embarcadero
Developer Camp

【T4】テクニカルセッション

「 FireMonkey ファーストインプレッション」

有限会社　エイブル
富永 英明

embarcadero
Developer Camp

はじめに

FireMonkey の存在意義

FireMonkey で使用する軽量 GUI コントロールのベースには、クロスプラットフォーム用の抽象表現があり、それが Windows、Mac OS X、iOS 用に実装されています。

軽量コントロールとはすべてのピクセルを FireMonkey で描画するという意味で、ネイティブの（重量）コントロールは使われません。

このアプローチでは、ホスト プラットフォームに対する忠実度よりもプラットフォームをまたがった忠実度を優先するため、重量クロスプラットフォーム フレームワークにつきものの "最小公倍数" の問題を回避し、FireMonkey で独自のコントロールやアプリケーション設計規則を作成することができます。

DocWiki「FireMonkey アプリケーションの設計」より

イロイロとゴタクを並べようと思いましたが...

- 他にやりたい事があるので省略します。

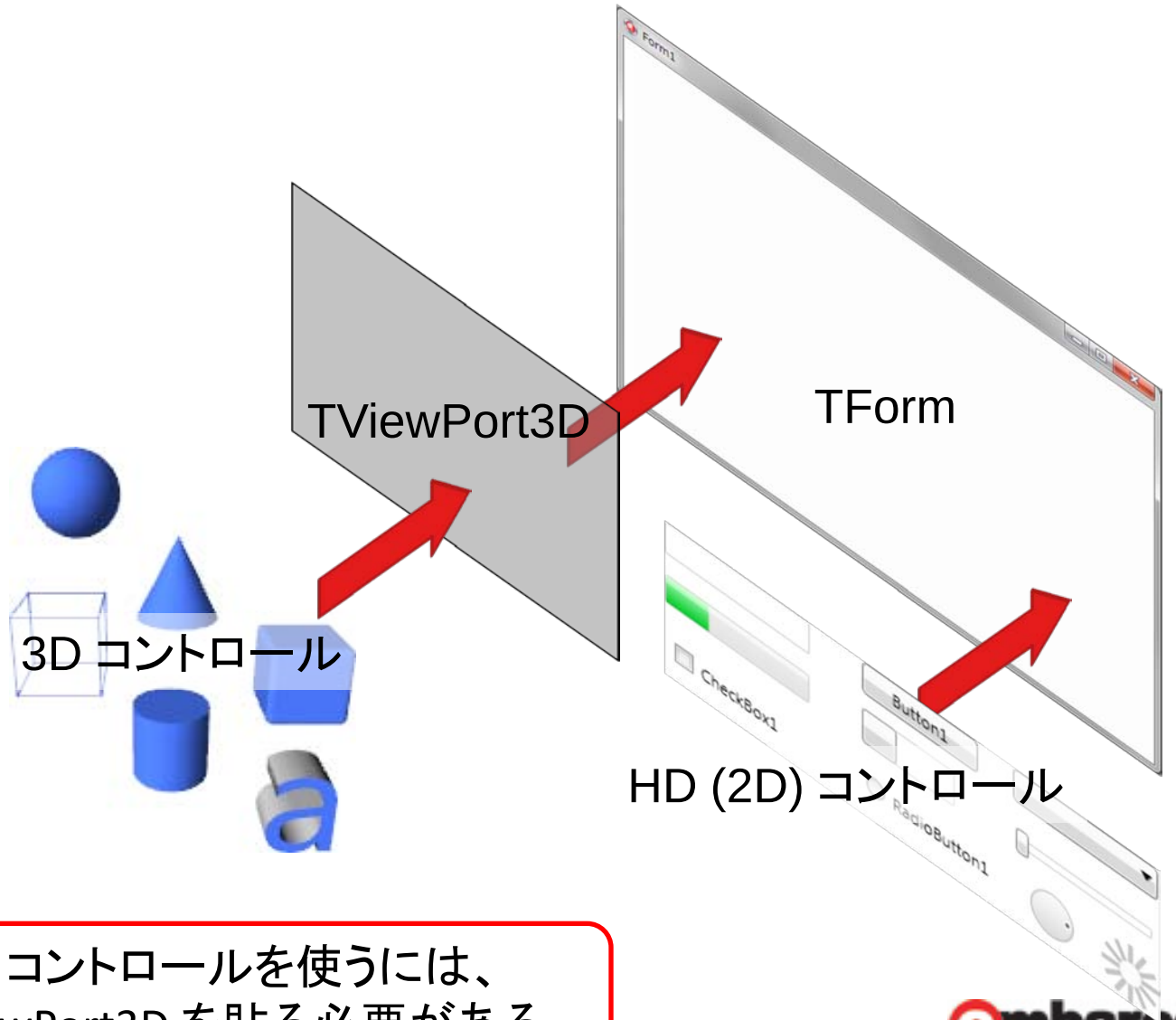
<!> WARNING <!>
GotoWebinar 経由で参加している方は、
この時間を利用してスクリーンショットを撮る
ツールをご用意下さい。

