも自動で割り振られます。

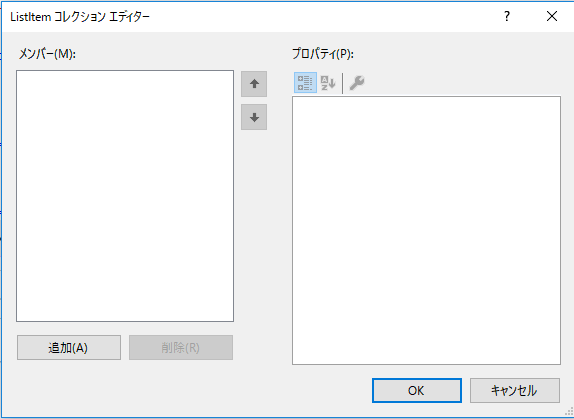
入力を終了して、上部の✕印をクリックします。

ここで、ブラウザーで表示を実行して、LBCategoryにデータ（項目）が表示されるのを確認して下さい。

実行画面を閉じます。

## RadioButtonListのItem設定

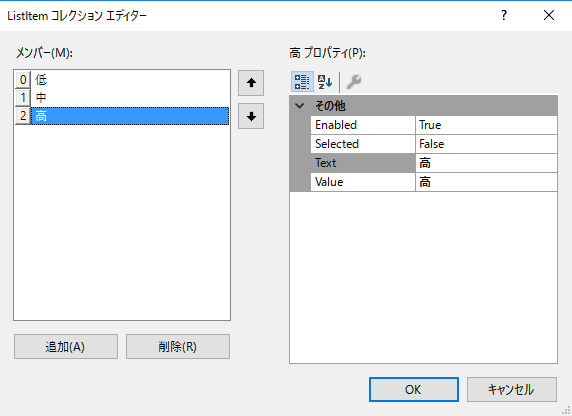
RadioButtonList“RBLPriority”が選択されている状態で、プ右肩の＞印をクリックして、RadoButtonListタスクメニューを表示させて、最上部から2行目の“項目の編集”をクリックします、下図のようにコレクションエディターが表示されます。



上図の、“追加”ボタンをクリクします、そのボタンの上部に“0 ListItem”という項目が表示されます、右側の“Text”という項目の右側の空白に“低”という文字を入力します。

その“Text”の項からマウスのフォーカスが外れると”Value“の項目に自動で”低“と表示されますが、そのままにしておきます。

次に、再び“追加”ボタンをクリックして、先程と同様に“Text”を”中“と入力、続けて追加ボタンをクリックして、”Text“を”高“にします、次図



OKボタンをクリックして終了します。

ブラウザーで表示、実行してここまでの成果を確認して下さい。

実行画面を閉じます。

以上ですべてのWebサーバーコントロールに、プロパティ、表示アイテム等の設定が終了しました。

## 入力必須の処理、検証コントロール

Webページからユーザーがデータを入力する際に、必須項目の入力を求めたり、そのデータの整合性を検証する必要があります。そのための便利なツールがVSに用意されています。

VSのツールボックス内の、“検証”タブに数個の「検証コントロール」があります。下図

|  |
| --- |
|  |

早速使用しながら機能を習得していきましょう。

まずPanelDateの中の各Webコントロールとタイトル入力用のTextBoxについて、入力確認のための検証コントロールの配置とプロパティ設定を行います。

検証コントロールの配置も、ドラッグアンドドロップして必要な場所に配置します。

次図のように各コントロールを配置して下さい。

|  |
| --- |
|  |

上図の“タイトル”入力用のTxtTitleに対する検証コントロール“RequiredValidator”コントロールの配置は、「次へ」ボタンの右横にマウスポインターを置き、リターンキーを押下して、ツールボックスの検証タブから、“RequiredValidator”コントロールをダブルクリック、またはドラッグアンドドロップします。

Htmlコードは下記のように「次へ」ボタンの終了要素の右側に改行要素が追加されています。

|  |
| --- |
|  |

同様にTxtStartDayの右側でリターンキーを押下して、「RequiredFieldValidator」コントロールをダブルクリックまたは、ドラッグアンドドロップして下さい。

Htmlコード

|  |
| --- |
| <asp:Label ID="Label3" runat="server" Text="開始予定日"></asp:Label>  &nbsp;&nbsp;  <asp:TextBox ID="TxtStartDay" runat="server"></asp:TextBox><br />  <asp:RequiredFieldValidator ID="RequiredFieldValidator1" runat="server |

TxtEndDayの場合

|  |
| --- |
| <asp:Label ID="Label4" runat="server" Text="終了予定日"></asp:Label>  &nbsp;&nbsp;  <asp:TextBox ID="TxtEndDay" runat="server"></asp:TextBox><br />  <asp:RequiredFieldValidator ID="RequiredFieldValidator3" runat="server" ………  <br />  <asp:CompareValidator ID="CompareValidator1" runat="server" ErrorMessage="Co |

必ず上記のようにしなければならないことはありません。エラーが発生した時にユーザーに気が付き易い位置を考慮したものです。このようなエラーの表示位置の他にも表示方法がありますので、次のPanelStatusの中の検証で実践します。

## PanelDate内の検証コントロールのプロパティ表

タイトル入力用のTxtTitleと開始予定日のTxtStartDay、終了予定日のTxtEndDayの検証コントロールのプロパティ設定を下表のように設定して下さい。

|  |  |  |  |
| --- | --- | --- | --- |
| **対象コントロールID名** | **検証コントロール名** | **プロパティ名** | **値** |
| TxtTitle | RequiredFieldValidator | ControlToValidate | TxtTitle |
| ErrorMessage | タイトルの入力は必須です |
| ForeColor | Red |
| TxtStartDay | RequiredFieldValidator | ControlToValidate | TxtStartDay |
| ErrorMessage | 開始予定日の入力は必須です |
| ForeColor | Red |
| TxtEndDay | RequiredFieldValidator | ControlToValidate | TxtEndDay |
| ErrorMessage | 終了予定日の入力は必須です |
| ForeColor | Red |
| CompareValidator | ErrorMessage | 開始日以降の日付を設定して下さい。 |
| ForeColor | Red |
| ControlTopCompare | TxtStartDay  (比較する相手となるコントロール名) |
| ControlToValidate | TxtEndDay  （比較元のデータとなるコントロール名） |
| Operator |  |
| Type | Date |

上表のCompareValidatorのOperatorプロパティの“GreaterThanEqual”の意味は、

比較相手のデータと同じか、それ以上であればOK（エラー表示しない）という意味です。日付の場合ですから、同じ日付か、それ以後の日付ならOK（エラー表示しない）という意味です。

次図に、検証コントロールを配置して、プロパティ設定後のデザイン画面を表示します。

デザイン画面

|  |
| --- |
|  |

ブラウザーで表示、実行して下さい。

下図のように、開始予定日と終了予定日を選択して、「次へ」ボタンをクリックします。

|  |
| --- |
|  |

「次へ」ボタンをクリックすると、上図のように「CompareValidator」コントロールがエラーを表示します。

|  |
| --- |
|  |

上図のような、開始予定日以降に、終了予定日が選択されているのに、検証コントロールがエラーを表示する仕組み（原因）を解説します。

## CausesValidationとは？

「次へ」ボタン（ID=BtnateOK）のプロパティウインドウを表示して、「CausesValidation」プロパティに注目して下さい。下図

|  |
| --- |
|  |

上図のように、Buttonコントロール等に、CausesValidationプロパティが定義されています。このプロパティは、「そのコントロール（Button等）がフォーカスを受け取るとページ内の検証が必要なコントロール（“RequiredFieldValidator”とか”ConpareValidator”）に対して,検証を実行するかどうかを示す値を設定します。」

Trueの値は、検証を実行、Falseは検証をしない。

従って今回のButtonコントロールの押下によって、“CausesValidation”プロパティが”True”ですのでぺージ内の、検証コントロールが一斉に検証を開始して“CompareValidator”コントロールだけがエラーを表示しました。

## CompareValidatorのTypeプロパティ

先程のエラーの原因を究明したいと思います。当該の“CompareValidator”の下図のプロパティは下図のようになっています。

|  |
| --- |
|  |

上図の各プロパティを総合して、文章にすると

1. TxtEndDayを元にして（ControlValidate）
2. TxtStartDayに対して（ControlToCompare）
3. 日付の値で比較します（Type）
4. 元になるTxtEndDayの日付がTxtStartDayの日付に比べて同じかそれ以降（GreaterThanEqual）

という事になります。

どこの箇所がエラーを発生しているのか、その原因は、比較する側TxEndDayと比較される側TxStartDayにあります。実行して、いづれかのカレンダーをクリックした時のプログラムの処理で、TxtStartDayとTxtEndDayに日本語式の日付を表示しました。

この日本語式の表示を検証コントロールは日付タイプとして認識しないようです。

日付タイプとして、認識する文字列は“2017/09/20”という文字列です、したがって筆者は比較用のTextBoxコントロールをデザインビューに追加配置して、ダミーに使用しました。

次図のように、TextBoxコントロールを各カレンダーの右側にドラッグアンドドロップして下さい。

|  |
| --- |
|  |

上図のように、各カレンダーの右側にドラッグアンドドロップすると、カレンダーの下側に配置されます。下表のようにプロパティを設定して下さい。

|  |  |  |
| --- | --- | --- |
| コントロール名 | プロパティ名 | 値 |
| 左側　TextBox | ID | TBStart |
| Visible | True |
| 右側　TextBox | ID | TBEnd |
| Visible | True |

次に下記のように、両カレンダーの“SelectionChanged”イベントにコードを追加して下さい。赤の部分

**CalStart\_SelectionChanged**

|  |
| --- |
| Protected Sub CalStart\_SelectionChanged(sender As Object, e As EventArgs) Handles CalStart.SelectionChanged  Dim StartDay As Date = DateValue(Me.CalStart.SelectedDate)  'カレンダーの日付がクリックされ、その日付を変数“startDay”に格納  Me.BStart.Text = DateValue(StartDay)  '検証用のTextBoxコントロールに日付データを代入  　　　　　‘DateVlue関数は“2017/04/15 0:00:00” (DateTime型)を  ”2017/04/15”（Date型）にします  Me.TxtStartDay.Text = Format(StartDay, "D")  '変数をFormat関数によって、日本語表記で表示する  End Sub |

**CalEnd\_SelectionChanged**

|  |
| --- |
| Protected Sub CalEnd\_SelectionChanged(sender As Object, e As EventArgs) Handles CalEnd.SelectionChanged  Dim EndDay As Date = DateValue(CalEnd.SelectedDate)  'カレンダーの日付がクリックされ、その日付を変数“EndtDay”に格納  Me.TBEnd.Text = DateValue(EndDay)  '検証用のTextBoxコントロールに日付データを代入  　　　　‘DateVlue関数は“2017/04/15 0:00:00” (DateTime型)を  ”2017/04/15”（Date型）にします  Me.TxtEndDay.Text = Format(EndDay, "D")  '変数をFormat関数によって、日本語表記で表示する  End Sub |

