【1C】Delphiチュートリアル セッション

「Delphi 7ユーザと 初心者のためのDelphi 2010入門」

株式会社シリアルゲームズ　取締役
細川　淳

EMBARCADERO TECHNOLOGIES

はじめに

Delphi 7 の紹介

- Delphi 7
  - Borland Delphi 7 Studio
  - 2002年 09月 26日発売
    - 歴代で最も人気のあった Delphi
    - .NET Framework 向け開発機能を提供（プレビュー版）
      - 本格的な .NET Framework 対応は Delphi 8
  - ラインナップ
    - Personal
    - Professional
    - Enterprise
    - Architect

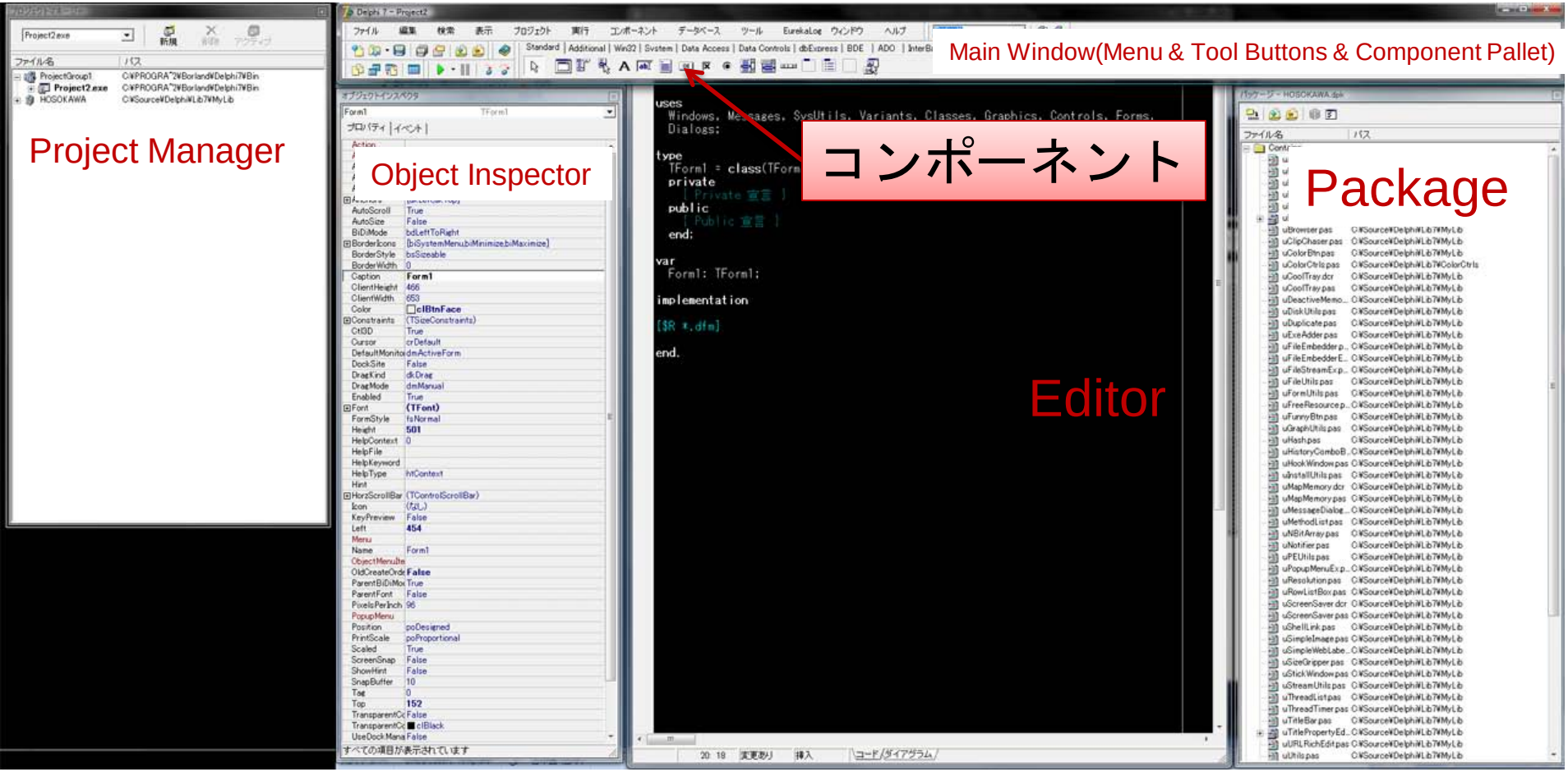
  - 「過去の投資を無駄にすることなく未来へ進むための手段を、多くの企業に提供するという当社の姿勢は変わりません。Delphi 7 Studioをお使いいただければ、開発者の皆さんがすでに構築されたスキルやリソースを使って、それぞれに合ったスケジュールで**スムーズに.NETに移 行していただけます**。モデリング、MDA、レポーティング、クロスプラットフォーム運用など、Delphi 7 Studioに追加された高品質なアプリケーション・ライフサイクル開発ソリューションにより、Delphi向けの新規および既存のアプリケーション開発 の機会が生まれるだけでなく、これから先も革新的なアプリケーション開発の場が提供されることになります」
    ボーランド・ソフトウェア・コーポレーション バイス・プレジデント兼RADソリューション事業ゼネラル・マネージャ サイモン・ソンヒ

Delphi 2010 の紹介

- Delphi 2010
  - 2009年 08月 25日発売
    - RAD Studio 2010 には Delphi 2010, C++Builder 2010, Delphi Prism 2010 が含まれる
    - Windows 7 にいち早く対応
    - ジェスチャーコントロールなどのタッチ対応アプリケーションの開発をサポート
  - ラインナップ
    - Professional
    - Enterprise
    - Architect
  - 「これは簡単で、スピーディだ！Delphi 2010はもうやめられない。これは『**これまでで最高のDelphiだ**』といっても過言ではないよ。本当にすばらしい！」-*Fabricio Pontes de Araujo, Systems Analyst and Programmer IBS - Info Business Solutions*