※ 後で資料配布されるとは思いますが...。

embarcadero
Developer Camp

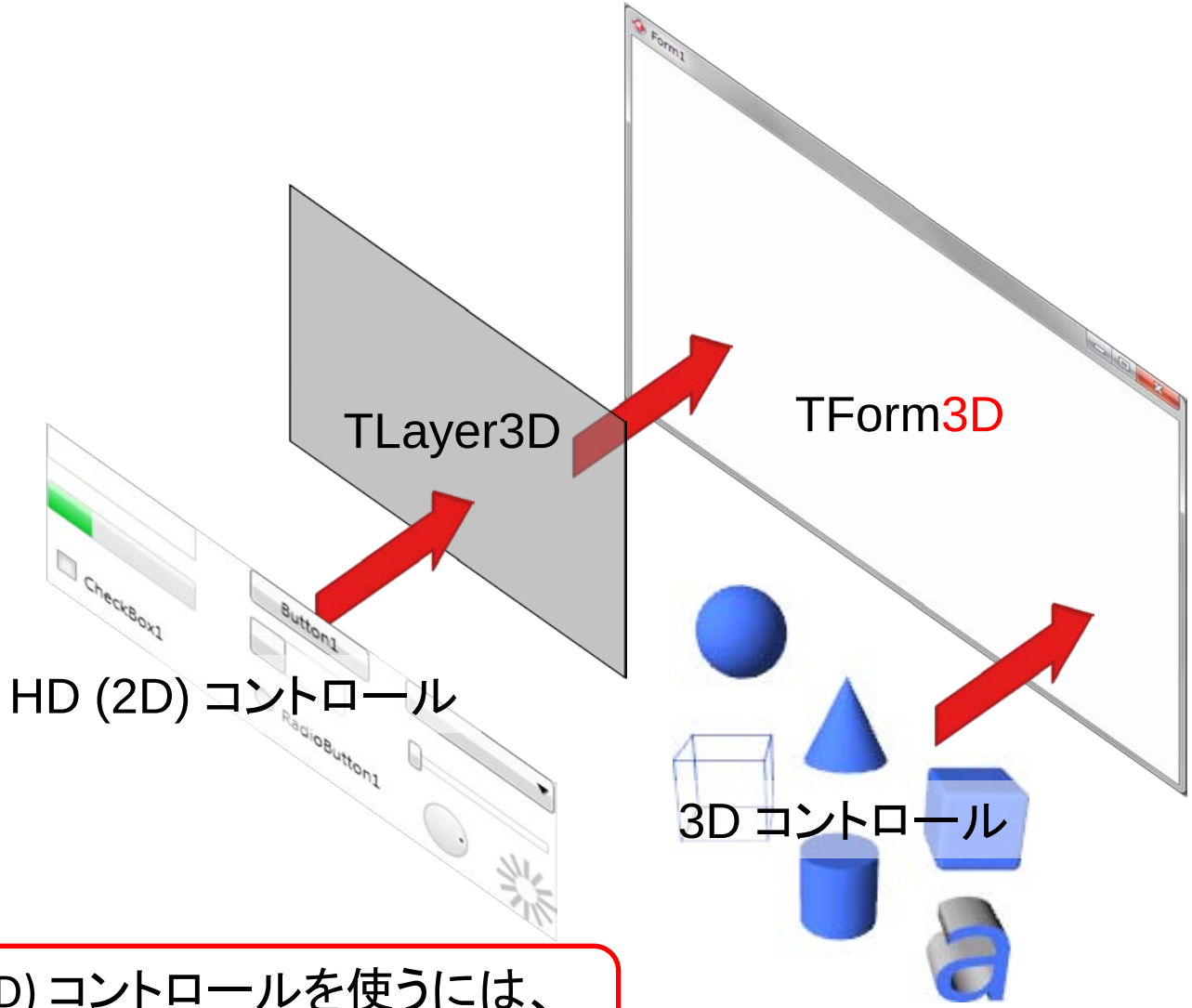
FireMonkey アプリケーション

FireMonkey HD アプリケーション



3D コントロールを使うには、
TViewPort3D を貼る必要がある。

FireMonkey 3D アプリケーション



HD (2D) コントロールを使うには、
TLayer3D を貼る必要がある。

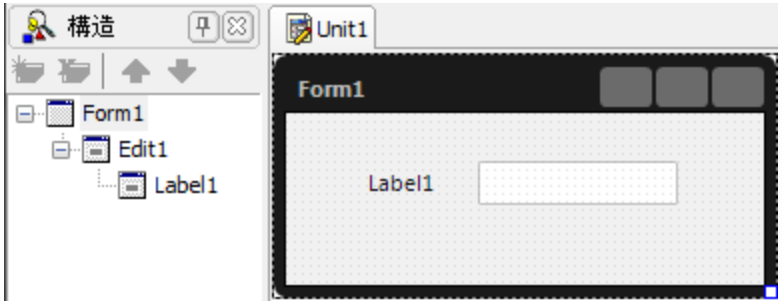
HD (2D) コントロールと 3D コントロール

HD (2D) コントロール	3D コントロール
Additional	3D Layers
Colors	3D Scene
Common Controls	3D Shapes
Dialogs	
Effects	
Grids	
Layouts	
Shapes	
Standard	
System	
ViewPorts	