ブラウザーで表示、実行して下さい。両カレンダーをクリックするところまで実行して下さい。

|  |
| --- |
|  |

上図のように、日本語式日付と、カレンダーの下の、西洋式の日付表示（Date型）が正しく表示されています。

実行画面を閉じます。

上図の後から追加したTextBoxをCompareValidatorの比較の対象にします。

デザインビューを表示して、先程追加したTBStartとTBEndのVisibleプロパティを“False”にして下さい。

次に、CompareValidatorを選択して、プロパティウインドウを表示して、下図の部分を変更して下さい。

|  |
| --- |
|  |

次に、プログラムコードの下記の部分にコードを追加します。以前のコードを削除して、赤色のコードをコピペします。

|  |
| --- |
| Protected Sub BtnDateOK\_Click(sender As Object, e As EventArgs) Handles BtnDateOK.Click  'このクリックイベントによって検証コントロールが機能して  'エラーがない場合は、Page.IsValid　＝True　エラーがある場合Page.IsValid = False　を返します  If Page.IsValid Then　　‘エラーがない場合  Me.PanelStatus.Visible = True　　‘実行する  End If  End Sub |

ブラウザーで表示、実行して下さい。

開始予定日と終了予定日等を変化させて、エラーも発生させて下さい。

エラーがある場合は、プログラムは進みません。

実行画面を閉じます。

TBStart,及びTBEnd 両TextBoxのVisibleプロパティを“False”にして下さい。

## PanelStatus内の検証コントロール

下図のように赤い文字の検証コントロールを配置して下さい。

|  |
| --- |
| ValidationSummaryコントロール |

上のDDLStatusから順に、解説していきます、DDLStatusには検証コントロールは使用しませんが、“AppendDataBoundItems”プロパティは”True” になっています。

|  |
| --- |
|  |

次に、“Items”プロパティの右側の”(コレクション)“は下図のように一つだけ項目を追加してあります、下図のValue値の、”選択して下さい”という文字列をプログラムで利用します。（前項、「DropDownListコントロールに“選択して下さい”を表示する」で解説済）

|  |
| --- |
|  |

次に、「BtnRegi」のValidationGroupプロパティに“Priority”と入力して下さい。重要です。

次に、「BtnRegi」の右側に、ツールボックスの”検証タブの”ValidateSummary“コントロールをドラッグアンドドロップして下さい。そしてプロパティウインドウの下図の2か所のプロパティを設定して下さい。

|  |
| --- |
| Priority  Red |

次に、優先度のRBLPriorityの右側に“RequiredFieldValidator”コントロールを配置して下さい。

次に、分類のLBCategoryの右側に“RequiredFieldValidator”コントロールを配置して下さい。

次の表に、今配置した、両コントロールのプロパティを示します。

|  |  |  |
| --- | --- | --- |
| **コントロール名** | **プロパティ名** | **値** |
| RBLPriorityの  RequiredFieldValidator | Text | ＊  （日本語の＊印） |
| ErrorMessage | 優先度を選択してください。 |
| ValidationGroup | Priority　（手入力） |
| ControlToValidate | RBLPriority |
| ForeColor | Red |
| LBCategoryの  RequiredFieldValidator | Text | ＊  （日本語の＊印） |
| ErrorMessage | 分類を選択して下さい。 |
| ValidationGroup | Priority　（手入力） |
| ControlToValidate | LBCategory |
| ForeColor | Red |

ここまでで、ブラウザーで表示、実行して下さい。

優先度と分類では、何も選択しないで、「登録」ボタンをクリック、エラー表示の状態を確認して、優先度と分類から、何かを選択してエラーが出ないことを確認して下さい。

また、エラーを発生したコントロールの位置に“＊”が表示されることを確認して下さい。

実行画面を閉じます。

DDLStatusの入力確認は、プログラムコードから検証します。

## もう一度!!コンピュータープログラムって難しい？

全然難しくありません。例えばあなたが自動車免許を取得するために、自動車学校へ行った時のことを思い出してください。交通法規、車の構造、走行コースでの運転等全てが初体験で、果たしてスムーズに免許取得が出来るのだろうか、と不安を感じながら自動車学校に通ったことでしょう。でも卒業後には、免許が交付されて初心者マークをつけて、一般道路を不慣れな運転で走行していました。その様な運転を続けた期間は決して短期間ではなかったと思います。半年、一年後には、スピード違反で切符を切られるほどにあなたの運転は一人前になりました。

運転免許取得のための学校に通い、そして一般道路で自動車運転を実践することであなたの脳内の運転に関する知識は膨大な量を蓄積したでしょう。（ある脳科学者談）。

コンピューターのプログラムを習得するのも、全くそれと同じです。私自身、独学で現在、自分のウエッブサイトを立ち上げて自分で作成したホームページを公開しています。

独学でITの知識を習得した経験者だからこそ言えることもあり、習得の近道も知り尽くしていると自負しています。私の講座における本実践解説を複数回、実践することにより、貴方は充分な基礎を身に着けた、Webプログラマーになっています。その様な充分な基礎を身に着けた貴方の将来は、産業用ロボットの開発、IoTの開発、自動運転自動車の開発等の一役を担う程にまで、あなた自身の力（独学）でスキルアップすることが出来ます。もちろんWebショップを運営したい方も短期間でその知識が習得出来ます。

ちなみに、コンピュータ―プログラムというと、理工系の得意な分野の人が向いている、と言った見解がありますが。私はそうは思いません、それは一昔前のコンピューター環境下での話だと思います。コンピュータのプログラムを構成するのに“プログラム言語”という言葉があって、数種類の言語がありますが、“言語”なんです、一例をあげますと、

|  |
| --- |
| If A > B Then ‘もしAがBより大きいなら  A.BackColor = Blue　　‘Aの背景色を青色にしなさい  End If　　　　　　　　　　‘ Ifの条件を終わる |

上記の緑色の説明文のように、全く“言語”の世界ですね。そして、上記の網掛けの部分のように“If ――Then～End If”は、定型文です。その定型文は、何回か使用しているうちに覚えてしまいますし、VisualStudioでは、コード記入時に“インテリセンス”機能といって、プログラマーが記入したい定型文を複数表示してくれます、それを貴方はその表示の中から、マウスで選択すればいいのです（ほんの一例）。

ITの知識を習得するのに専門の学校でお金（一年で１５０万円）をかけて、自由時間も奪われて、行く必要はありません。

一番重要なことは、貴方に“やる気”があるかどうかです。もしあなたが躊躇しているなら、この言葉を贈ります。**Never Try Never Learn**（何もしないで、何も得られません）

急ぐ必要はありません、毎日をどうやって過ごすかという事です、生き物は生きていくために、食物を食べますが、人間は、同時に知識も食べています。その知識の中身は人によって違うでしょう、ある人はテレビのバライティショウの番組で無意味な知識を得、ある人は読書で自分以外の人生を経験し、ある人は、他国の言語を勉強して自分の上達に、達成感を感じたり、という風に…。

私は、このITの勉強を独学で始めたときに、最初はプログラムコードがピンとこなくて霧の中を歩いているようでした。相当な期間霧の中を歩いているようでしたが、長続きした理由を今考えると、先程の“知識を食べる”の中の後者の“達成感”によって後押しされて長続き出来たように思います。独学ですから私の場合毎日がパズルを解くようなゲーム感覚でした。パズルが解けたときはその達成感に感動して次のパズルに挑戦する力が湧いてくるような感覚でした、でも読者の皆様、私の場合はパズルでしたがこれからの私の講座、チュートリアルの説明は、パズルではありません、初心者の誰しもがすぐ理解できるように丁寧に解説していくつもりです。そして今の時代は、インターネットの発達により氾濫するほどの情報がインターネット上にあり、自分に必要な情報を効率よく探し当てることも重要です、そのような必要な情報を紹介しながら、チュートリアルを進めます。したがって読者様はスムースに独学を効率よく進めて行くことが出来るでしょう。そのノウハウもこの講座を実践していくほどに身に着けることが出来るでしょう。

最後に、VisualStudio という開発ツールは、コンピュータ―プログラム開発において世界最高レベルの開発能力のあるツールです、マイクロソフトは、日々このツールのアップデート（更新）に努めており、プログラミング効率と使用し続ける将来に対する信頼性は疑いの余地はありません。こんな高機能な開発環境を無料でダウンロードを容認するのもマイクロソフトの製品の自信を感じます。もちろん有料の製品もあります、それはさらに高機能です。

私が、始めたころ、VisualStudioを25年前、30万円で購入しました。その時の私の想いは、コンピュータープログラマ―になろうとは思っていませんでした。暇な時にパチンコ屋によく行ったものでした、何回か行くうちに空しさを感じ始めて、何か趣味でも身につくものがないかと、考えてコンピュータープログラミングを始めた次第です。

読者の皆様もいい趣味を持ちましょう。そして、毎日知識をたべて、達成感を味わいましょう。人生がかわります、ハイレベルな達成感で幸福になれます。

## 「登録」ボタンクリックイベントのプログラム（１）

ToDo.aspxページに戻ります。

デザインビューの「登録」ボタンの右側にLabelコントロールをドラッグアンドドロップして下さい。IDプロパティを“LblOK”にして、Visibleプロパティは、Falseにして、ForColorプロパティを“Red”にして下さい。Textプロパティは、” **状態の項目が選択されていません**“　と入力して下さい。

デザインビューの「登録」ボタンをダブルクリックしてBtnRegi\_Clickのイベントハンドラーを作成します。そのBtnRegi\_Clickイベントハンドラー内に、下記のコードを追加します。

|  |
| --- |
| Protected Sub BtnRegi\_Click(sender As Object, e As EventArgs) Handles BtnRegi.Click  'DDLStatusの有効な値を検証します  If Me.DDLStatus.SelectedValue = "選択して下さい" Then  Me.LblOK.Visible = True  Exit Sub 'ユーザーがドロップダウンリストから選択ミスの場合  　　　　　'ここでこのSubプロシージャを終了して,プログラムを中止  ElseIf Me.DDLStatus.SelectedValue <> "選択して下さい" Then  Me.LblOK.Visible = False  End If　　　‘ここまででDDLStatusの検証を終了  　　　　If Page.IsValid Then　　'入力ミスがない場合PageはIsValid=Trueを返す  　　　　　　Me.LblOK.Visible = True　　‘LblOKを表示して  Me.LblOK.Text = "Page.IsValid = " & Page.IsValid.ToString()  　　　　　　　　　　‘Page.IsValidの値を確認するために、LblOKに表示する  End If  End Sub |
| **解説**  　　If Me.DDLStatus.SelectedValue ＝ “選択して下さい” Thenについて  　　　　　DDLStatusのプロパティで“AppendDataBoundItems”をTrueにしました。  　　　　　そして、次図のように“選択して下さい”をItemsコレクションに追加しました。    　　　上図のValueプロパティは“選択して下さい”になっています。  　　　したがって、If Me.DDLStatus.SelectedValue = "選択して下さい" Then  　　　（もし、DDLStatusの現在選択されているValue（値）が"選択して下さい"ならば）、という式が成り立ちます。  もし、DDLStatusの現在のValueの値が"選択して下さい" ならば  　　　LblOKのVisible をTrueにして表示して、プログラムを中止します。  DDLStatusの現在のValueの値が"選択して下さい" 以外ならば  　　LblOKのVisible をFalseにして表示しません。  そして、If　の条件式を終わります。End If  If Page.IsValid Then　　'入力ミスがない場合PageはIsValid=Trueを返す  　　‘何もエラーがない場合、ここのプログラムコードを実行する  　　Me.LblOK.Visible = True  Me.LblOK.Text = "Page.IsValid = " & Page.IsValid.ToString()  　　　‘Page.IsValidの値を確認するために、LblOKに表示する  End If |