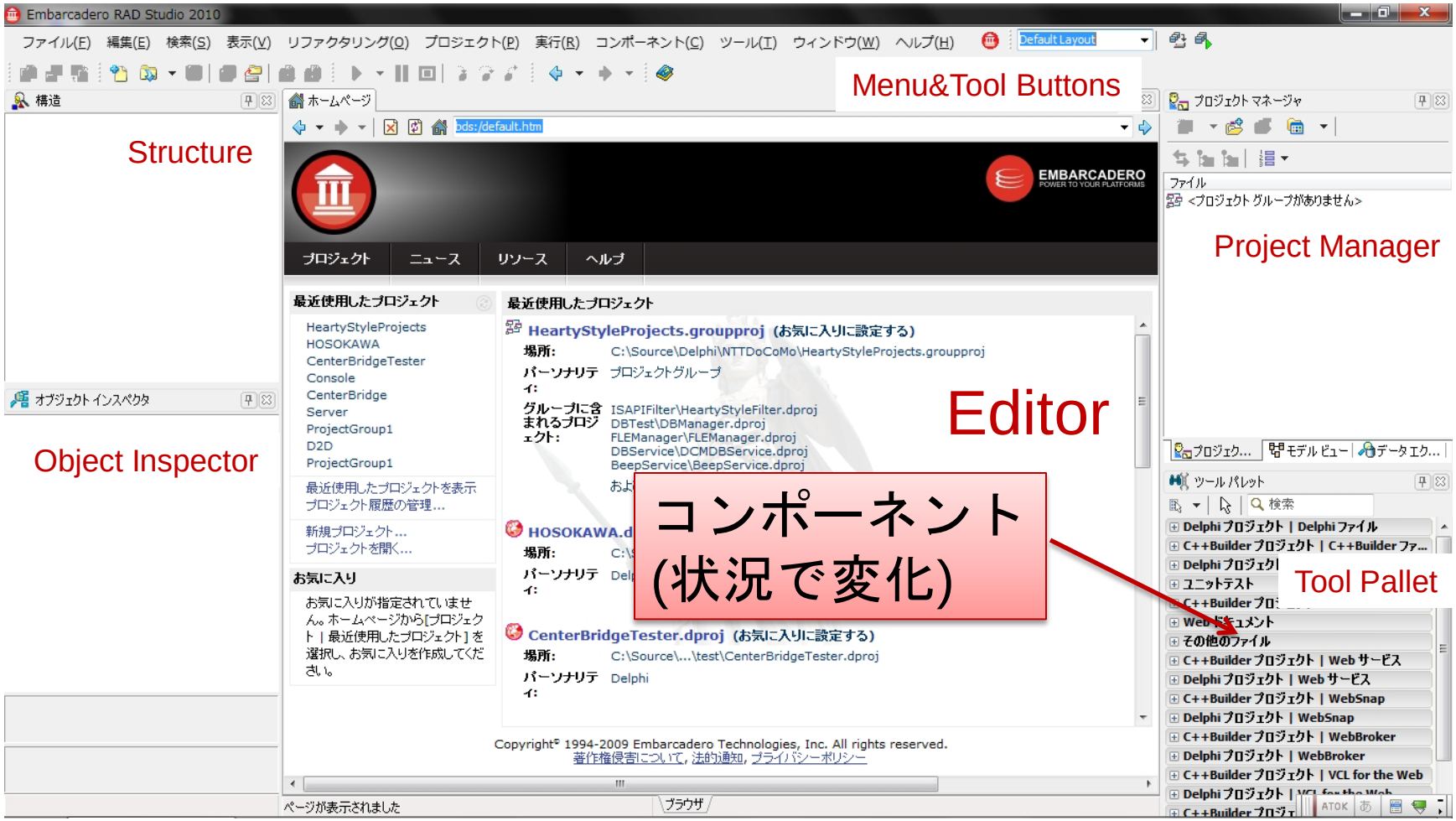
IDE

Delphi 7 の IDE



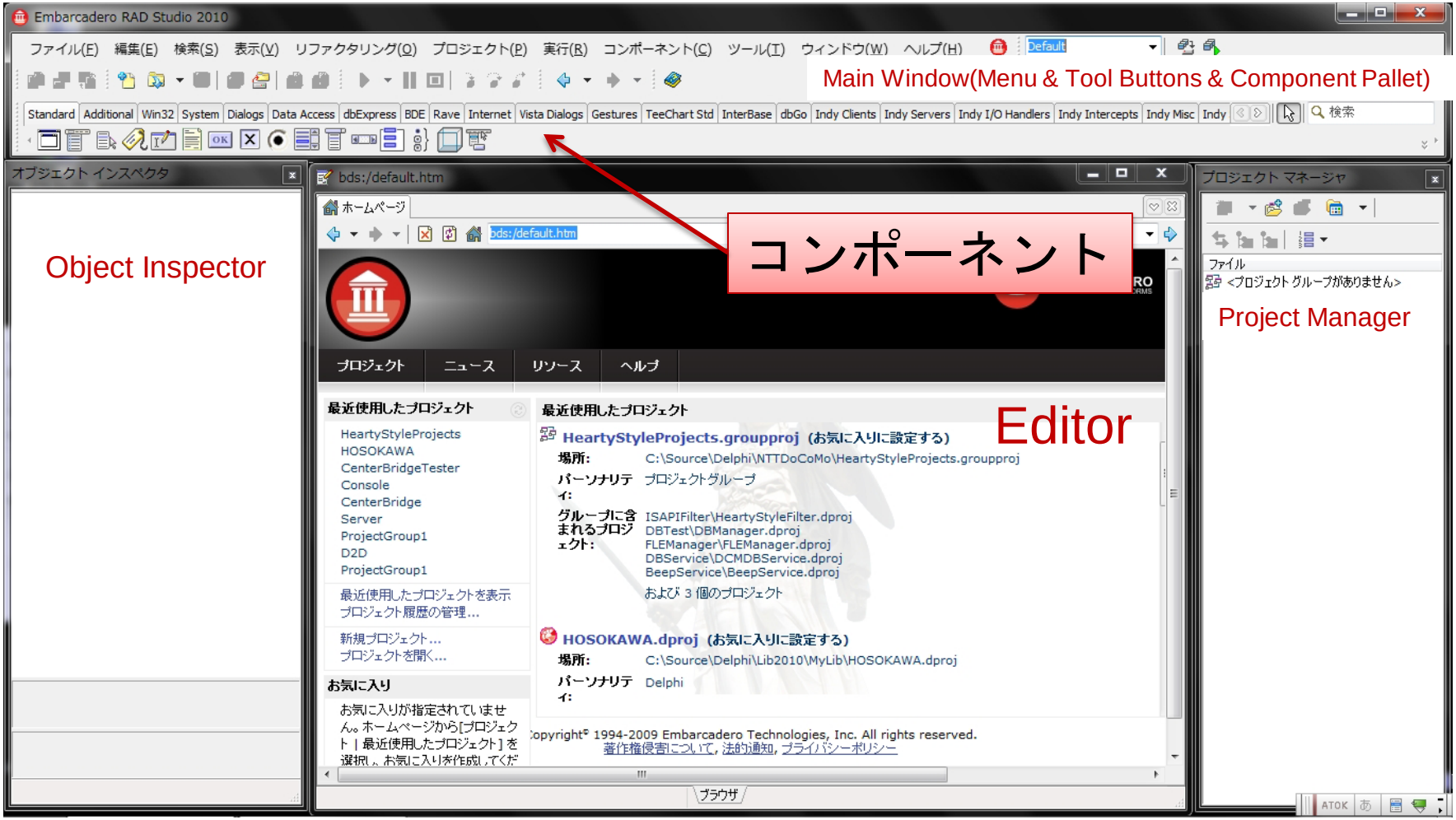
講演者の環境

Delphi 2010 の IDE (Galileo)



Delphi 2010 起動直後の画面
Galileo IDE

Delphi 2010 の IDE (Classic Undocked)



Delphi 2010 Classic Undocked

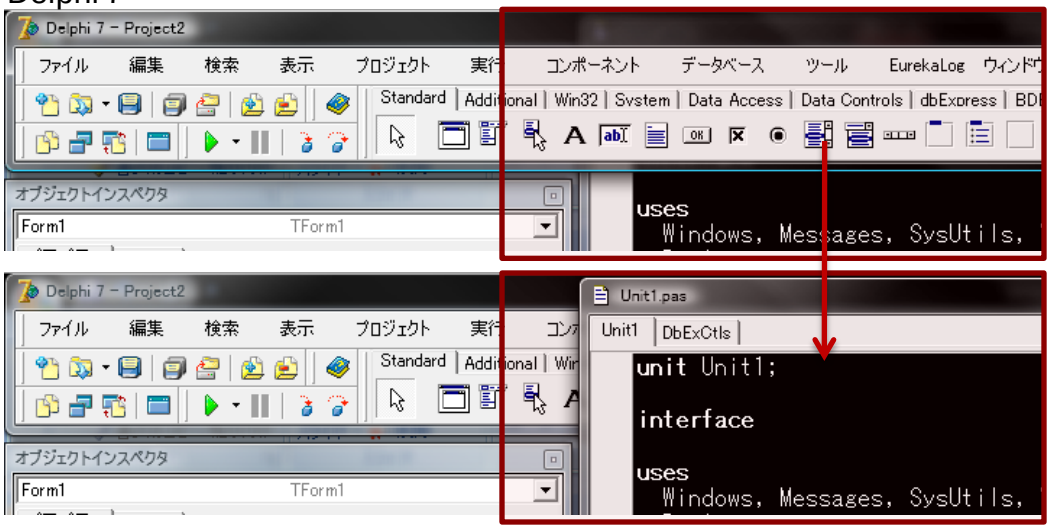
Delphi 2010 Classic Undocked

- 設定するには……
  - 「表示」→「Classic Undocked」を選択
    - 各ウィンドウが分離
  - 「ツール」→「オプション」→「ツールデザイナ」→「埋め込みデザイナ」のチェックを外す
    - デザイナが分離