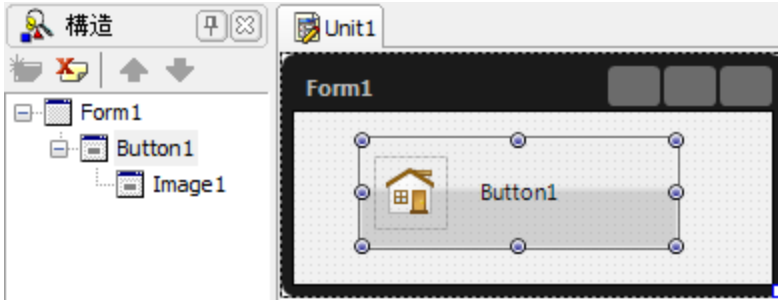
※共通 / サードパーティ製コンポーネントを除く。

親子関係

FireMokey のコントロールは何でも親子関係にできる。



TEdit の子として TLabel を配置すると、VCL の TLabeledEdit 同等のものが作れる。

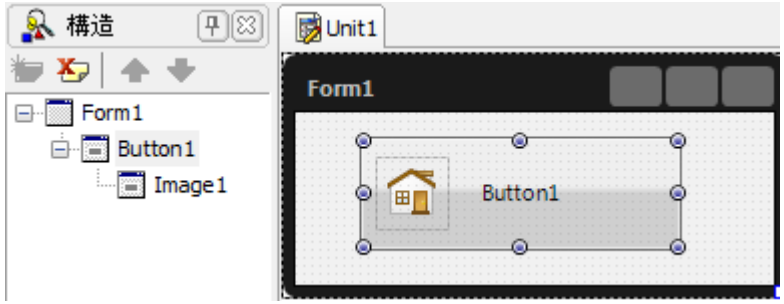


TButton の子として TImage を配置すると、VCL の TBitBtn 同等のものが作れる。

- 親コントロールを移動すると、子コントロールも一緒に移動する。
- 親コントロールの ClipChildren プロパティを True にすると、配置された子コントロールは親コントロールの外に配置できなくなる。
- 親子関係は [構造] ペインで D&D する事によっても変更できる。

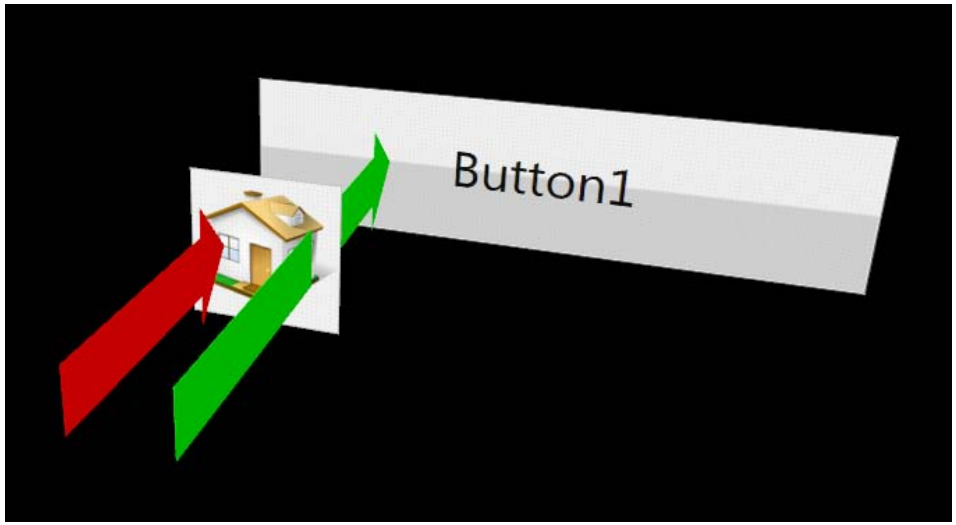
HitTest プロパティ

HitTest プロパティはマウス押下を処理するかどうかを指定する。



先程の例だと、TImage をマウスで押下しても、ButtonClick 処理は発生しない。

HitTest プロパティが False だと、
マウス押下イベントは透過的に親コントロールに伝えられる。



HitTest のイメージ。

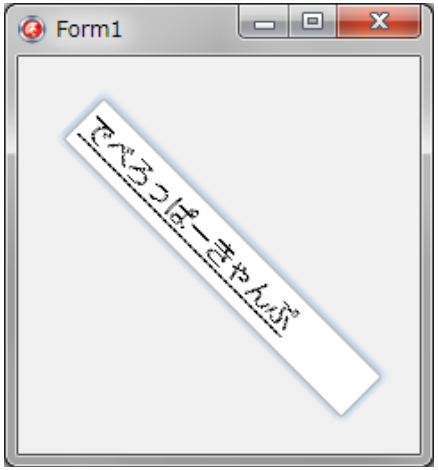
赤の短い矢印が HitTest = True、
緑の長い矢印が HitTest = False。

スケール

- VCL で ChangeScale() を使った場合、任意の倍数を指定しても、コントロールのサイズがフォントに依存する（またはその逆）事があるため、思ったようなスケール変更ができない事があった。
- FireMonkey のコントロールはそもそも独自描画されているため、スケールを変更してもコントロールサイズ / フォントサイズが破綻しない。
- HD（2D）コントロールの場合、Scale プロパティには X / Y があり、縦 / 横それぞれの縮尺を変更できる。
- 3D コントロールの場合、加えて Z（奥行き）の縮尺を変更できる。
- マウスオーバー時にコントロールのスケールをアニメーション（後述）する事で、視覚効果を狙うことができる。

回転

- ハードウェアによる画面の回転では、一瞬画面が消える上に、
  1～2 秒 (ハードウェア依存) 程度のタイムラグが発生する。



- FireMonkey の回転はコントロール単位で行え、
  入力系のコントロールは回転した状態で入力可能。

- RotationAngle プロパティで時計回りに回転する。

- HD (2D) コントロールの場合、
  回転軸を RotationCenter プロパティで指定する事ができる。

  指定する値は比率で、X=0 / Y=0 の場合にはコントロールの左上、
  X=0.5 / Y=0.5 の場合にはコントロールの中央、
  X=1 / Y=1 の場合にはコントロールの右下が回転軸となる。