ここでブラウザーで表示、実行して色々な状況を確認して下さい。

実行画面を閉じます。

メモについては、ユーザーがメモを必要としない（書かない）場合があるため、空白の場合にどうするか等、プログラムコードで処理します。

## スケジュール（予定表）を保存する

せっかく立てたスケジュールですから、保存して、一覧表を作成したり、スケジュールの変更をしたり、いろいろ活用したいものです、そのためには、データベースの機能が不可欠です。

Column

|  |
| --- |
| なぜデータベースか＿？ |

### スケジュールデータ登録用のデータベーステーブルの作成

下表のようなテーブルを作成します。読者様もレベルがあがって、下図のTIDのプロパティ設定、デザイン画面、T-SQLステートメント、でテーブルを作成することが出来るでしょう。

テーブル、Todoのデザイン

|  |  |  |
| --- | --- | --- |
|  | 下記の赤色のコードをデザイン画面の、T-SQLタブの中に、コピペして下さい。   |  | | --- | | CREATE TABLE [dbo].[Todo]  (  [TID] INT IDENTITY (1,1) NOT NULL,  [Title] NVARCHAR (50) NOT NULL,  [Status] NVARCHAR (50) NOT NULL,  [Priority] NVARCHAR (50) NOT NULL,  [Category] NVARCHAR (50) NOT NULL,  [Startday] DATE NOT NULL,  [Endday] DATE NOT NULL,  [Memo] NVARCHAR (100) NULL,  [Deleted] BIT NOT NULL,  PRIMARY KEY CLUSTERED ([TID] ASC)  ) | |

「Todo」データテーブルを作成して下さい。

スケジュールのデータを登録するためのデータテーブルが出来ました。

早速、プログラムしましょう。

### ページを追加デザインする

まず、登録されたスケジュールの一覧表として表示するためには、前項“データコントロールとは”で紹介しました。取得したデータを表示する表示系のコントロールを使用します。データ表示系のコントロールで、代表的なのは、ツールボックスの“データ”タブにある“GridView”、“FormView”、”DataList”、等があります。今回は“GridView”を使用します。

表示位置を下図の位置に表示するようにhtmlコードを追加します。

|  |
| --- |
| <class=”col-md-4”>  <class=”col-md-8”>  スケジュール一覧表 |

PanelIchiran

前図のレイアウトを作成するために、順を追って、下記のように、htmコードを追加します。

1. </asp:Content>の直上に新規のページ幅を範囲に持つ、<div class=”row”>を作成する。

|  |
| --- |
|  |

1. <div class=”row”の中に全体を一列とする<div class=”col-lg-12 col-md-12 col-sm-12>～</div>というコードを挿入する。

|  |
| --- |
|  |

1. 列の中に、Panelコントロールを挿入します。

|  |
| --- |
|  |

1. デザインビューとhtmlコードビューと同期して、デザイン画面のPanelコントロールの中に、ツールボックスの「データ」タブの中のGridViewを配置して下さい。（ドラッグアンドドロップして下さい。）

|  |
| --- |
| この中へ、GridViewコントロールを挿入 |

結果は下図のようになります。

|  |
| --- |
|  |

GridViewのIDプロパティを“GVTodo”にして、Widthプロパティを１００％にして下さい。

次に、デザインビューで、GVTodoの右側に、SqlDataSourceコントロールをドラッグアンドドロップして下さい。

配置したSqlDataSource1のIDプロパティを“SdsTodo”にして下さい。

次に、SdsTodoの右肩の＞印をクリックして、タスクメニューから、“データソースの構成”をクリックします。

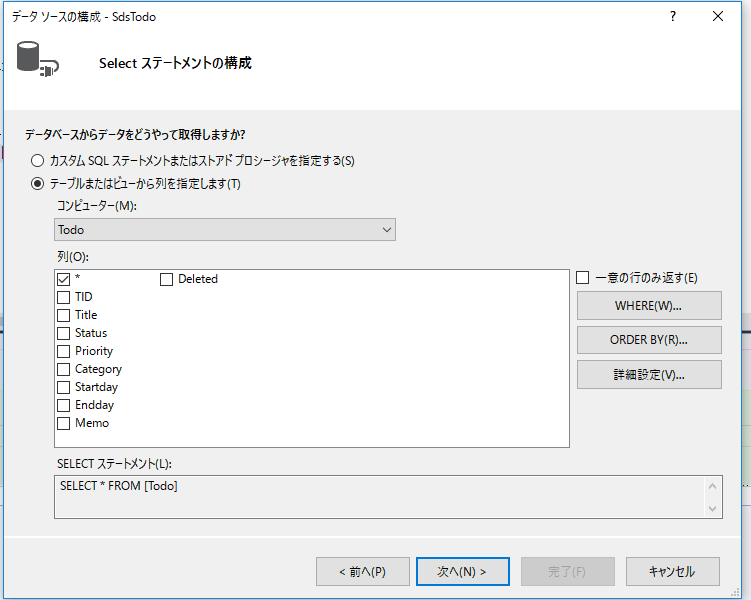
## データソースの構成

下図の「新しい接続」の左側の、ドロップダウンを開いて、“SiteSqlServer”を選択して、

|  |
| --- |
|  |

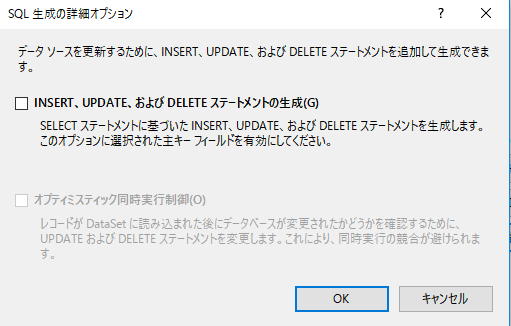
「次へ」ボタンをクリックします。

“テーブルまたはビューから列を指定します”のチェックを入れて、下のドロップダウンを開いて、Todoテーブルを選択します。次図



上図の右側の「詳細設定」ボタンをクリックします。

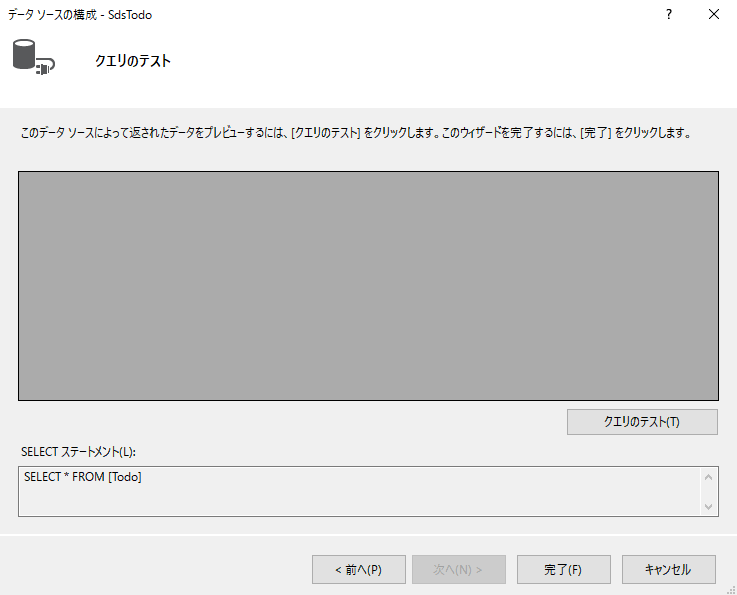
下図が表示されます。



上図の“INSERT,UPDATE,及び………の行のチェックボックスにチェックを入れて、「OK」ボタンをクリックします。

Selectステートメントの構成ダイアログに戻ります、「次へ」ボタンをクリックします。

下図が表示されます。

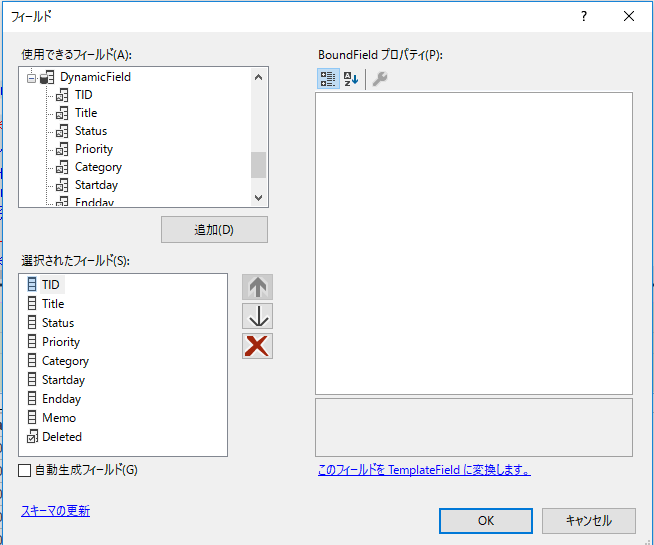


「完了」ボタンをクリックします。

### データコントロール（GridView）とデータソースを関連つける

GridTodoを選択状態にして、右肩のタスクボタンをクリックします。

表示された、メニューの「データソースの選択」のドロップダウンを開いて、「SdsTodo」を選択します。続けてGVTodoの右側にGridViewタスクが表示されます、中央付近の「列の編集」を選択します。フィールドを編集するための下図が表示されます



上図の「選択されたフィールド」の欄の“TID”を選択します。選択と同時に下図のように、

TIDのプロパティ設定用のウインドウが表示されます。

|  |
| --- |
|  |

プロパティ名“Visible”の右側の値欄（現在：Trueになっている）を”False”に設定して下さい。

次に「「選択されたフィールド」の欄の“Title”を選択して、”HeaderText”プロパティを下図のように、“タイトル”と入力して下さい。

同じようにして、各フィールドのプロパティを下表のように設定して下さい。

GVTodoのフィールドのプロパティ表

|  |  |  |
| --- | --- | --- |
| フィールド名 | プロパティ名 | 値 |
| TID | Visible | False |
| Title | HeaderText | タイトル |
| Status | HeaderText | 状態 |
| Priority | HeaderText | 優先度 |
| Category | HeaderText | 区別 |
| StartDay | HeaderText | 開始予定日 |
| EndDay | HeaderText | 終了予定日 |
| Memo | HeaderText | メモ |
| Deleted | Visible | False |