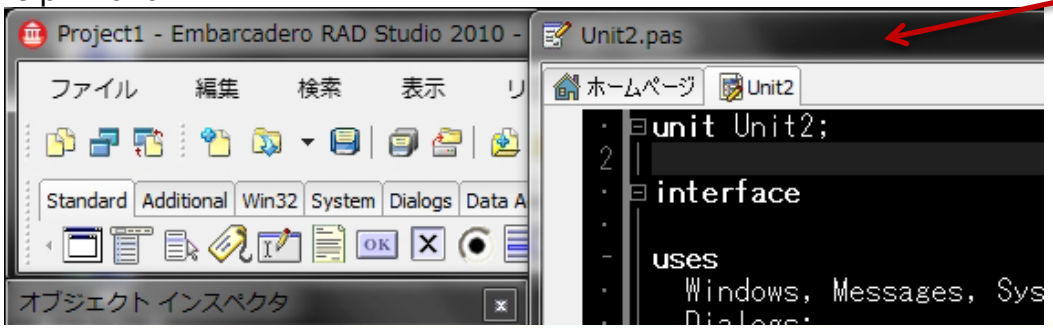
Delphi 2010 Classic Undocked のバグ

Z順序が入れ替わらない！

Delphi 7



Delphi 2010



Delphi 7
エディタを縦方向に最大化して使っている（例えばノートとかで縦の解像度が少ない場合）時、コンポーネントを置くには、メインウィンドウをクリックすればZが入れ替わり、コンポーネントパレットにアクセスできました。
そのためノートでも快適に作業ができたのですが……

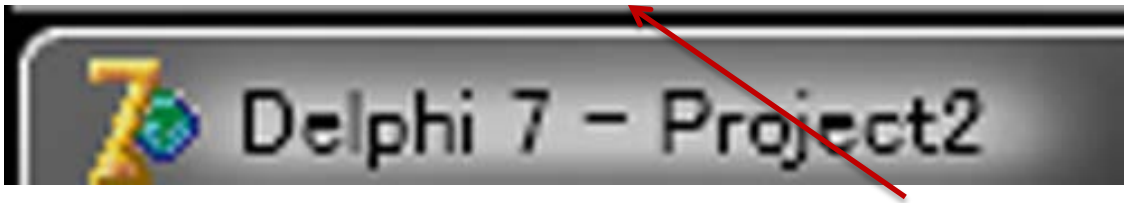
Delphi 2010
Zが入れ替わりません！
常にエディタがトップにいます

エディタが一番上に。
コンポーネントパレットにアクセスできません

Delphi 2010 Classic Undocked のバグ

スナップが切れない！

Delphi 7



ランチャ（クリックで出現）

Delphi 7
メインウィンドウは自由に動かせました。
そのためタスクバーやランチャーを画面上部に置いてもメインウィンドウを数ドット下に下げるだけで、それらを利用できました。

Delphi 2010
メインウィンドウがスナップするため、数ドット下に下げる、という操作ができません！
画面上部にタスクバーやランチャを置いている場合致命的です。

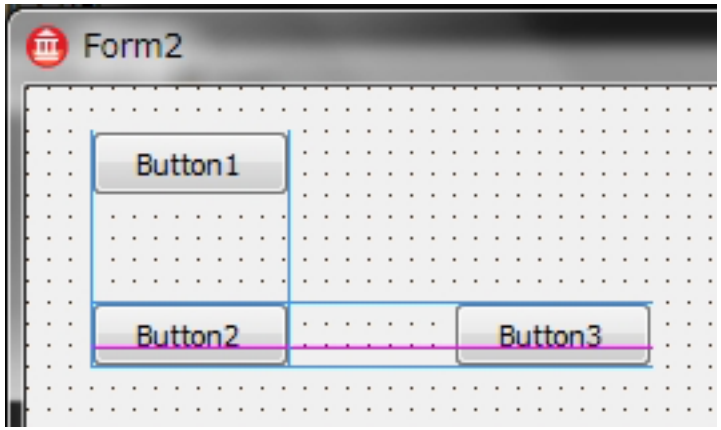
Delphi 2010

モニタ上部にピッタリくっついてしまう！

無理矢理下に下げるとこんなに空隙が！

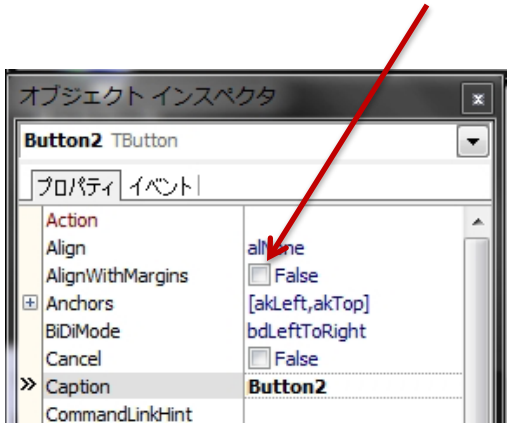
Delphi 2010 IDE のスゴイところ

- コンポーネントの位置合わせが楽！
  - Delphi 2005 くらい？で実装



最近のデザイナでは当たり前の機能

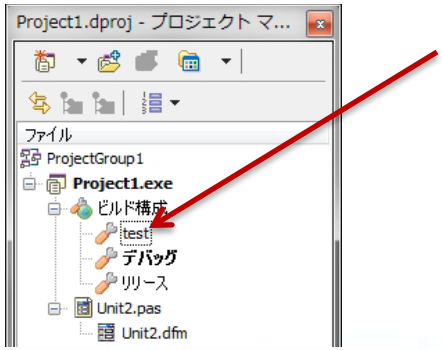
- オブジェクトインスペクタが変わった！
  - True, False を選ぶ物など、集合型では、各要素に
  - チェックボックスが付いて、より簡単に選択できる！
  - 他にもDate型はカレンダーコントロールで設定できたり
    いくつかの型でUIが更新されています



Delphi 2010 IDE のスゴイところ

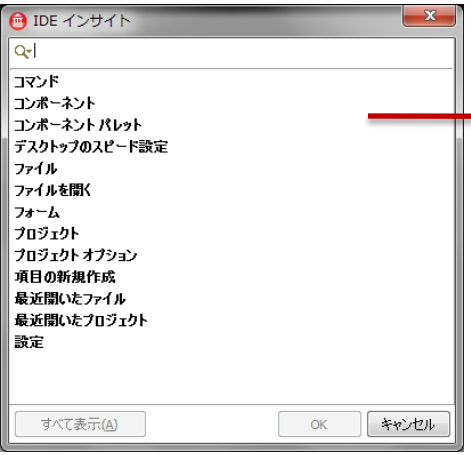
- MSBuild の使用
  - Delphi 2007 で採用
  - ビルド前、ビルド後に実行するコマンドを指定できるようになりました。
    - DOS のコマンドが使えます。
    - 例えば、できあがったファイルを所定の位置へコピーしたりできます。

- プロジェクトマネージャの機能強化
  - ビルド構成を任意に作成可能
    - 例えばコンパイラオプションや、警告、アプリケーションのタイトルなどなど、オプションを切り替えられます。