※Update 4 から 3D コントロールにも
RotationCenter プロパティ が追加されている。

アニメーション

フィルムアイコンのあるプロパティには、オブジェクトインスペクタから、
各種アニメーションを設定できる。

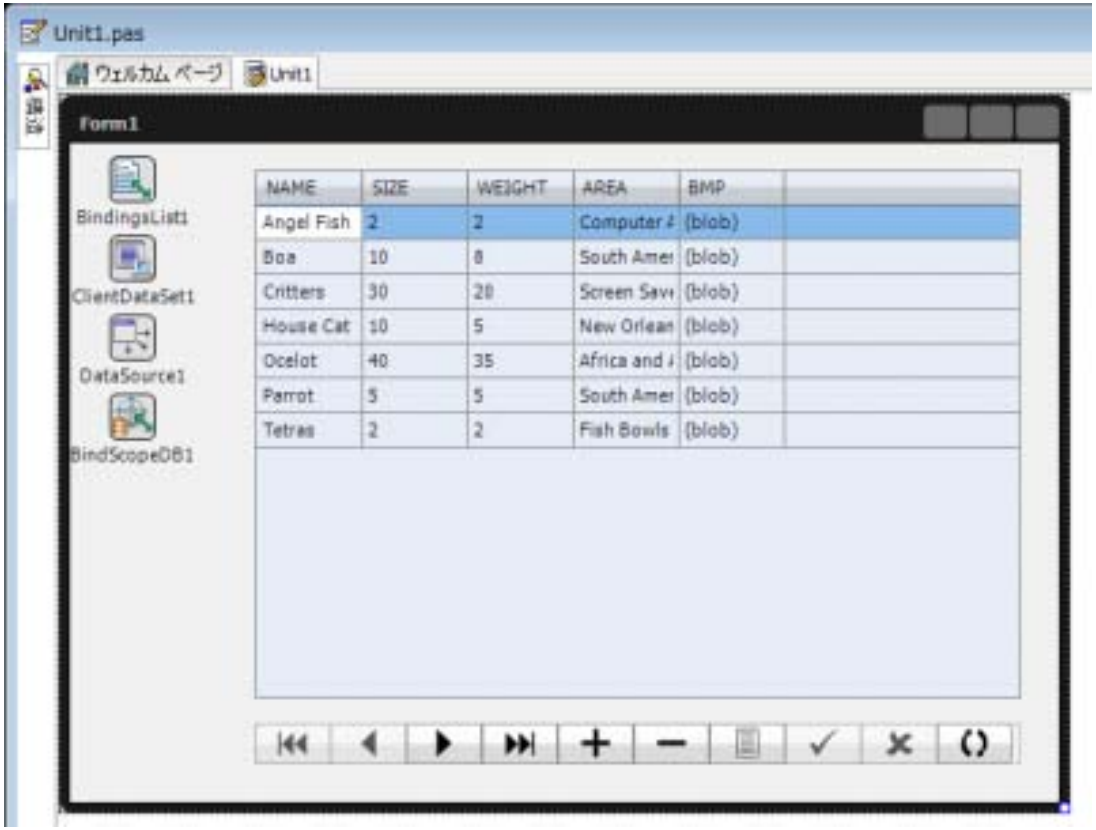
フィルムアイコンがないプロパティでも、型さえ合っていれば
アニメーションが使える。

1. [Animations] にあるコントロールを子コントロールとして貼りつける。
2. PropertyName プロパティでアニメーションさせたいプロパティを設定。

[FireMonkey のアニメーション効果 (DocWiki)]
http://docwiki.embarcadero.com/RADStudio/ja/FireMonkey_のアニメーション効果

LiveBinding (1)

- データベースとバインディングする事によって TDBGrid のようなものを作ったりできる。



- ...が、それ以外のバインディングに関しては？

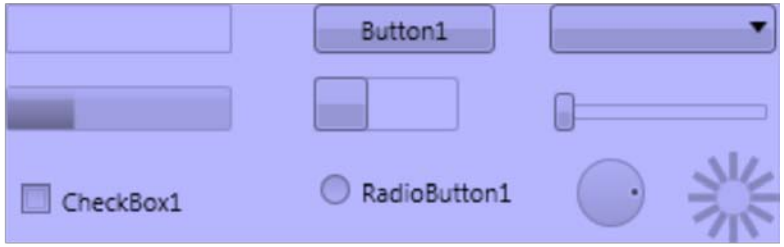
LiveBinding (2)

TBindList

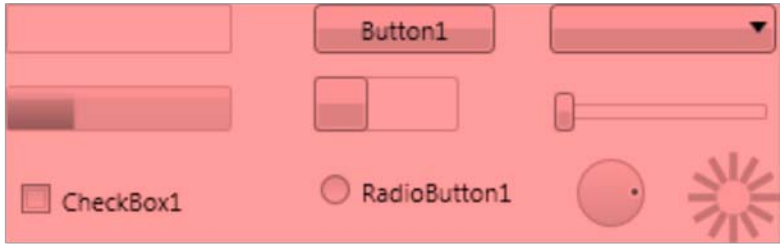
サンプルフォルダ中の LiveBindings¥bindexpression¥fmx にある SynchControlsSampleProject が参考になる。
(双方向バインディング等のサンプルもある)

変更通知
(Notify)

```
procedure TForm1.Edit1Change(Sender: TObject);
begin
  BindingsList1.Notify(Sender, '');
end;
```