上表のように設定したら、「OK」ボタンをクリクして、終了します。

GVTodoの“TID”と”Deleted”のフィールドが表示されていないことを確認して下さい。下図

|  |
| --- |
|  |

ここで、表示のテストをするために、サーバーエクスプローラを表示して、Todoテーブルを右クリックして、「テーブルデータの表示」をします。（テーブルが表示されていない場合は、「テーブル」フォルダを右クリック→「最新の情報に更新」を選択して下さい。）

下図のように、手入力でデータを入力して下さい。

|  |
| --- |
| 2018/12/04  2018/12/10 |

上図の状態で２行目のどこかをクリックして下さい。TIDの列に自動で番号“１”が挿入されます。

前図のテーブルデータの表示を閉じてください。

次に、GVTodoのWidthプロパティを１００％にして下さい。

ここで、ブラウザーで表示、実行して下さい。次図のようにGVTodoにデータが表示されたでしょうか。

|  |
| --- |
|  |

上図の表示の中で、開始予定日、及び、終了予定日のデータの表示を日本式の日付表示で表示させたいと思います。

実行画面を閉じます。

### GridViewの日付表示を日本式の表示にする

Todo.aspxページの、GVTodoを選択状態にして、タスクメニューから、「列の編集」をします。「選択されたフィールド」の「開始予定日」を選択します、プロパティ設定欄の下図のように、「DataFormaString」プロパティを「{0:D}」と入力して下さい。

|  |
| --- |
|  |

「終了予定日」も「DataFormatString」プロパティも同じようにして下さい。

「OK」ボタンをクリックして「列の編集」を終了します。

ブラウザーで表示、実行して日本式日付表示になっていることを確かめて下さい。

実行画面を閉じます。

次は、データベースを理解してから、プログラムを続行します。

データベースを操作するために,特化したプログラム言語をSQL（StructuredQueryLanguege）といいます。C言語とか、VB言語等とは、違います。

以下に説明する、“SQLステートメント”とは、“SQL言語を使用した命令文”と認識して下さい。したがって、そのような意味から、“SELECT…..” “INSERT….”UPDATE…”

“DELETE….” から始まる文を“SELECT文”とか“SELECTコマンド”等と言います。

## テーブル内のデータの操作

### 行の参照：Selectステートメント

テーブルに登録された行（Row）の参照や検索はSELECTステートメントを使用します。

書式

|  |
| --- |
| SELECT　列名　FROM　テーブル名 |

行の参照　例１ 次の例はTodoテーブルから全ての行を表示します。

|  |
| --- |
| SELECT \* FROM　Todo |

行の参照　例２　特定の列を表示するには、列名を明示します。

|  |
| --- |
| SELECT　Title、Status、Priority　FROM　Todo |

行の参照　例３　行を限定して表示するには、WHERE句を使用して検索条件を指定します。

|  |
| --- |
| SELECT　Title、Status、Priority　FROM　Todo　WHERE　TID　＝　２ |

並び替えを指定したい場合は、“ORDERBY”句を指定します。

|  |
| --- |
| SELECT　Title、Status、Priority　FROM　Todo　ORDERBY　TID　DESC（またはASC） |

DESCは“降順”ASCは“昇順”に並べ替えます。

### 行の挿入：INSERTステートメント

テーブルにデータを挿入するにはINSERTステートメントを使用します。

|  |
| --- |
| INSERT　INTO　テーブル名 ([列名１],[列名２],….) VALUES (列名１の値、列名２の値…..) |

SQLステートメントでは文字列のデータは「,」（シングルクｵート）で囲みます。

例：テーブル名　Products　というテーブルがあるとします。 ()内はデータ型

|  |  |  |  |
| --- | --- | --- | --- |
| ID（int） | ProductName(nchar) | CategoryID(int) | Price(double) |
|  |  |  |  |
|  |  |  |  |

上の表のようなテーブルにデータを挿入する場合のINSERTステートメントは下記のようになります。

行の挿入　例１

|  |
| --- |
| INSERT INTO Products (ID, ProductsName, CategoryID, Price) VALUES(100, ’リンゴ’, 3, 1200) |

　　　　　　　　　　100 リンゴ　 3 1200

### 行の更新：UPDATEステートメント

テーブルに挿入済のレコードの更新はUPDATEステートメントを使用します。

UPDATEステートメント

|  |
| --- |
| UPDATE　テーブル名 SET 列名 = 値　WHERE(検索条件) |

次の例は、Productsテーブルの全ての行のPrice列５％値上げします。

|  |
| --- |
| UPDATE Products SET Price = Price \* 1.05) |

次の例はProductsteテーブルのProductIDが１０１の、ProductsName列の値を変更します。

|  |
| --- |
| UPDATE Products SET ProductsName = ‘サンプル商品’ WHERE ID = 101 |

### 行の削除：DELETEステートメント

行の削除はDELETEステートメントを使用します。

DELETEステートメント

|  |
| --- |
| DELETE FROM テーブル名　（WHERE　検索条件） |

次の例は、Productsテーブルから全てを削除するものです。

|  |
| --- |
| DELETE FROM Products |

次の例は特定の行を削除するためにWHERE句で指定しています。この例はPrice列の値が2000以下の行を削除しています。

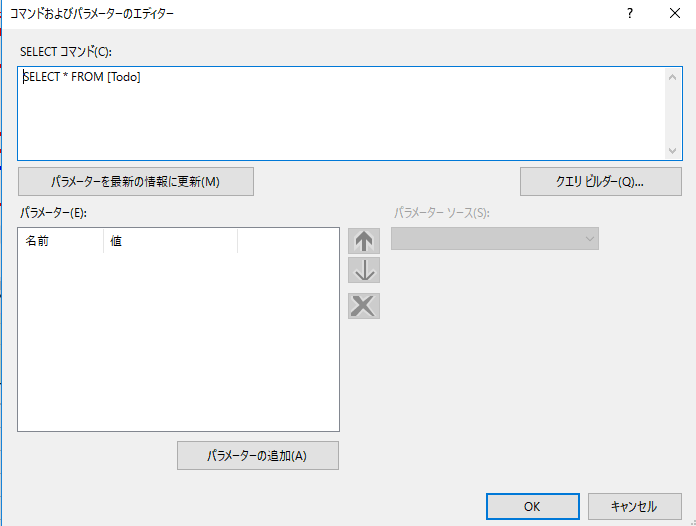
|  |
| --- |
| DELETE FROM Products WHERE Price <= 2000 |

## データソース、SqlDataSourceコントロールの中身

デザインビューのSdsTodoを選択して、プロパティウインドウを表示して、既にデータソースの構成済の自動で作成された各プロパティを見てみましょう。

SdsTodoのSelectQueryプロパティです。

### Selectコマンド（Select文）

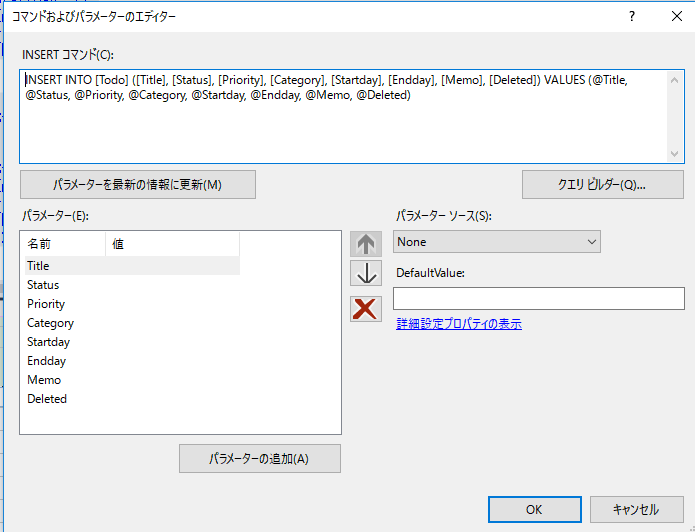


Select」文です。

Select \* From[Todo]は

Todoテーブルの全ての列を選択します。

### Insertコマンド

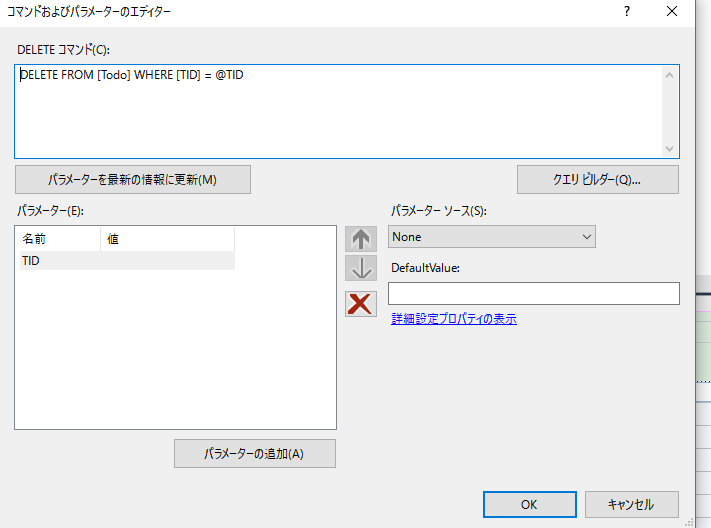


＠Titleは

パラメーター「Title」の値が代入される

Insert文です

### Deleteコマンド

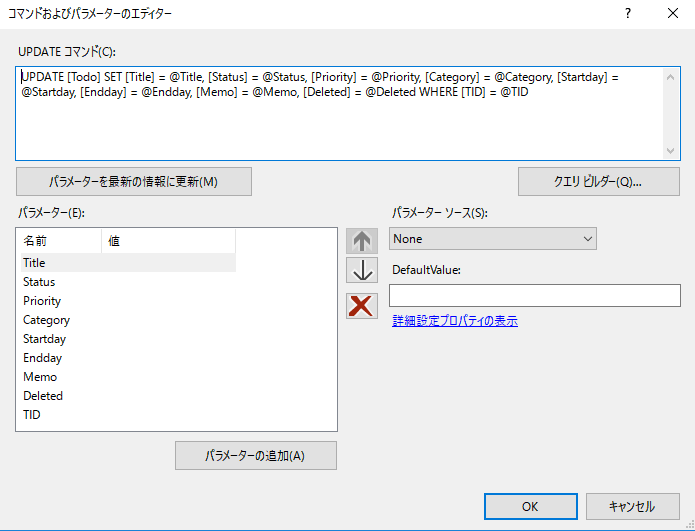


＠TIDの値によって、その行を削除します。

パラメータTIDの値

Delete文です

### Updateコマンド



WHERE TID = ＠TIDによって特定された行の列[Status]のデータは、パラメータStatusの値によって＠Statusに代入されて、変更される。

Update文です

この項での「＠パラメータ名」の値は、プログラムコードから設定する場合と、前図の

「SelectQuery」、「InsertQuery」、「DeleteQuery」、「UpdateQuery」それぞれの

「コマンド及びパラメータのエディター」ダイアログで、設定出来ます。

このように解説書として、文字で説明すると難しく感ずると思いますが実際にこのチュートリアルで実践（複数回）すれば簡単に理解出来ます。

プログラムコードに戻って、「登録」ボタンのイベントハンドラーをプログラムします。

現在スケジュール一覧表と、その元となるデータソースの「SdsTodo」の各プロパティが設定されています。一覧表となる、GridViewコントロールについては、基本的なプロパティ設定は出来ていますが、まだデザイン面でのプロパティ設定の余地はあります。