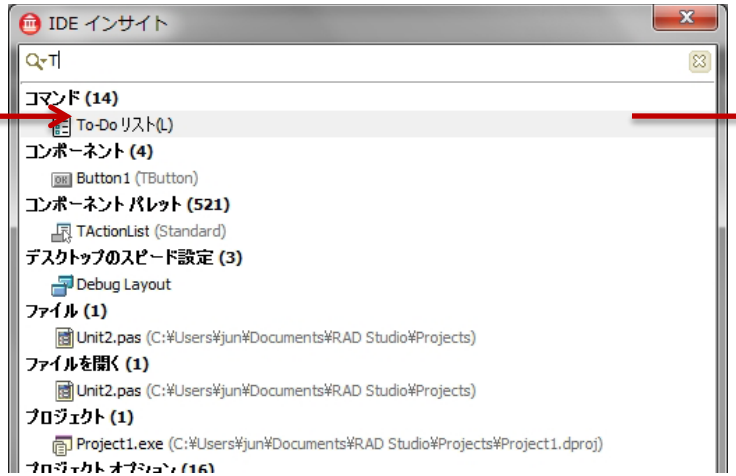


Delphi 2010 IDE のスゴイところ

- IDE インサイトがスゴイ！
  - F6 キーで起動
  - マウスを使わずに IDE の各機能を呼び出せる！

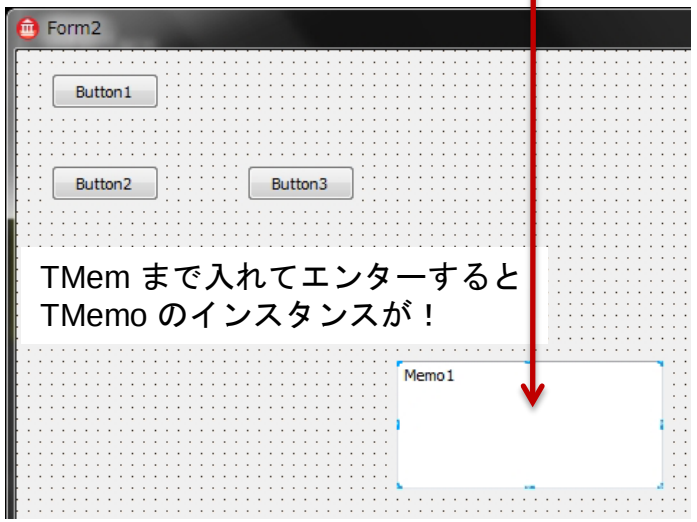


起動直後



「T」と入力
このとき IDE の要素全てから T が関係する物をリストアップ
続けて文字を入力することで、どんどんインクリメンタルに対象が絞られる

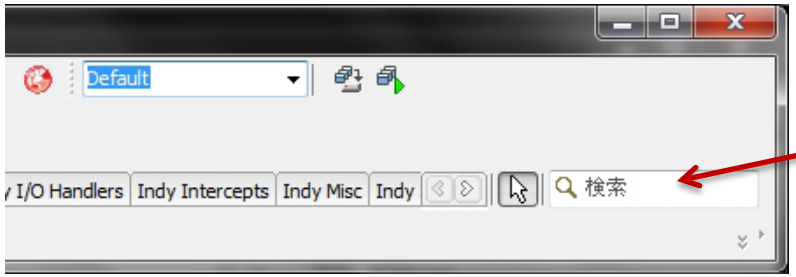
続けて「TMe」まで入れると、TMediaPlayer と TMemo が該当



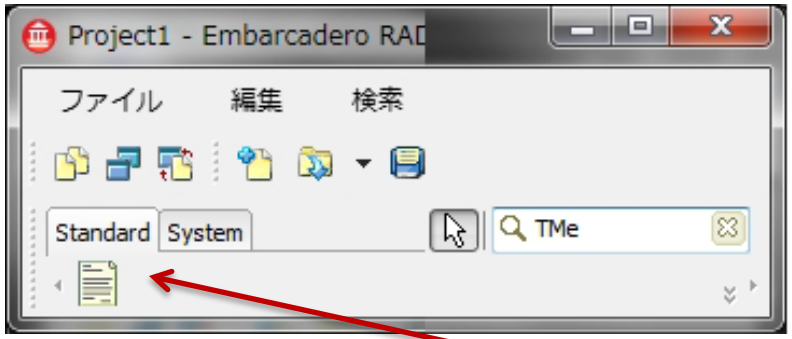
TMem まで入れてエンターすると TMemo のインスタンスが！

Delphi 2010 IDE のスゴイところ

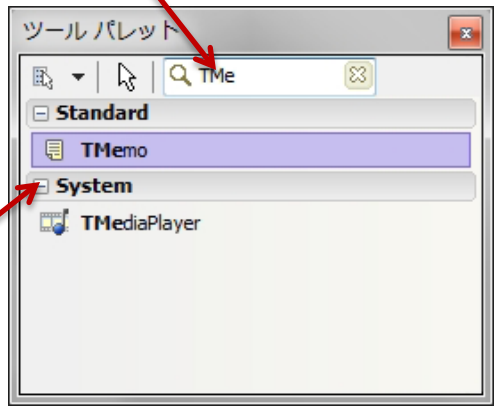
- コンポーネントパレット＆ツールパレットでコンポーネントの絞り込み
  - 地味に便利（IDE インサイトで代用できますが）



ココに文字を入力するとインクリメンタルに絞り込みされる



コンポーネントパレット



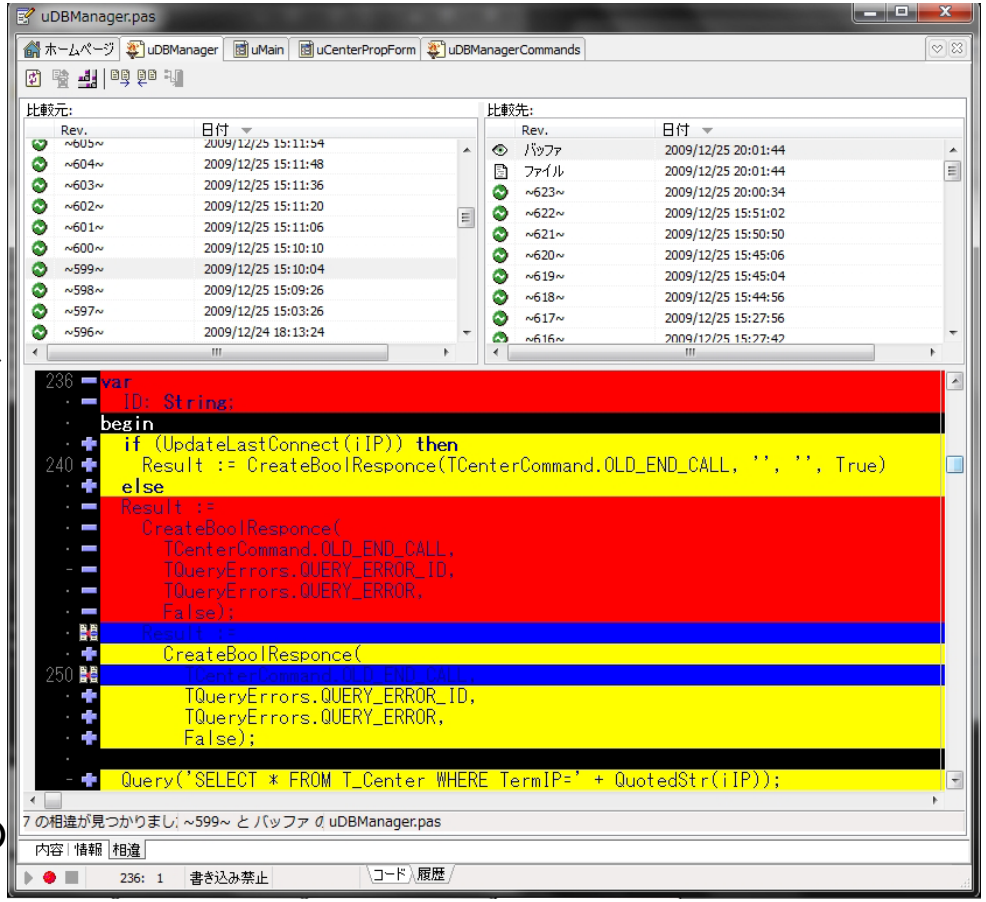
ツールパレット

TMe まで入れると該当する TMemo とTMediaPlayer のみ表示される

Delphi 2010 IDE のスゴイところ

- 他には
  - IDEの言語切り替えツールの提供
    - エラーメッセージなどのリソースを切り替えられます。
    - 国際化対応に便利
  - IDE に統合されたファイルブラウザ
    - IDE からファイルが参照できます
  - クラス図と UML モデリング
  - トランスレーションエディタ
  - 履歴タブ
    - 自動的にローカルで履歴を取っています。
      昔のソースとの違いを見たり、昔のソースと入れ替えたりもできます。