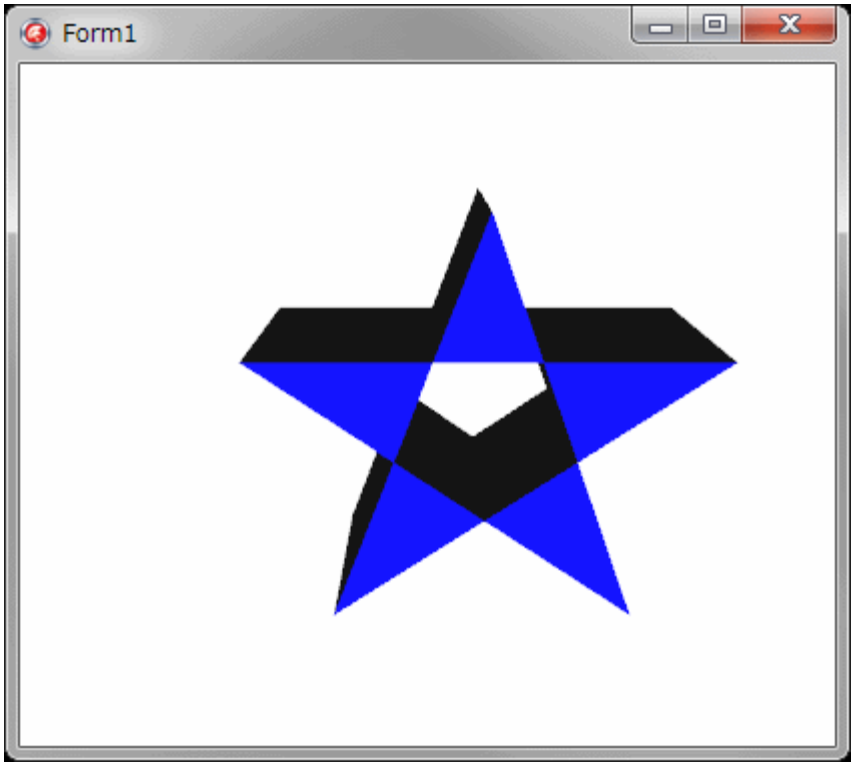


バインディングソース



バインディングターゲット

パス (その1)




```
M 5,20 L 35,20 L 10,40
L 20,10 L 30,40 Z
```

※画像は TPath3D のものです。

- TPathLabel [Standard]
- TPath [Shapes]
- TPath3D [3D Shapes]

で、共通するパス。

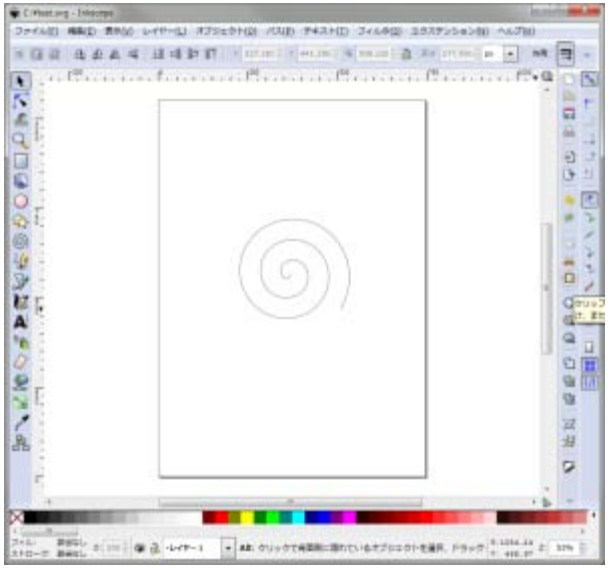
- M が MoveTo (絶対値) X, Y
- L が LineTo (絶対値) X, Y
- Z が描画終了

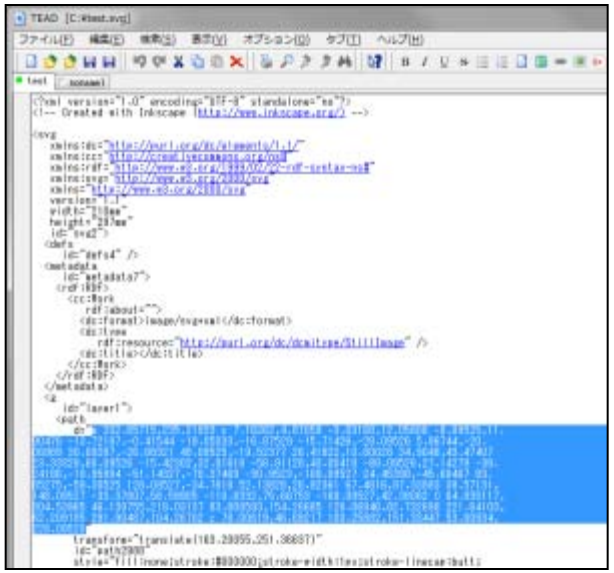
詳細は以下を参照の事。

[パス マークアップ構文 (MSDN)]
http://msdn.microsoft.com/ja-jp/library/ms752293.aspx

パス (その2)

- 「簡単なパスはいいけど、複雑なパスはメンドイよ！」
  では、GUI でパスデータを作ってしまいましょう。





1. InkScape (http://inkscape.org/index.php?lang=ja) でパスデータを作り、SVG (プレーン) 形式で保存する。

2. SVG をテキストエディタで開き、パスデータ部分をコピーする。

3. RAD Studio のパスデザイナにパスデータを貼り付ける。

embarcadero
Developer Camp

FireMonkey HD (2D)
コントロールの概要

要注意コンポーネント (1)

- VCL と同名のコンポーネントには面食らってしまうかも？

-  TComboBox [Standard]
  キーボードからの文字入力を行う必要がある場合には TComboEdit を使う。Items プロパティは public なので、オブジェクトインスペクタから設定する事はできない。以前は Items プロパティそのものがなかったらしい。

-  TMemo [Standard]
  Update 3 以前だと、Lines プロパティは public なので、
  オブジェクトインスペクタから設定する事はできない。

-  TPanel [Standard]
  Caption プロパティはない。TLabel を子コントロールとして貼りつければ VCL 版と同等の事ができるが、レイアウトのためだけに使うのであれば、
   TLayout の方が使い勝手がいい。

要注意コンポーネント (2)

-  TImage [Shapes]
  VCL の TImage と同等。画像はアスペクト比を保ったまま、
  自動的に拡大/縮小される。画像はコントロールの中央に配置される。

-  TImageControl [Standard]
  TImage と同等だが、背景は透過しない。フォーカスを持つ。

-  TImageViewer [Additional]
  背景は透過しない。自動的に拡大/縮小しないが、
  BestFit() メソッドを呼ぶと、画像サイズをアスペクト比を保ったままコントロールサイズに合わせる事が可能。コントロールサイズよりも画像の方が大きい場合、マウス / タッチ操作によるパニングが可能。また、マウスホイールで拡大/縮小も可能。