## 「登録」ボタンのイベントの処理（データを登録）

現在までのBtnRegi\_Clickイベントハンドラー

|  |
| --- |
| Protected Sub BtnRegi\_Click(sender As Object, e As EventArgs) Handles BtnRegi.Click  'DDLStatusの有効な値を検証します  If Me.DDLStatus.SelectedValue = "選択して下さい" Then  Me.LblOK.Visible = True  Exit Sub 'ユーザーがドロップダウンリストから選択ミスの場合  'ここでこのSubプロシージャを終了して,プログラムを中止  ElseIf Me.DDLStatus.SelectedValue <> "選択して下さい" Then  Me.LblOK.Visible = False  End If 'ここまででDDLStatusの検証を終了  If Page.IsValid Then '入力ミスがない場合PageはIsValid=Trueを返す  Me.LblOK.Visible = True　　'LblOKを表示して  Me.LblOK.Text = "Page.IsValid = " & Page.IsValid.ToString()  'Page.IsValidの値を確認するために、LblOKに表示する  End If  End Sub |

上図の赤色のコードを削除します。

上図の“登録ミスがない場合” をプログラムします。下図のようなプログラム手順になります。（赤色の文）

|  |
| --- |
| End If  If Page.IsValid Then '入力ミスがない場合  'ここでテーブルTodoにデータを登録する  End If  End Sub |

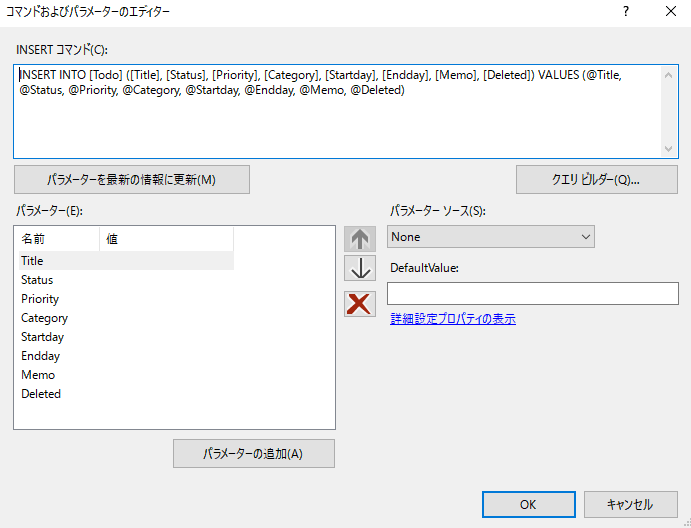
上記の“ここでテーブルTodoにデータを登録する”という作業は“INSERTコマンド”で行います。Todoテーブルは下図のような構造です。

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | TID | Title | Status | Priority | Category | StartDay | EndDay | Memo | Deleted |
| 型 | int | nvarchar | nvarchar | nvarchar | nvarchar | date | date | nvarchar | bit |
|  |  |  |  |  |  |  |  |  |  |

上図の全ての列にデータを登録します。

### 「コマンド及びパラメータのエディター」で編集

「ToDo.aspx」ページ内のSdsTodoのプロパティウインドウを開いて、“InsertQuery”プロパティの右の列（欄）をクリックして、詳細ボタンを表示させて、下図の「コマンド及びパラメータのエディター」を表示します。



上図の赤い四角の「パラメータ」欄に注目して下さい。パラメータの「名前」の右に「値」の欄があります。この値が空白（データベース用語では、“NULL”と表記します）になっていると、エラーになる場合があります、パラメータに値を代入する方法は、上図のパラメーターソース欄から設定する方法とプログラムコードから値を代入する方法があります。

パラメーターの値がNULL（何もない）を受け付けない場合、コンピューターはエラーを表示して、プログラムが止まってしまいます。この状況になることを次図で、説明します。

データテーブル作成時の設定

|  |
| --- |
| NULLを許可しないという記述  自動で＋１づつ登録する  チェックが入っていない列は  データの空白は許されない。  データの空白（NULL）OK |

上図のように、各列に、空白は許されない列のある、テーブルの設定になっています。

Memo列は、NULL（空白）が許可されています。

TID列は、自動でデータを入れるようになっているので、「コマンド及びパラメータのエディター」の“パラメータ―欄”の一覧にはありません。

### 「コマンド及びパラメータのエディター」で値を設定

下図の「パラメーター」欄のパラメータ名の上から順にパラメーターの値を設定していきます。

|  |
| --- |
|  |

1. 「名前」の“Title”をクリックします。
2. 「パラメータ―ソース」のドロップダウンを開いて“Control”を選択します。
3. 「ControlD」のドロップダウンを開いて、「ToDo.aspx」ページ内にあるコントロールの一覧が表示されます。“TxtTitle”を選択します。

上図のような表示の結果になります。上記の①から③までの操作で、「Titleの値はTxTitleのTextの値である」に設定しました。

続いて、Status～Memoまで、同じように設定して下さい。

|  |
| --- |
| この体験で、各コントロールのIDプロパティの名前付けの重要さを理解して下さい。 |

パラメータの設定完了、下図

|  |
| --- |
|  |

上図の「パラメーター」欄の最後の“Deleted”の列を設けた理由は、後で説明しますが、このデータ型は、Bit型にしてあります。この型は、０（False）か、１（True）どちらかのデータを格納する列です。つまり０または１の違いで判断する時にこの型を使用します。

この”Deleted”のデータはユーザーの入力欄にはありません、プログラマーが使用するためのデータです。今回のDeletedのパラメータ設定は、「コマンド及びパラメータのエディター」で設定します。

下図のようにDeletedを選択して、「DefaultValue」に入力して下さい。

|  |
| --- |
|  |

これで、全てのパラメーターの値が設定されました。

「コマンド及びパラメータのエディター」の「OK」ボタンをクリックして終了します。

## 「登録」ボタン、クリックイベントに、データ登録のコードを追加

ここまでで、データベースのテーブル「Todo」に新規のデータを挿入する準備ができました、実際にテーブルにデータを挿入するにはプログラムコードで指示しなければなりません。「登録」ボタンのClickイベントに下記のようにコードを追加します。赤色のコード、コピペ可能です。

|  |
| --- |
| End If  If Page.IsValid Then '入力ミスがない場合  'ここでテーブルTodoにデータを登録する  　　　　　　SdsTodo.Insert()  LblOK.Text = “登録しました”  LblOK.Visible = True  　　　　End If  End Sub |

早速、ブラウザーで表示、実行して下さい。

実行結果が次図のように表示されたら、大成功です。

|  |
| --- |
|  |

実行画面を閉じます。

次は、このアプリケーションが最初に表示された時にユーザーは新規の予定を作成するのか、既に作成された予定の一覧表だけを参照するのか、解りません。

その様な問題を解決するための手段として、ユーザーの目的に応ずるためのボタンを表示します。

## 簡単なメニュー機能を作成

「ToDo.aspx」のコードビューの一番上の、下記の部分に注目します。目印の“ｋｋ”という文字を入力

|  |
| --- |
|  |

<h3>…</h3>の直下に3行4列のテーブルを挿入します、次に、各列の幅を２５％にしてください、次にテーブルの2行目の2列目、と3列目に“Button”コントロールを配置して下さい、配置したButtonのIDはそのままにします。下表のようにプロパティを設定します。

|  |  |  |
| --- | --- | --- |
| コントロールID名 | プロパティ名 | 値 |
| Button1 | CousesValidation | False（クリック時に検証コントロールに信号を送らない） |
| Text | 新規予定作成:表示、非表示 |
| Button2 | CousesValidation | False（クリック時に検証コントロールに信号を送らない） |
| Text | 一覧表：表示、非表示 |

目印“ｋｋ”を削除します。

次に、コードビューのそれぞれのタグの位置に下記のコードを挿入して下さい。

## Bootstrap-のボタンのデザインを利用する

赤色の選択部分のコード（Bootstrapのclass btn定義を利用）赤色のコード、コピペ可能です。

|  |
| --- |
| <td style="width: 25%">  <asp:Button ID="Button1" runat="server" Text="新規予定作成:表示、非表示" class="btn btn-primary" CausesValidation="False"/>  </td>  <td style="width: 25%">  <asp:Button ID="Button2" runat="server" Text="一覧表：表示、非表示" class="btn btn-primary" CausesValidation="False"/> |

デザインビューは下図のようになっています。

|  |
| --- |
|  |

次に、タイトル入力部分のhtmlコードを下図のように、Panelコントロールで包みます。

手順１、Panelコントロールの配置

|  |  |
| --- | --- |
|  | この部分に、Panelコントロールをドラッグアンドドロップして下さい。 |

手順２、Panelコントロールの終わり要素</asp:Panel>を選択

|  |
| --- |
|  |

手順３、選択して要素を「切り取る」下図のように要素を選択して、右クリックしてメニューを表示します。「切り取り」を選択します。

|  |
| --- |
|  |

手順４、切り取った要素を目的の位置に、貼り付ける

|  |
| --- |
| この位置に、「貼り付け」 |

手順５、PanelコントロールのIDを手入力で設定、下表のように、IDを“PanelTitle”に設定して下さい。

|  |  |
| --- | --- |
|  | IDを“PanelTitole”に変更した。  いづれの手順の行番号は、開発環境によって違いますので、参考にして下さい。 |

次に、PanelTitleのVisibleプロパティををFalseにして下さい。

次に、先程の「ToDo.aspx」ページに追加した、Button1、Button2のイベントハンドラーを作成して下さい。

作成されたButton1、Button2のイベントハンドラーに下記のようにコードを追加して下さい。

赤色のコードを追加

|  |
| --- |
| Protected Sub Button1\_Click(sender As Object, e As EventArgs) Handles Button1.Click  If Me.PanelDate.Visible = False Then  Me.PanelIchiran.Visible = False  　　　　　　Me.PanelTitle.Visible = True  Else  Me.PanelDate.Visible = False  End If  End Sub  Protected Sub Button2\_Click(sender As Object, e As EventArgs) Handles Button2.Click  If Me.PanelIchiran.Visible = False Then  Me.PanelIchiran.Visible = True  Me.PanelDate.Visible = False  Me.PanelTitle.Visible = False  Else  Me.PanelIchiran.Visible = False  End If  End Sub  End Class |