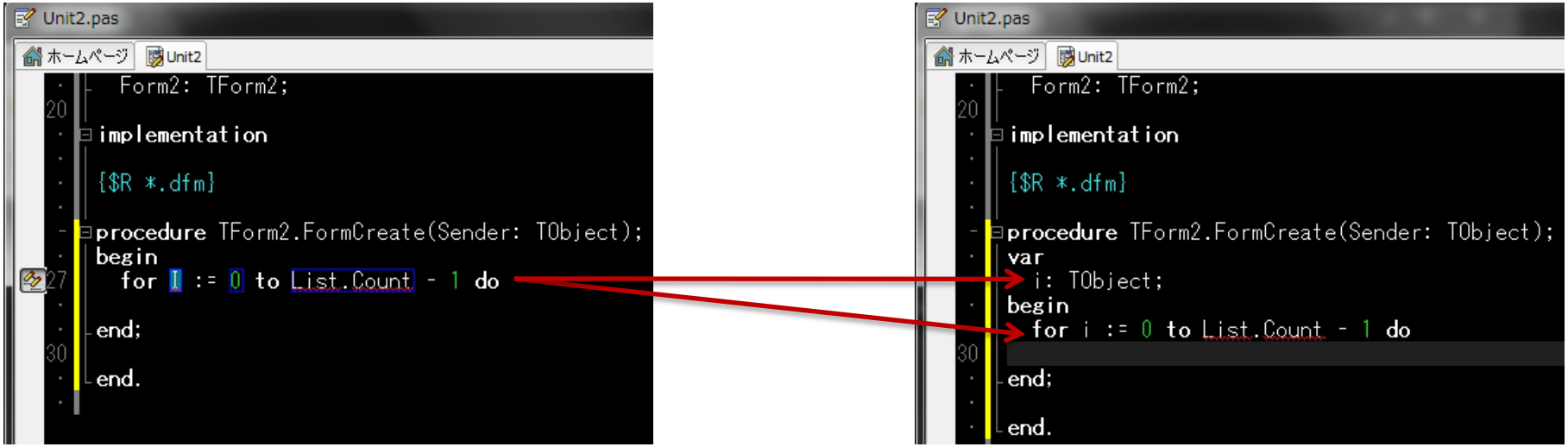
……など



履歴タブ

Delphi 2010 エディタのスゴイところ

- ライブテンプレートがスゴイ！



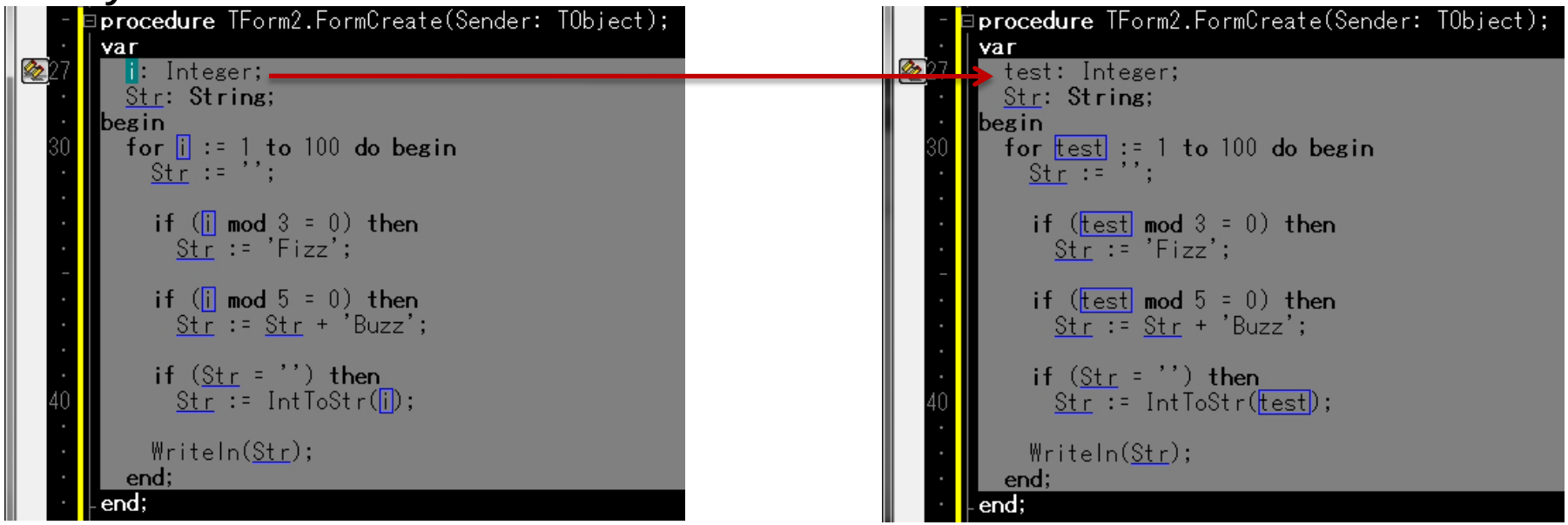
Delphi 7 までの Code Template に変わって、Live Template が導入されました。
これは、もっとインテリジェントなテンプレートです。
これを使えば、自動的に変数を宣言させるなどコーディングをある程度自動化できます。

上の例は、for 文のテンプレートです。
自動的に変数が宣言されているのが分かります。

Live Template は、XML で記述されています。自分で Live Template を作ることもできます。

Delphi 2010 エディタのスゴイところ

- SyncEdit がスゴイ！

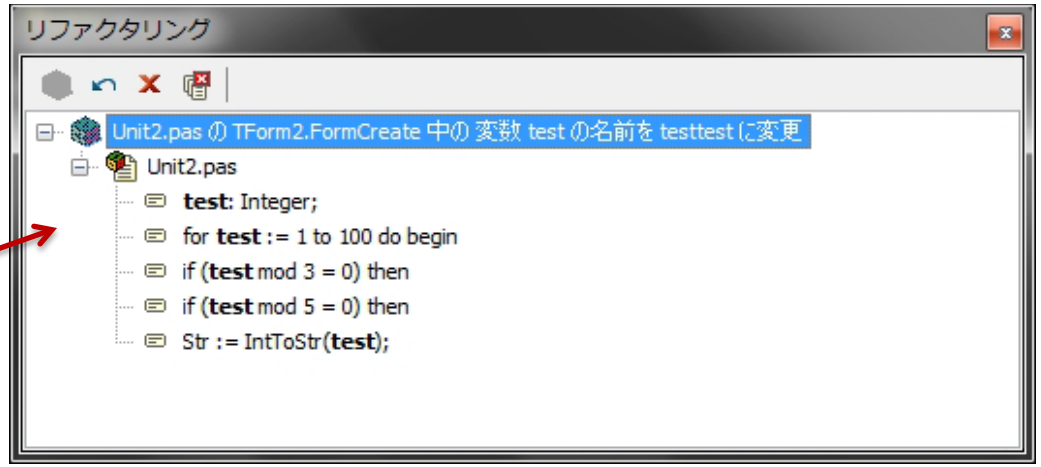


SyncEdit はスゴイ機能です。簡単なアイデアなのに Delphi IDE 以外の開発環境で見たことがありません。
選択範囲内の変数名を、同期的に（Sync）編集（Edit）してしまうのです。

上の例では、変数「i」を変数「test」に変えています。
実際に編集したのは一番上の「i」だけです。しかし選択範囲無いにある「i」（青枠が付いている）は全て同期的に変更されています

Delphi 2010 エディタのスゴイところ

- リファクタリングがスゴイ！
  - 豊富なリファクタリングコマンド
    - 名前の変更
    - メソッドの抽出
    - 変数の宣言
    - 変数の導入
    - 変数のインライン化など



リファクタリングー名前の変更で、変数名を変更する例
Delphi は他のエディタや開発環境に比して、豊富なリファクタリング手段を提供しています。
しかも、Delphi 2010 のリファクタリングはジェネリクスにも適用できます。
また、他のユニットで使われているプロパティやメソッドも探しだし、正しくリファクタリングを適用します。
単なる名前変更程度にとどまらない、Delphi のパワフルなリファクタリングを是非体験してください。