-  TPaintBox [Shapes]
  VCL の TPaintBox と同等。

要注意コンポーネント (3)

-  TToolBar [Standard]
  フォーム上部に張り付く TPanel のようなもの。ボタンを作る機能はない。
  TSpeedButton 等を子コントロールとして配置する事で VCL 版と同等の機能を実現可能。

-  TStatusBar [Standard]
  フォーム下部に張り付く TPanel のようなもの。
  パネルを作る機能はない。SimpleText プロパティもない。
  TLabel 等を子コントロールとして配置する事で VCL 版と同等の機能を実現可能。

-  TSpeedButton [Standard]
  Gryph プロパティを持たないので画像付きボタンにはできない。
  TImage を子コントロールとして配置する事で VCL 版と同等の機能を実現可能。

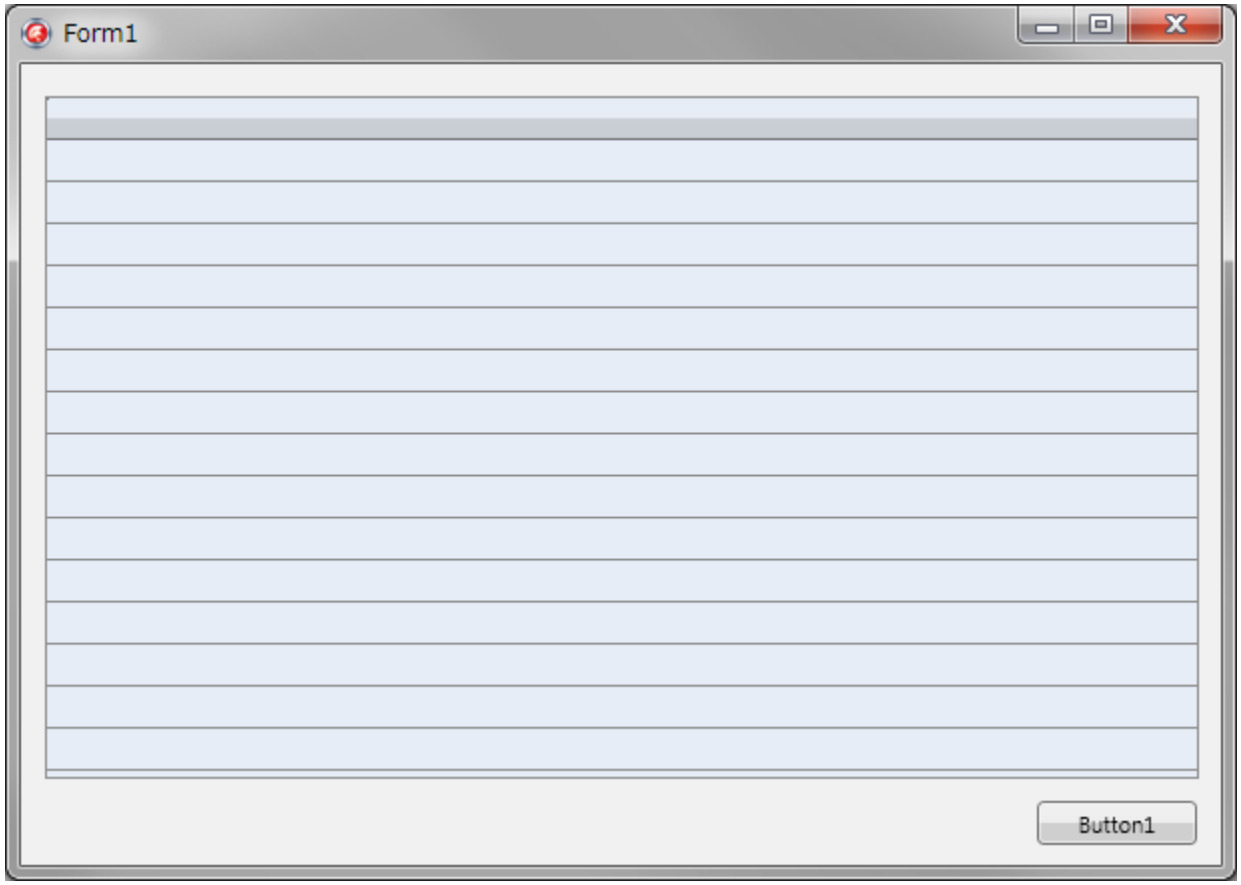
要注意コンポーネント (4)

-  TColorBox [Colos]
  同名の VCL と同等なのは  TColorComboBox の方。
  用途が全く違うので、VCL の TColorBox なのか FMX の TColorBox なのか、ハッキリ区別した方がいい。

-  TStringGrid [Grids]
  かなり作法が違う。VCL 版の ColCount プロパティはなく、
  代わりに ColumnCount プロパティがあるが、ReadOnly なプロパティであり、これを用いてカラムを増減させる事はできない。

TStringGrid (1)