ブラウザーで表示、実行します。

アプリケーションらしくなりました。

実行画面を閉じます。

次は、新規にスケジュールのデータをデータべーステーブルに登録直後に一覧表を表示するプログラムをします。

## イベントハンドラーの作成方法

## データベーステーブルにデータ挿入時のイベント

データベーステーブルにデータ挿入時のイベントは、SdsTodoの“Inserted”イベントが発生します。プログラムコードファイルの一番上部の３列に区切ってある、ドロップダウンを含んだ列の中央の列のドロップダウンを開いてください。下図

|  |
| --- |
|  |

次に、表示された、ドロップダウンのリストの下部にある「sdsTodo」を選択します。下図のように、中央の列と、その右の列の表示が変わります。

|  |
| --- |
|  |

次に、右の列の“DataBinding”と表示された、ドロップダウンを開き、”Inserted”という項目をクリックします、下図

|  |
| --- |
|  |

上図の“Inserted”をクリックすると、プログラムコードファイルの一番末尾に下図のような「sdsTodo\_Inserted」イベントハンドラーが作成されます。

|  |
| --- |
|  |

前図のSdsTod\_Insertedイベントコード内に下記のコードを追加して下さい。赤のコード、コピペ可能です。

|  |
| --- |
| Private Sub sdsTodo\_Inserted(sender As Object, e As SqlDataSourceStatusEventArgs) Handles sdsTodo.Inserted  Me.PanelIchiran.Visible = True  End Sub |

「登録」ボタンをクリックして、新規にデータが登録されて、一覧表を表示するように、プログラムしました。

本実践解説のこのアプリでは、このコードを追加しなくても“PanelIchiran”は表示されますが、他のアプリでは、このイベントを利用して、データの登録直後にプログラムすべき事柄はこの“Inserted”イベントハンドラーを利用して下さい。

そして、プログラムコードファイル内に、各コントロールのイベントハンドラーの作成の方法を解説しました。

## 登録したデータを変更する

## GridViewのデータを編集する

スケジュールが進行した場合“状態”が変化します、“未着手”のスケジュールが“開始”されたり、“開始”が“終了”になったりします。その様な場合、登録したデータを変更しなければなりません。ユーザーが簡単に変更できるように、これからプログラムします。

スケジュール一表の“GVTodo”にその機能を付加します。

現在の“GridView1”は下図のようになっています。デザインビュー

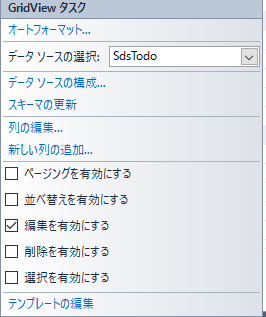
|  |
| --- |
|  |

GridViewの編集機能について、

1. 編集したい列をTemplateFieldにする
2. テンプレート編集時には、各列のEditItemTemplateで編集できる

|  |
| --- |
| 技術情報：“[GridView サンプル](https://msdn.microsoft.com/ja-jp/library/dd297767.aspx)”、“[GridViewの概要](https://docs.microsoft.com/ja-jp/dotnet/framework/wpf/controls/gridview-overview)”  “[編集インターフェイスのカスタマイズ](https://docs.microsoft.com/ja-jp/aspnet/web-forms/overview/data-access/custom-formatting/using-templatefields-in-the-gridview-control-vb)”の記事は重要です |

上図のGVTodoを選択状態にして、右肩のタスクボタンをクリックして、「GridViewタスク」メニューを表示し、チェックボックスのついたメニューの内「編集を有効にする」を選択して、✔を入れます。



そのまま表示されているタスクメニュー、または表示されていない場合再びタスクメニューを表示して、「列の編集」を選択して、表示された「フィールド」のダイアログの外側（下枠）をマウスの操作で、「選択されたフィールド」の全ての列名が表示されるように、大きくして下さい。次図

|  |
| --- |
|  |

上図の一番上の「編集、更新、キャンセル」という列名を選択状態にして、その欄の右側の下向き↓をクリックしながら一番下まで、移動します。次図

|  |
| --- |
| “変更”にする |

次に、右のプロパティ欄の「EditText」プロパティを“変更”にします。

次に、上図の「OK」ボタンの直上の青い文字の「このフィールドをTemplateFieldに変換します」という文章をクリックします。すると「選択されたフィールド」の先程の「編集、更新、キャンセル」という列名が下図のように、TemplateFieldという表示に変わります。

|  |
| --- |
|  |

同様にして、「選択されたフィールド」の“タイトル”から“メモ”までの列をTemplateField（編集が可能になるように）にして下さい。下図

|  |
| --- |
|  |

「OK」ボタンをクリックして、「列の編集」を終わります。

デザインビューに戻ると、**GridView1**の右端に下図のように“変更”という文字のボタンが表示されています。下図

|  |
| --- |
|  |

### デザインビューで、各テンプレートを編集する

#### テンプレートを編集画面を表示する方法１

GVTodoのタスクメニューを表示して、一番下の「テンプレートの編集」をクリックします。下図が表示されていると思います。または画面幅いっぱいに表示されている場合もあります。

|  |
| --- |
|  |

この状態で、上図の右肩のタスクメニューを表示します。表示された「表示名」は“ItemTemplate”になっています。更に「表示名」のドロップダウンを開いて、「EditItemTemplate」を選択します。下図の「Column[1]-タイトル」の”EditItemTemplate”が表示されます。

|  |
| --- |
|  |

#### テンプレートの編集画面を表示する方法２

GVTodoのタスクメニューを表示して、一番下の「テンプレートの編集」をクリックします。下図が表示されていると思います。または画面幅いっぱいに表示されている場合もあります。

|  |
| --- |
|  |

この状態で、上図の右肩のタスクメニューを表示します。表示された「表示名」は“ItemTemplate”になっています。更に「表示名」のドロップダウンを開いて、

「Column[2]-状態という文字列をクリックします。下図が表示されます。

|  |
| --- |
| テンプレートフィールド |

上図のように、「状態列」の全てのテンプレートが表示されて、各テンプレートの中が編集出来ます。このような特定の列のテンプレートの集合体を“テンプレートフィールド”

（TemplateField）と呼びます。

以上のように、各列のテンプレートを編集するための編集画面の表示の仕方は、２種類ありますので、読者様の好みに応じて、使い分けして下さい。

例えば、「開始予定日」のテンプレートフィールドを見てみます。

|  |
| --- |
|  |

上図のように、GridView の既定の編集機能では、編集インターフェイスが最適とは言えません。既定では、GridView 行のすべての編集可能な列が TextBox で表示されます。

しかし列によってはTextBoxでは、変更するデータがTextBoxでは妥当ではありません。

### 編集インターフェイスでの代替 Web コントロールの使用

たとえば、日付フィールドを編集する場合は、TextBox に日付をタイプするよりも、「Calendar Web コントロール」を使用して日付を選択する方が簡単に編集、変更することができます。

同様に、適切な値が制限され、別のデータベース テーブルにそれが定義されているような参照データの場合は、通常、DropDownList、またはListBox が最適な編集インターフェースとなります。

さあ！！実践しましょう。

GVTodoのテンプレートの編集を終了して下さい。

現在、データベースの「ToDo」テーブルにデータが正しく挿入されていな場合は、次の操作でデータを削除して下さい。

データが正しい場合は、削除しなくて、次へ進んで下さい。

### データベーステーブルのデータ行を削除する

サーバーエクスプローラを表示して、現在のデータベーステーブル「Todo」のデータを表示して下さい。

|  |
| --- |
|  |

上図のテーブル「Todo」の一番左のカーソルを選択して、右クリックメニューから、データを全て削除します。多数の行がある場合は、「Shift」キーを利用して、全行を選択し、右クリックメニューから削除します。テーブルデータの表示を終了して下さい。

### GridViewの編集機能をテスト表示する

次に、ブラウザーで表示、実行して、新規にデータを登録して下さい。登録後に表示されたGridView（GVTodo）の「変更」ボタンをクリックして見て下さい。当該行が編集可能な状態になったことを確認して下さい。下図

|  |
| --- |
|  |

上図の編集用のコントロールが全てTextBoxコントロールです、編集のために、マウスでTextBox内をクリックして見て下さい。確認できます。そして、「開始予定日」「終了予定日」の編集、更新は、TextBoxでは、出来ないことはありませんが、不便です。

右端のキャンセルボタンをクリックして、終了して下さい。

実行画面を閉じます。

次に、この項のタイトル「編集インターフェイスでの代替 Web コントロールの使用」を実践します。

### 日付のデータ列の編集インターフェースをカレンダーコントロールにする

下図のように「開始予定日」の列の“EditItemTemplate”を表示して下さい。

前項の「テンプレートの編集画面を表示する方法１」で表示してあります。

|  |
| --- |
|  |

上図TextBoxを選択して、タスクメニューを表示します、次図

|  |
| --- |
|  |

「DataBindingの編集」をクリックします。下図が表示されます。

|  |
| --- |
|  |

上図、赤楕円で囲んだ部分は、「Bind(“StartDay”)」というコードが自動で挿入されています。この意味は、「GVTodoのデータソースSdsTodoにおいて、テーブル“Todo”内の”StartDay"列のテータを参照して、かつ変更も可能であることを意味しています。

参照だけの場合は「Eva(“StartDay”)」というコードです。

また、SdsTodoのUpdateQueryプロパティのUPDATEコマンドの「パラメータ」内の“StartDay”の値と直結しています。下図

|  |
| --- |
|  |

実践によって、「Bind(“StartDay”)」の機能を確認していきます。

先程のデザインビューの開始予定日のEditItemTemplateに戻って、自動で挿入された“TextBox５”を選択して、右クリックメニューの”切り取り“を選択して、削除して下さい。次に、削除して位置にツールボックスのCalendarコントロールを配置して下さい。