Delphi 2010 エディタのスゴイところ

- 変更バー

変更があった部分に変更バー（黄色の部分）が表示されます。

- ブロック補完
  - begin といれてエンターを押すと自動的に end が補完されたりします。
    - 慣れるまでは自分で end を入れたいのに！と思いますが、慣れてしまうと、補完されないのは非常にイライラするようになります。

- Unicode 対応
  - Delphi 2009 からエディタが完全に Unicode 化されました。
    - 日本語の変数名も使えます（是非は置いておいて）

Delphi エディタで知っておくと便利なコマンド

- ブックマーク
  - 個人的に大プッシュしたい機能がブックマークです。
  - Ctrl + Shift + 0～9 キーで、ブックマークを設定
  - Ctrl + 0～9 キーで、設定したブックマークに移動
    - いまここを編集してるんだけど、他のところを、ちょっと見たい！という時に、ブックマークを設定して、他のところを見に行って、直ぐ戻ってきたり、という風に使います。
- 宣言部、実現部の移動
  - Ctrl + Shift + ↑↓
    - 実現部で↑を押せば宣言部に、宣言部で↓を押せば宣言部に移動します。
- 自動的に実現部のスケルトンを作る
  - Ctrl + Shift + C
    - 例えば宣言部で procedure Test(const iStr: String); と書いて Ctrl + Shift + C と押すと
    - procedure Test(const iStr: String);
    - begin
    - end; と実現部のベースが自動的にできあがります。

Delphi エディタで知っておくと便利なコマンド

- インデント・アンインデント
  - Ctrl + Shift + I でインデント
  - Ctrl + Shift + U でアンインデント
- コード補完
  - Ctrl + J
    - tryf などの登録されているテンプレートに対して Ctrl + J を押すと

```
tryf

finally
end;
```

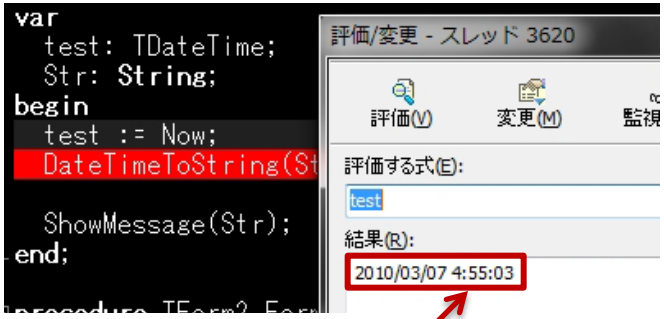
    - に補完されます
- SyncEdit
  - Ctrl + Shift + J
    - 選択されている部分を同期編集します

Delphi エディタで知っておくと便利なコマンド

- 一行削除
  - 個人的によく使うのが一行削除です。これで空行を削除したりします。
  - Ctrl + Y
- ハイパーリンク
  - Ctrl を押しながら型や変数、関数などをクリックすると宣言部にジャンプします

Delphi 2010 はデバッガもスゴイ

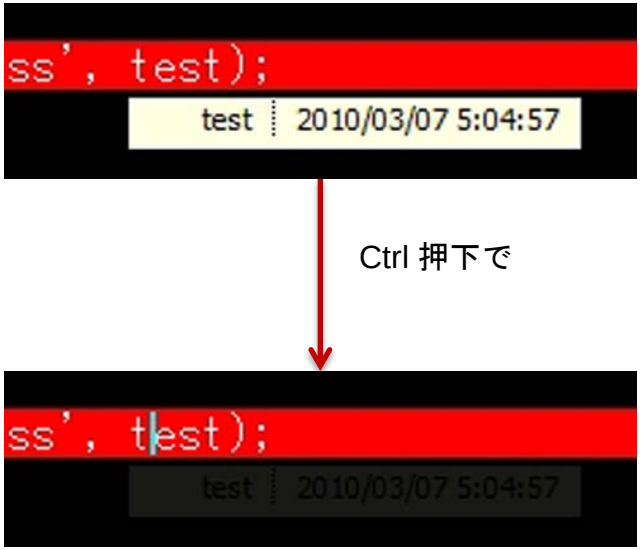
- デバッガ
  - デバッガには、ビジュアライザが搭載されました。
    - TDateTime 型なら日付で、TStrings なら文字列のリストが、といった具合に型に即した表示ができます。
    - ビジュアライザは、自分でも作成できます。
  - スレッドのデバッグ
    - スレッド毎に、実行・停止・再開を制御できます。
  - ユーザーブレイクポイント以外を無視
    - ユーザーのブレイクポイント以外の例外や停止は、そのまま続行されます。

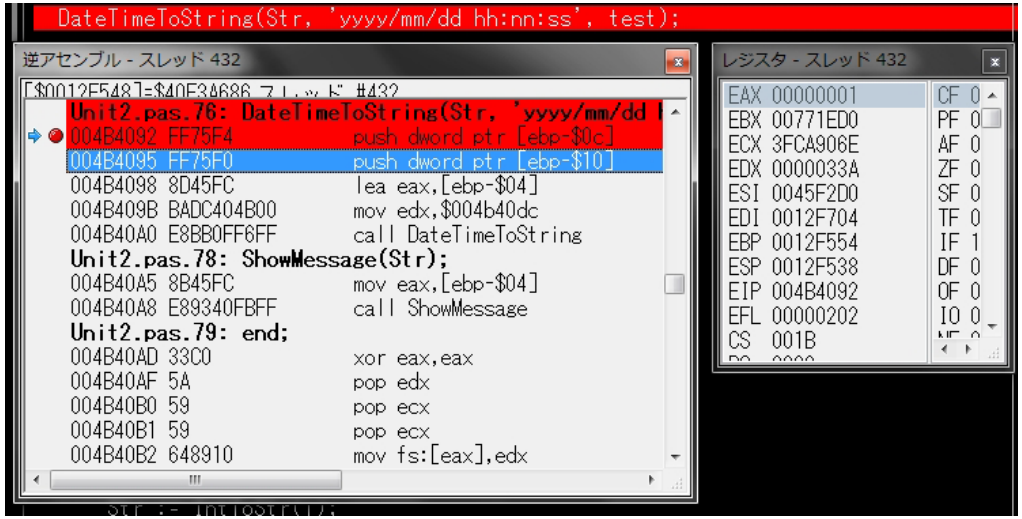


TDateTime 型のビジュアライザ

Delphi 2010 はデバッガもスゴイ

- デバッガ
  - CPU ビューは機能別に独立可能に。
  - Ctrl を押すとデータ表示ツールチップが半透明に
    - ツールチップが邪魔でコードが見えなかった時に便利。





逆アセンブル結果とレジスタの内容を独立したウィンドウで

その他もスゴイ

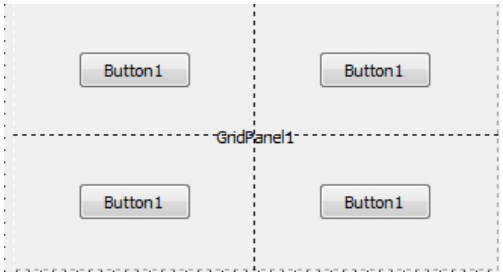
- 統合されたユニットテスト
  - DUnit をベースにユニットテストフレームワークが統合されました。
- データベース接続だってスゴイ
- Web アプリケーションの開発がスゴイ
- フォーマッタとかもある！
- とにかくスゴイ