- TStringGrid の カラムをコードで生成するにはどうすれば？



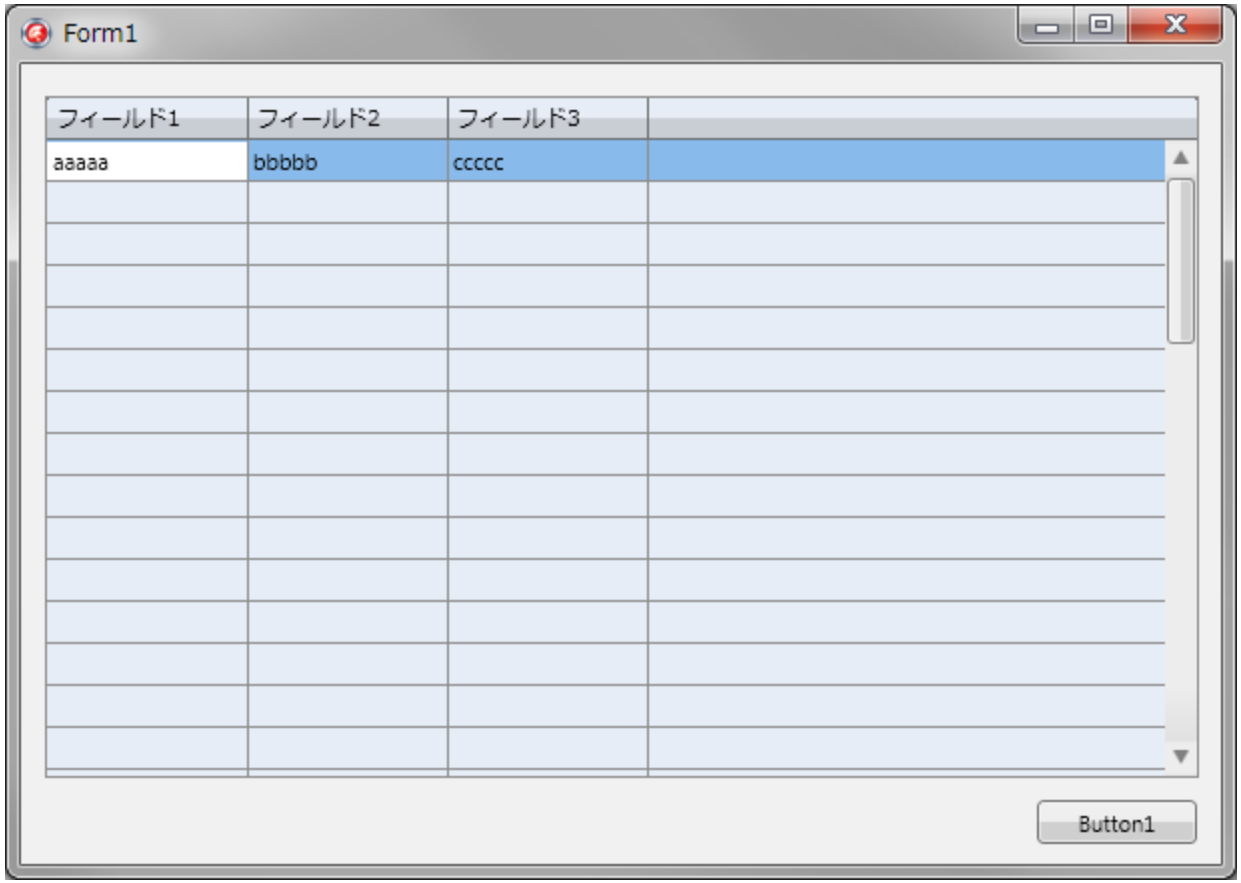
TStringGrid (2)

以下のようなコードを記述する。

```
// カラムを 3 つ作成
StringGrid1.AddObject(TStringColumn.Create(StringGrid1));
StringGrid1.AddObject(TStringColumn.Create(StringGrid1));
StringGrid1.AddObject(TStringColumn.Create(StringGrid1));

// カラムヘッダを設定
StringGrid1.Columns[0].Header := 'フィールド1';
StringGrid1.Columns[1].Header := 'フィールド2';
StringGrid1.Columns[2].Header := 'フィールド3';

// データは 50 行
StringGrid1.RowCount := 50;

// データ (1 行目)
StringGrid1.Cells[0, 0] := 'aaaaa';
StringGrid1.Cells[1, 0] := 'bbbbb';
StringGrid1.Cells[2, 0] := 'ccccc';
```

TStringGrid (3)