同じようにして、終了予定日のEditItemTemplateを表示して、TextBox６を削除して、Calendarコントロールを配置して下さい。

そのまま、終了予定日のEditItemTemplateのカレンダーコントロールを選択して、右肩のタスクメニューの「DataBindingの編集」を選択次に表示されるダイアログの下図のように「Bind(“EndDay”)と入力します。

|  |
| --- |
|  |

「OK」ボタンをクリックします。

開始予定日のEditItemTemplateのCalendarコントロールも同じように設定してください。

開始予定日の場合のDataBindingは、「Bind(“Startday”)」です。

次に、SqlDataSourceコントロールのSdsTodoのプロパティウインドウを表示して、UpdateQueryプロパティのUPDATEコマンドの「パラメータ」を展開して、下図の「Deleted」の部分を選択して、パラメータソースの「DefaultValue」の空欄に「False」と入力します。次図

|  |
| --- |
|  |

次に、パラメータ“TID”を選択して、下図のように、パラメータソースのドロップダウンの中から“Control”を選択、次に”ControlID”のドロップダウンから、GVTodoを選択します、赤の矢印のように値が自動で設定されます。

|  |
| --- |
|  |

「OK」ボタンをクリックして、終了します。

前図の“GVTodo.SelectedValue”のコードは、GridViewコントロールのID「GVTodo」のDatakeyNameプロパティの”値“です。GVTodoのテンプレートの編集を終了して、デザインビューで、GVTodoを選択して、DatakeyNameプロパティを確認して下さい。”TID”に（自動で）なっている筈です。

手入力で、変更も出来ます。筆者の上級レベルのチュートリアルでは、もっと詳しく解説しています。

|  |  |
| --- | --- |
| GVTodoがDatakeyの値を返す状態   |  | | --- | |  |   上図のGridViewの変更ボタンをクリックして行を選択状態にした場合、に当該GridViewのDatakeyNameプロパティが“TID”であると、上図の一番左のの”TID”列の値が返されます。その値をプログラムコードでデータとして受け取るときは、“変数myTid = GVTodo.SelectedValue”というコードで、TIDの値を取得します。  上図は、GVTodoの列の編集でTID列をVisible＝Trueにした実行画面の表示です。  「TID」列をVisible＝Falseにしても、TIDの値は取得できます。 |

ここで、ブラウザーで表示、実行して、「一覧表：表示、非表示」ボタンをクリック、登録済のデータを表示して、変更ボタンをクリックして、（両カレンダーには、登録された日付がグレー色の背景色で表示されています）開始日と終了日をカレンダーから変更して、“更新”ボタンをクリックして下さい。瞬時に変更が反映されて、GVTodoに変更された日付が表示されます。

**すげえ！！目からウロコ！**

実行画面を閉じます。

しかし、これでは、ユーザビリティ（デザイン）が悪すぎます。ユーザビリティを改良しましょう。

開始予定日のEditoItemTemplateを表示して、カレンダーを選択します。CalendarのSlectedDayStyleプロパティをドロップダウンして、BackColor、を青に、BorderStyleをSolidに、BorderWidthを１ｐｘにして下さい。下図

|  |
| --- |
|  |

終了予定日の列、EditoItemTemplateのカレンダーも同じプロパティを設定して、BackColor を赤にして下さい。

テンプレートの編集を終了します。

次に、もう一つ、変更ボタンをクリックした時にその行の枠線を赤色にして、変更すべき行が一目でわかるようにします。デザインビューのGVTodoを選択して、GVTodoのプロパティウインドウのEditRowStyleプロパティ各項目を下記のように設定します

|  |
| --- |
|  |

ここで、ブラウザーで表示、実行して下さい。一覧表を表示して、「変更」ボタンをクリックして、カレンダーの選択日の色、編集行の枠線を確認して下さい。下図は登録行数が１行の状態ですが、行数が多くなるとこの選択行の枠線でユーザビリティが良くなります。

|  |
| --- |
|  |

実行画面をとじます。

### 状態の列編集インターフェースをドロップダウンリストにする

状態の列のEditItemTemplateの、TextBox2のDataBindingは「Bind(“Status”)」になっていることを確かめて、TextBox２を削除して、ツールボックスからDropDownListコントロールを配置して下さい。IDを“DDLStatus”にします、タスクメニューの“データソースの選択”を選択して、下図のデータソース構成ウィザードのように設定して下さい。

|  |
| --- |
| データフィールド名が表示されない場合ここをクリックして下さい |

「OK」ボタンをクリックして、終了して下さい。

次に、DDLStatusを選択して、右肩のタスクメニューの「DataBindingの編集」を選択して、下図のように設定して下さい。

|  |
| --- |
| Bind(“Status”) |

「OK」ボタンをクリックして、終了します。

### 優先度列の編集インターフェースをRatioButtonListにする

状態の列のEditItemTemplateの、TextBox3のDataBindingは「Bind(“Priority”)」になっていることを確かめて、TextBox3を削除して、ツールボックスからRadioButtonListコントロールを配置して下さい。IDを“RBLPriority”にします、RBLPriorityのプロパティウインドウから、下図の“Items”プロパティの（コレクション）をクリックして、下図のように設定します。それぞれのメンバーのTextとValueは同じにして下さい。

|  |
| --- |
|  |

「OK」ボタンをクリックして、終了して下さい。

次に、RepeatDirectionプロパティを“Horizontal”にして下さい。

RBLPriorityのDataBindingを「Bind(“Priority”)にして下さい。

### 区別列の編集インターフェースをListBoxにする

区別の列のEditItemTemplateの、TextBox4のDataBindingは「Bind(“Category”)」になっていることを確かめて、TextBox4を削除して、ツールボックスからListBoxコントロールを配置して下さい。IDを“CategoryList”にして、Widthプロパティを120pxにします、タスクメニューの“データソースの選択”を選択して、下図のデータソース構成ウィザードのように設定して下さい。

|  |
| --- |
|  |

DataBindingを「Bind(“Category”)」にして下さい。

「OK」ボタンをクリックして、終了して下さい。

### メモ列の編集インターフェース「TextBox」プロパティの変更

メモの列のEditItemTemplateの、TextBox7のプロパティウインドウで下図のように設定して下さい。

|  |
| --- |
|  |

以上で、テンプレートの編集を終了して下さい。

ブラウザーで表示、一覧表を表示して、実行して下さい。

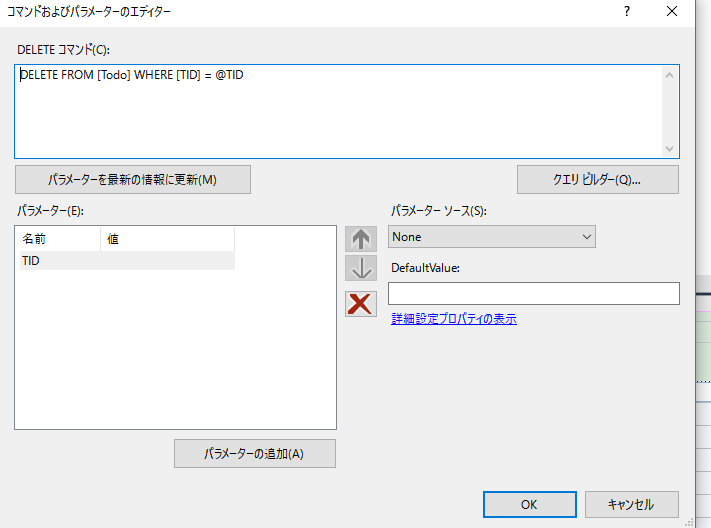
まず、データを登録して、表示された一覧のどこかの行を変更して見て下さい。

実行画面を閉じます。

### 登録したデータを削除する。

ここまで、データテーブルにデータを登録する手順、そして登録されたデータを変更、更新する手順、を実践してきました。もう一つ重要な事として不要なデータをデータベースから削除する手順を解説します。

sdsTodoコントロールのプロパティウインドウから、DeleteQueryプロパティの詳細ボタンをクリックして、下図を表示して下さい。



＠TIDの値によって、その行を削除します。

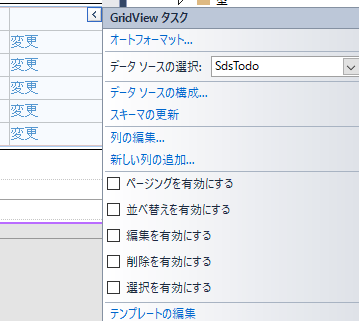
パラメータTIDの値

Deleteコマンドは非常に簡潔です。「DELETE FROM [Todo] WHERE [TID] = @TID」

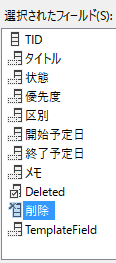
というコマンド文です。つまり、パラメータのTID列はテーブル“Todo”の主キーです、主キーの値は重複しませんので、パラメータTIDの値を”3“にした場合、テーブル内のTIDの列の値”3“の行が削除されます。

上図の「キャンセル」ボタンをクリックして、ダイアログを閉じて下さい。

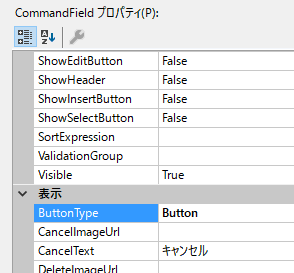
次に、GVTodo のタスクメニューを表示して、下図



「削除を有効にする」チェックボックスにチェックを入れます。続いて、「列の編集」をクリックして、「選択されたフィールドの」の一番上の「削除」というフィールドを下図のように下から2番目まで移動して下さい。下図



上図のように「削除」フィールドを選択した状態で、右の欄の「CommandFieldプロパティ」で、下図のように「表示」の“ButtonType”を“Link”から“Button”に変更します。



列の編集の「OK」ボタンをクリックして、終了します。

ブラウザーで表示、はしないで下さい。

#### GridViewから削除する時確認のメッセージを表示す

現在の状態で、実行して、任意の行の「削除」ボタンをクリックすると、即座にその行が削除されます、しかし、あまりにも削除作業が早すぎて、誤って削除ボタンを押したときには削除されてしまい、取り返しがきかない場合も生じますので、「削除してもいいですか？」という確認の表示がほしいものです。

GVTodoのタスクメニューを表示して、「列の編集」を選択、「削除」フィールドを選択して、右欄の下の、「このフィールドをTemplateフィールドに変換します」という、Linkボタンをクリックします。列の編集「OK」ボタンをクリックして終了します。

次に、GVTodoのタスクメニューから、「テンプレートの編集」をクリックして、削除フィールドの「Column[9]」の「ItemTemplate」を選択します。下図

|  |
| --- |
|  |

既に、「削除」ボタンがあります。そのボタンを選択状態にして、プロパティウインドウを表示します。次図のように「OnClientClick」プロパティに入力します。そして他の赤丸の部分を確認して下さい。

|  |
| --- |
| **確認** |

上図の「return confirm(“削除してもいいですか？”)」のダブルクオーテーションマーク「“」は半角文字であることに注意して下さい。

「テンプレートの編集」を終了して下さい。

ブラウザーで表示、実行して、適当にデータを挿入して、削除作業を行って下さい。

下図が表示されます。

|  |
| --- |
|  |

「キャンセル」ボタンをクリックします。

次に、もう一度GVTodoの「削除」ボタンをクリックして、「OK」ボタンをクリックします。プログラム通り、削除されました。　目から**ウロコ、ウロコ、ウロコ！！！**