VCL

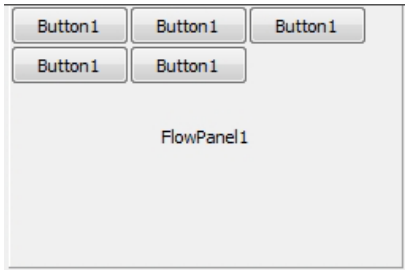
Delphi 7 以降に導入された VCL

- 多くの VCL が新たに導入されています。
  - いくつかピックアップします。

- TGridPanel
  - グリッドに分割された領域に1つコンポーネントが置かれるパネルです



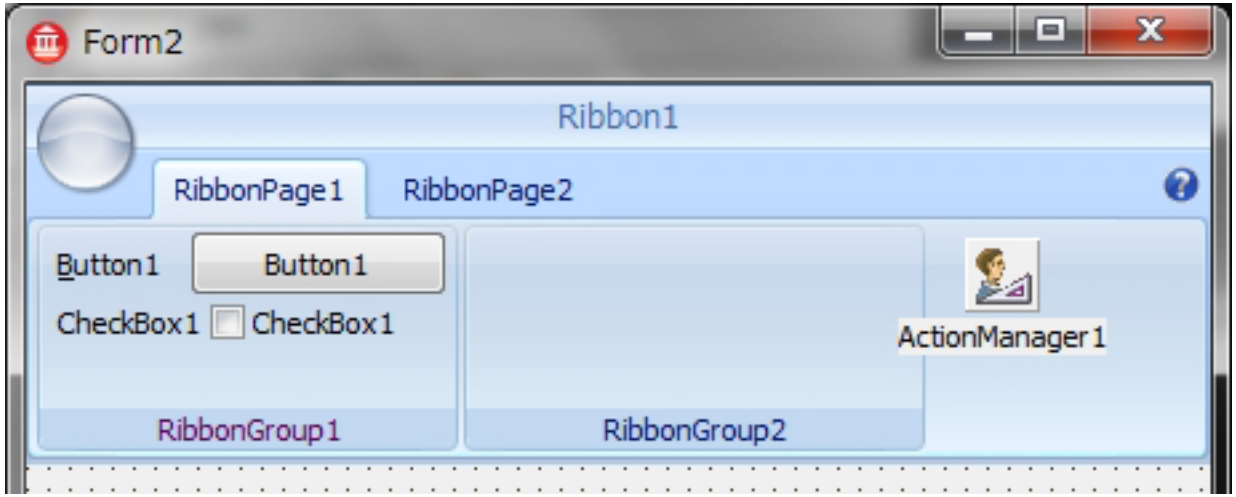
- TFlowPanel
  - 自動的に整列するパネルです。



- TCategoryPanelGroup
  - 最近よく見る開閉式のパネルのグループです。

Delphi 7 以降に導入された VCL

- リボンコントロール
  - 賛否両論のリボンコントロールが使えます。
    - 個人的にはリボンコントロールは賛成派です。

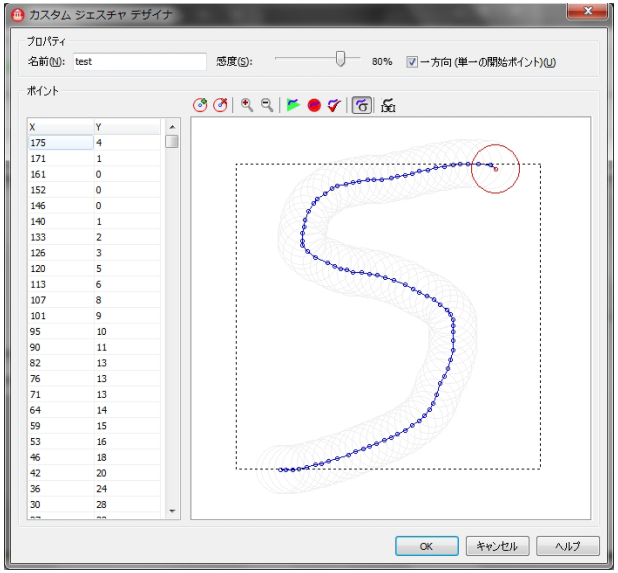


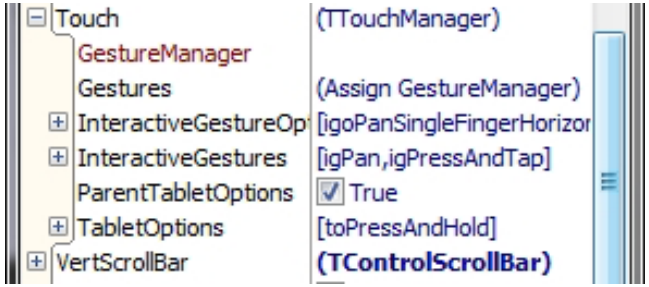
Delphi 7 以降に導入された VCL

- Vista スタイルのダイアログ
  - TFileOpenDialog, TFileSaveDialog, TTaskDialog
- タスクトレイアイコン
  - TTrayIcon
  - TBalloonHint
- Indy
  - Indy はバージョン 10 に。
- その他にも色々追加されています。

Delphi 2010 で導入された VCL

- ジェスチャーコントロール
  - TGestureManager
    - ジェスチャーの管理・作成を受け持ちます。
    - 右図のようにジェスチャーを自分で作成できます。
    - 標準的なジェスチャーは最初から用意されています。
    - 自作したカスタムジェスチャーも標準ジェスチャーも
      コンポーネントの Touch プロパティに結びつける
      ことで機能を自由に組み込むことができます。





Touch プロパティ。ここに GestureManager を結びつけ、Gestures でジェスチャーを選択する。
登録されたジェスチャーが実行されると、イベント OnGesture が発生する。

Delphi 2010 で導入された VCL

- TTouchKeyboard
  - タッチパネル用にデザインされたタッチキーボード
  - ちゃんと漢字も打てます。

Delphi 2010 で追加されたクラス

- TMonitor
  - 個人的にお勧めしたいクラスです。
  - System ユニットに含まれるクラスで同期処理を実現します。
    - EnterCriticalSection, LeaveCriticalSection のような機能です。
  - System.TMonitor.Enter(AObject: TObject);
    - これで AObjectオブジェクトをモニターとして、ロックを取得します
      - EnterCriticalSection などと比べて優れているのは、引数に TObject のインスタンスを取るところです。
      - つまり、全てのクラスのインスタンスを使えると言うことを意味します。
      - そして、インスタンスが違えば、同期処理も別々に行われると言うことになります。
        - » Indy を利用した通信クラスなどで、別々のインスタンスで同期が取れるのは非常に便利です

```
procedure TForm1.Button1Click(Sender: TObject);
begin
  System.TMonitor.Enter(Self);
  try
    // 処理
  finally
    System.TMonitor.Exit(Self);
  end;
end;
```

Enter から Exit までのブロックは別スレッドからのアクセスがブロックされます。
Thread を処理しているときなど、何度もやってくる場合があっても
Enter ～ Exit 間は、一度に１つのスレッドしか処理できません。
ロックを取得しているスレッドの処理が終わるまで他のスレッドは Enter でブロックされます。
もちろん同一スレッド内では、ロックの複数回取得(再入)は可能です

Delphi 2009, 2010 で追加されたクラス

- TDirect2DCanvas
  - Windows 7 で追加された Direct2D を扱えるキャンバスです。
  - これを使うと簡単に Direct2D の機能を扱えます。

- TStringBuilder
  - 文字列の追加・挿入・置換、などを高速に行うクラスです。

Delphi 2009, 2010 で拡張されたTObject

- TObject には、下記の仮想メソッドが追加されました
  - ToString
  - GetHashCode
  - Equals

- RTTI サポート
  - 今までの RTTI 用メソッドよりも詳細かつ柔軟な RTTI サポートが追加されました。

Delphi 2010 はヘッダファイルがスゴイ

- Delphi 2010 では、Win32 SDK で配布されているヘッダファイルのほぼ全てが Delphi 用のユニットに変換されています

- 今までのように自分でヘッダファイルを変換する前に、下記のフォルダを検索する用にしましょう。
  - RAD Studio¥7.0¥source¥Win32¥rtl¥win¥