実行画面を閉じます。

## スケジュール作成アプリのユーザビリティをアップする

ここまで、スケジュール作成の基本的な下記の機能をプログラムしてきました。

1. スケジュール作成のためのデータ入力を支援する、VSのWebコントロールを配置
2. ページを形成する、CSS、bootstrap-等の使用法
3. ユーザーの入力手間を省くデータ表示系のコントロールを使用
4. データベースによる、データの登録、変更、更新、削除、等の機能を装備

新規スケジュール作成の入力画面は下図のようになっています。

|  |
| --- |
|  |

そもそも、アプリケーションの作成開始時点では、ユーザビリティ（デザイン）というものは、完全な形でイメージ出来ません、作成途中か骨格が出来たときに、修正箇所に気づいたりするものです。プログラミングの作業中においても常に、ユーザーの使いやすさ等を考慮にいれて、プログラム作業をすべきです、ユーザビリティ（デザイン）の修正が必要だと思ったら、プログラム作業の途中でも修正することを心掛けてください。

後になってからでは、修正が困難になったり、非常に修正のための時間を費やしたりします。

下図のような修正案を考えます。

|  |
| --- |
|  |

これで「本格的なアプリ開発を経験」の実践解説を終わります。

本解説書はあくまでも初心者の方がVisualStudioを使用してプログラムの世界に足を踏み入れるための解説書であって、実用的なアプリを提供するような解説書ではありませんが、本解説書は、VisualStudioの使用法を習得し、複数回実践することでプログラムの基礎を身につけることが出来ます、またプログラマーとしての必須の知識であるデータベースの基本も身につけることが出来ます、そして次の「Webショップ構築（１）」を実践する上でこの解説書を何度も参照してこの解説書の内容を身につけて下さい、そして新しい技術の取得による達成感でわくわくして次のステップに上がって下さい。

次の「Webショップ構築(1)」での実践内容はWebショップ構築に不可欠なプログラムを丁寧に解説してあります。

解説シリーズのタイトルは“Webショップ構築”としてありますが本当の目標は、Webアプリケーションのプログラミングを習得するのが目標です。

コンピュータプログラムを習得するためには、イメージ画像処理、高度なデータベース処理、インターネットテクノロジー、等、結構必要不可欠な知識が広く求められます、また時間的にも長期の期間も必要です、そのためのモチベーションの維持も必要です、“Webショッププログラム構築”という目標を定めることで実践者がモチベーションを切らすことなく節目節目で、達成感を味わいながら目標の頂きに登ることが出来ます。そのモチベーションの維持こそ短期間でコンピュータプログラムの知識の習得が出来る原動力です、そして、山の頂きが高くあればある程、周囲が良く見えてきます、その場所に立ったら更にプロフェッショナルになるため道筋（最新のプログラミング手法、インターネットテクノロジー等を理解する力）がくっきりと見えてきます。

「Webショップ構築(1)、Webショップ構築(2)」はそのような、読者様の背中を押してくれる解説書です。

# 付録

**共通する設定**

以下の記事は、本実践解説において複数回、同じ手順を実行する場合があります、そのような場合この手順を参照して下さい。

## VisualStudioのテーブルとその配置と使用法

### VisualStudio のテーブルとは、下図の位置のメニューにある項目の事を指します。注：デザインビューでの作業中に表示されます。

|  |
| --- |
|  |

上図、「テーブル」メニューをクリックして、下図の手順でテーブルを編集します。

|  |  |  |  |
| --- | --- | --- | --- |
| **１** | **２** | **３** | **４** |
|  | 左図の黄色の「テーブルの挿入」をクリックします。  右図が表示されます。 |  | 次項に左図のテーブルの挿入ダイアログの具体設定例を解説しておきます。  以下、本チューとリアルでの解説の中でテーブルとは、このテーブルの事を言います。  データベースのテーブルの事は、データテーブルと呼びます。 |

**テーブルの挿入ダイアログ設定例とその他の設定例**

DataListコントロールのItemTemplate内にテーブルを配置して、各種のプロパティ設定例

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 下図はデフォルトのItemTemplateの状態    上図の“SCID[SCIDLabel]の行の右にマウスカーソルを置いて、テーブルを配置します。 | | 表示されたダイアログを下図のように設定します。  （必要な行数をあらかじめ決めておきます）後で行数は増やすことも、削除することも出来ます。 | | 「OK」ボタンをクリック後ItemTemplateは下図のようになります。    上の図の状態を④図のようにデザインします。 |
| ４ | 左図のImageコントロールの行は、行をマウスで右クリックして、  右図のように「セルの結合」を選択します。  セルの結合をしてからImageコントロールを配置します。又「編集」リンクボタンの部分もセルの結合をしてからLinkButtonコントロールを配置します。 | |  | |

**配置したテーブルの枠線の表示方法を解説**

|  |  |  |  |
| --- | --- | --- | --- |
|  | 左図のようにテーブルを配置した状態で、右図のように列全体を選択します（テーブルの外側をマウスでなぞると下向↓が出現します）。次にこの状態でプロパティウインドウを開き、下から６行目の「Style」という行をクリックします。その行の右端の詳細ボタンをクリックします。  下図のようなプロパティ設定ダイアログが表示され、「枠線」を選択します。   |  | | --- | |  |   次に、下図のように設定します、  手入力   |  | | --- | |  |   「OK」ボタンをクリックします。次図のような結果になります。 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | 左図のようにテーブルに枠線を表示するためには、列毎にここまでの手順を繰り返します、そして外側の枠線はテーブル全体を選択して、同じ要領で枠線を設定します。  結果右図 | |  | | --- | |  | |  |

**セルを結合してコントロールを配置する**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 下図のように、結合するセルを選択して、右クリックメニューすると下図のように表示されます。   |  | | --- | |  | | 左図の「セルの結合」をクリックすると下図のようになります。   |  | | --- | |  | | 左図の赤丸で囲んだ位置に、コントロール（Image）を配置します。 |  |

**テンプレートに配置したテーブルに既存のコントロールを移動する**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 右図のようにItemTemplate内にTextやLabelコントロールが配置してある場合を例に各コントロールの移動と新しいコントロールの配置を解説します。  このような状況は他のデータコントロールのTemplateにも共通する事ですので是非実践して経験をして下さい。 |  | 左図のテーブル内にコントロールを移動する場合で、表示すべきデータと表示しないデータをあらかじめ決めておくべきです。そして表示するデータの、位置、タイトルも事前に決めて置いてください。まず１行目にはImageコントロールを配置することにします。１行目にツールボックスからImageコントロールをドラッグアンドドロップして下さい、後は必要な既存のコントロールを移動します、５行目はセルの結合をして、そこにツールボックスからLinkButton を配置します。既存のコントロール（この場合はLabelコントロール）を移動する時はコントロールをドラッグして移動し、削除する時は、「BackSpace」キーと、「Delete」キーをうまく使用して行います。  各コントロールのIDの付与、及び変更もこの際に行った方がいいです。  右図が結果です。 | |  | | --- | |  |   上図のItemTemplate内のテーブルもデザインして、見栄えも良くすることが出来ます。次に項で解説します。 |

**テンプレート内のテーブルをデザインする**

**テーブル内のイメージ、テキスト等の表示を中央に表示する**

前項のItemTemplate内のテーブルは下表のようにhtmlコードで形成されています。

|  |
| --- |
|  |

上図の左端の行番号１４２の<table cellpadding=”0” ,,,,,,,,>の行の任意の位置にマウスカーソルを置き、VisualStudioのプロパティウインドウを開きます。次図

|  |
| --- |
|  |

前図のプロパティウインドウの赤で囲んだ「style」という項目をクリックして、詳細ボタンが表示されるので、クリックします。

下図のようなダイアログが表示されます。

|  |  |  |  |
| --- | --- | --- | --- |
|  | 左図のようにカテゴリ「ブロック」を選択して「Text-align」を“center”にします。   |  | | --- | |  |   このダイアログの右下の「OK」ボタンをクリックします。  結果を確認すると、Imageと各テキストが中央に表示されています。下図   |  | | --- | |  | |

**VisualStudioのテーブルをbootstrapのテーブルに変身させる**

是非このサイトの情報を確認して下さい。Bootstrap参考サイト：カテゴリーは「CSS」　 [bootstrap3日本語リファレンス](http://bootstrap3.cyberlab.info/)

上記サイトのテーブル（表）スタイルの解説を見ると下表のように用途によっては使えるスタイルがあります。是非活用して下さい。

|  |
| --- |
|  |

VisualStudio のテーブルをbootstrapのテーブルに変身させるHtmlコードの該当する部分、網掛け部分のtableタグ

|  |
| --- |
| <ItemTemplate>  <table cellpadding="0" style="border: 1px solid #000000; text-align: center;">  <tr>  <td colspan="2" style="border: 1px solid #000000">  <asp:Image ID="Image2" runat="server" />  </td>  </tr> |

上記のVisualStudioのテーブルをbootstrapのテーブルに変身させるには下記のようにbootstrapのclass名を追加します（VisualStudioはデフォルトの設定でbootstrapが使用できるようになっています）。赤色部分

|  |
| --- |
| <table class=”table” style="border: 1px solid #000000; text-align: center;">  又は、<table class="table table-striped"　style="border: 1px solid #000000; text-align: center;"> |

そして、テーブルの行に背景色を付けるには、下記のようなコードになります。

|  |
| --- |
| <ItemTemplate>  <table class="table" cellpadding="0" style="border: 1px solid #000000; text-align: center;">  <tr class="danger">  <td colspan="2" style="border: 1px solid #000000">  <asp:Image ID="Image2" runat="server" />  </td>  </tr>  <tr class="warning">  <td style="border: 1px solid #000000">商品名</td>  <td style="border: 1px solid #000000">  <asp:Label ID="ProductNameLabel" runat="server" Text='<%# Eval("ProductName") %>' />  </td>  </tr> |

上記の“danger”,”warning”というclass名につては、この項冒頭に紹介したサイトを参照して下さい。Bootstrapのtableに変身して、背景色を設定した場合には、VisualStudioのプロパティウインドウからの枠線（罫線）の作成は不要かと思います。読者様の判断にお任せします。

これで、VisualStudioのテーブルの使用法、の解説を終わります。

# あとがき

この章で作成された、そのものは、実用出来るものとは思えませんが、この章の目的は、VisualStudioの操作方法を知る事、またVisualStudio に備わっているWebサーバーコントロールを知る事とその機能の利用の仕方を経験すること、そしてWebアプリを作成する場合で他の開発作業に比してプログラム効率の高さを実感する事を目的とした章です。この後、「Webショップ構築」シリーズの解説書がありますが、どうしてもこの章を実践しなければ、次の実践の体力がつきませんのでこの章を執筆した次第です。この章の少し物足らない部分と完結しない部分で不満足な面をお詫び申し上げますが、次のステップでは、読者様の満足できる、お役に立てる情報が満載ですのでよろしくお願い申し上げます。