言語

Delphi 2009 で変更された文字列型

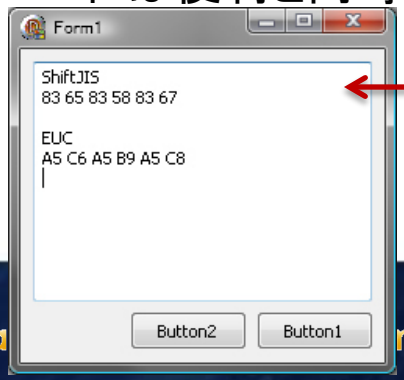
- UnicodeString の導入
  - Delphi 2009 から言語・IDE は Unicode を基本文字セットとするようになりました。
  - これによって文字列型が変更されました。

文字列型名	特徴	用途
String	デフォルトの文字列型	UnicodeString に同じ
ShortString	8ビットANSI文字列	下位互換用
AnisString	8ビットANSI文字列	ASCIIやシフトJISなど
WideString	Unicode 文字列	COM 用
UnicodeString	Unicode 文字列	デフォルトの文字列
RawByteString	コードページの変換がない	文字列そのままのデータが欲しい時。バッファとして利用する時など

Delphi 2009 で変更された文字列型

- AnsiString にコードページを指定可能に
  - AnsiString にはコードページを指定できるようになりました。
  - 例えば右記のコードです。
    このコードでは AnsiString に
    コードページを指定した
    新しい文字列型を作っています。
  - ここで作られた新しい文字列型は相互に
    代入互換性があります。
    代入時に自動的にコードページの変換が
    おこります。
    これは便利と同時に、注意すべき点です。

```
type
  SJISString = type AnsiString(932);
  EUCString = type AnsiString(20932);
var
  SJIS: SJISString;
  EUC: EUCString;

  procedure ShowCode(const iCode: String; const iStr: RawByteString);
  var
    i: Integer;
  begin
    Memo1.Lines.Add(iCode);

    for i := 1 to Length(iStr) do
      Memo1.Lines.Text := Memo1.Lines.Text + IntToHex(Ord(iStr[i]), 1) + ' ';

    Memo1.Lines.Add(sLineBreak);
  end;

begin
  SJIS := 'テスト';
  EUC := 'テスト';

  ShowCode(SJIS);
  ShowCode(EUC);
end;
```



右記のコードを実行した結果

Delphi 2009 で変更された文字列型

- RawByteString とは？
  - 前ページで見たように、文字列型は相互に代入互換性があります。
    - そのため、意図しない変換や、ループ内での代入によるCPUリソースの消費などの問題があります。
  - RawByteString は、コードページの変換を伴わない生のデータを扱う文字列型です。
    - RawByteString を使えば、意図しない変換は起きません。
  - String 型をバッファとして使うテクニックにも利用できます。
    - 例えば、このように。

```
begin
  SetLengt(Str, 1024);
  ポインタを使う関数(PRawByteString(Str)^, foo, bar);
end;
```

Delphi 2009 で変更された文字列型

- TEncoding クラスの導入
  - TEncoding クラスは、様々なエンコードを表すクラスです。
  - このクラスを使うと、文字列を扱うクラス（TStringList など）でコードページを指定できるようになります。
  - 例えば、下記の用に使えます。

```
procedure TForm1.Button1Click(Sender: TObject);
var
  EUC: TEncoding;
begin
  with TStringList.Create do
    try
      Add('テスト');

      EUC := TEncoding.GetEncoding(20932); // EUC-JP のインスタンスを取得
      try
        SaveToFile('C:¥Temp¥EUC.txt', EUC);
      finally
        EUC.Free;
      end;
    finally
      Free;
    end;
end;
```

左記のコードでは文字列が EUC として扱われます。

Delphi 7 以降変更された Class

- Class に Delphi 2005(※) で下記の可視性が追加されました

  ※Delphi 8 から追加されたものの Win32 で使えるように成ったのは Delphi 2005 です。

  – strict protected
  – strict private
  – Delphi 言語では、同一ユニット内のクラスは互いに private, protected が見える、という一種の friend 的機能が使えましたが、strict を付けると他のクラスから見えなくなります。

- Class 変数、定数
  – Class には、Class 変数、定数を指定できるようになりました。

```
type
  TTest = class
    class var SFoo: Integer;
    const CBar: Integer = 10;
  end;

  TCommands = record
    const COMMAND_SEND: String = 'send';
    const COMMAND_RECEIVE: String = 'recv';
  end;
```

左記の例での SFoo は TTest.SFoo としてアクセスできます。
同様に CBar も TTest.CBar としてアクセスできます。

実は、Class 同様 Record にも定数を定義できるようになりました。
これを使えば、グループ化された定数を宣言できます。

ジェネリクス

- Delphi 2009 からジェネリクスが導入されました。
  - Java ではおなじみ、C++ で言うところのテンプレートのような物です。
  - 今までは、型が違うだけの同じ処理は、泣く泣く、同じアルゴリズムを書いていました。ジェネリクスを使えば、同じアルゴリズムのクラスを何度も書く必要はありません。

```
type
  TTest<T> = class
  private
    FValue: T;
  public
    constructor Create(const iValue: T); reintroduce;
    property Value: T read FValue;
  end;

constructor TTest<T>.Create(const iValue: T);
begin
  FValue := iValue;
end;
```

ジェネリクスの例。ここでは String と Integer という異なる型でも、同じクラスを利用できることを示しています。

```
procedure TForm1.Button1Click(Sender: TObject);
var
  Str: TTest<String>;
  Int: TTest<Integer>;
begin
  Str := nil;
  Int := nil;
  try
    Str := TTest<String>.Create('テスト');
    Int := TTest<Integer>.Create(4649);

    Memo1.Lines.Add(Str.Value);
    Memo1.Lines.Add(IntToStr(Int.Value));
  finally
    Str.Free;
    Int.Free;
  end;
end;
```

無名関数

- Delphi 2009 から無名関数(匿名関数とも)がサポートされました。
  - 無名関数は、関数レベルでの多態性を確保します。

```
type
  // 匿名関数の型定義
  TFigureFunc = reference to function(const iMsg: String): String;

procedure ShowFigure(const iFigureFunc: TFigureFunc);
begin
  ShowMessage(iFigureFunc('辺の数は'));
end;
```

驚くべき事に、無名関数は関数内関数と同じようにローカル変数にアクセスできます。
この例では Side という変数にアクセスしています。
Side の数によって ShowFigure 関数に渡す無名関数を選択しています。
このように無名関数を使うと、関数レベルで多態性を確保できます。

```
procedure TForm2.Button1Click(Sender: TObject);
var
  Triangle: TFigureFunc;  // 匿名関数を代入する
  Rectangle: TFigureFunc; // 変数
  Side: Integer;
begin
  Triangle := function(const iMsg: String): String
  begin
    Result := Format(iMsg + ' %d, 三角形', [Side]);
  end;

  Rectangle := function(const iMsg: String): String
  begin
    Result := Format(iMsg + ' %d, 四角形', [Side]);
  end;

  Side := 3;

  if (Side = 3) then
    ShowFigure(Triangle);

  if (Side = 4) then
    ShowFigure(Rectangle);
end;
```

まとめ

Delphi 2010 へ今すぐ行こう！

- Delphi 2010 は非常に安定したバージョンです。
- 起動も高速！まるで Delphi 7 の再来です。
- しかも、時代の要請に応えた様々な機能追加、言語使用の拡張が施されています。
  - もう Delphi って Unicode じゃなかったの？って言われない！！
- Delphi 2010 は、これからの時代の Delphi のベースとなるバージョンとなるでしょう。

参考文献

- Delphi 2010 機能一覧
  - delphi-2010-features-matrix-jp.pdf
- Delphi 7以降の IDEの新機能
  - http://edn.embarcadero.com/jp/article/34361
- あなたの環境でもきっと役に立つDelphi 7以降の新機能トップ10 - by Pawel Glowacki
  - http://dn.embarcadero.com/jp/article/37433
- Delphi 2010の新機能
  - http://www.embarcadero.com/jp/products/delphi-2010-what%E2%80%99s-new
- Delphi 2009で追加された3つの便利な機能をコードで検証
  - http://techtarget.itmedia.co.jp/tt/news/0811/11/news02.html
- Delphi 2010 のヘルプ