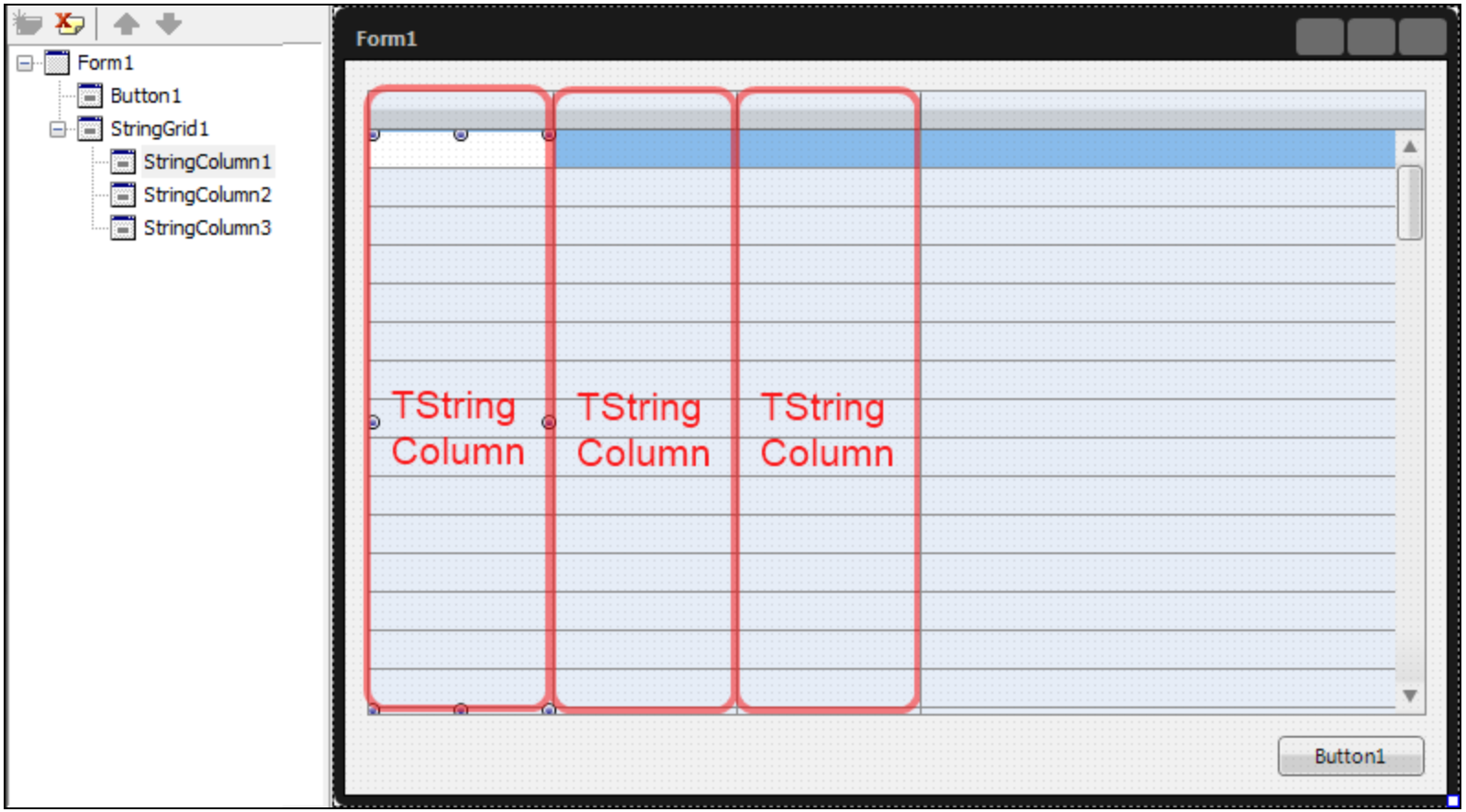
できました (^o^)/



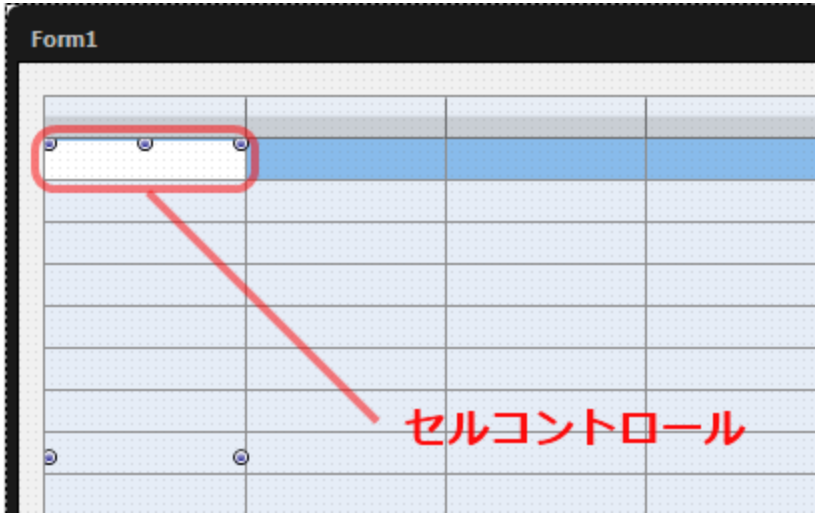
カラムに右寄せとかを設定したいなぁ…。

TStringGrid (4)

TStringGrid のカラムは TColumn から派生した TStringColumn で構成されている。これがカラムコントロール。



TStringGrid (5)



セルコントロールは VCL の TStringGrid で言うインプレースエディタ。

TStringGrid のカラムコントロールである TStringColumn が生成するセルコントロールは TTextCell。

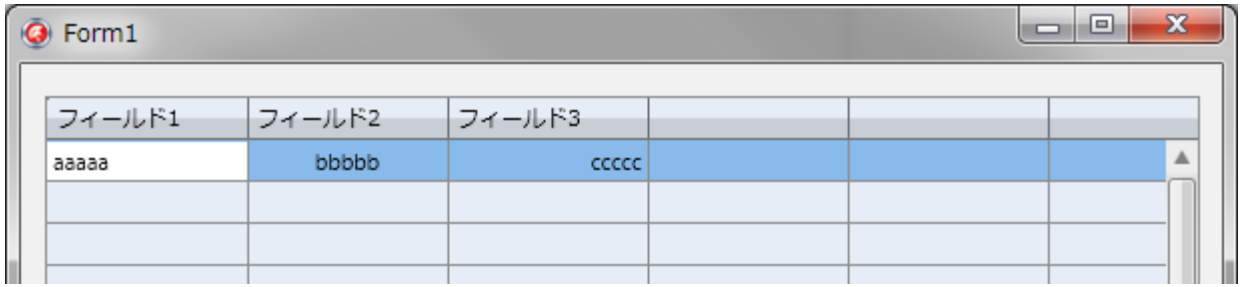
TTextCell = class(TEdit) なので....

```
procedure TForm1.Button1Click(Sender: TObject);
begin
  TEdit(StringGrid1.Columns[0].CellControlByRow(0)).TextAlign := TTextAlign.taLeading;
  TEdit(StringGrid1.Columns[1].CellControlByRow(0)).TextAlign := TTextAlign.taCenter;
  TEdit(StringGrid1.Columns[2].CellControlByRow(0)).TextAlign := TTextAlign.taTrailing;
end;
```

こんなコードで左/中央/右寄せできるのでは？

TStringGrid (6)

予想通りできました。



…が、実はこの方法には問題がある。

セルコントロールは表示されている範囲のものしか生成されない。
つまり、見えていないセルの TextAlign は設定できない。

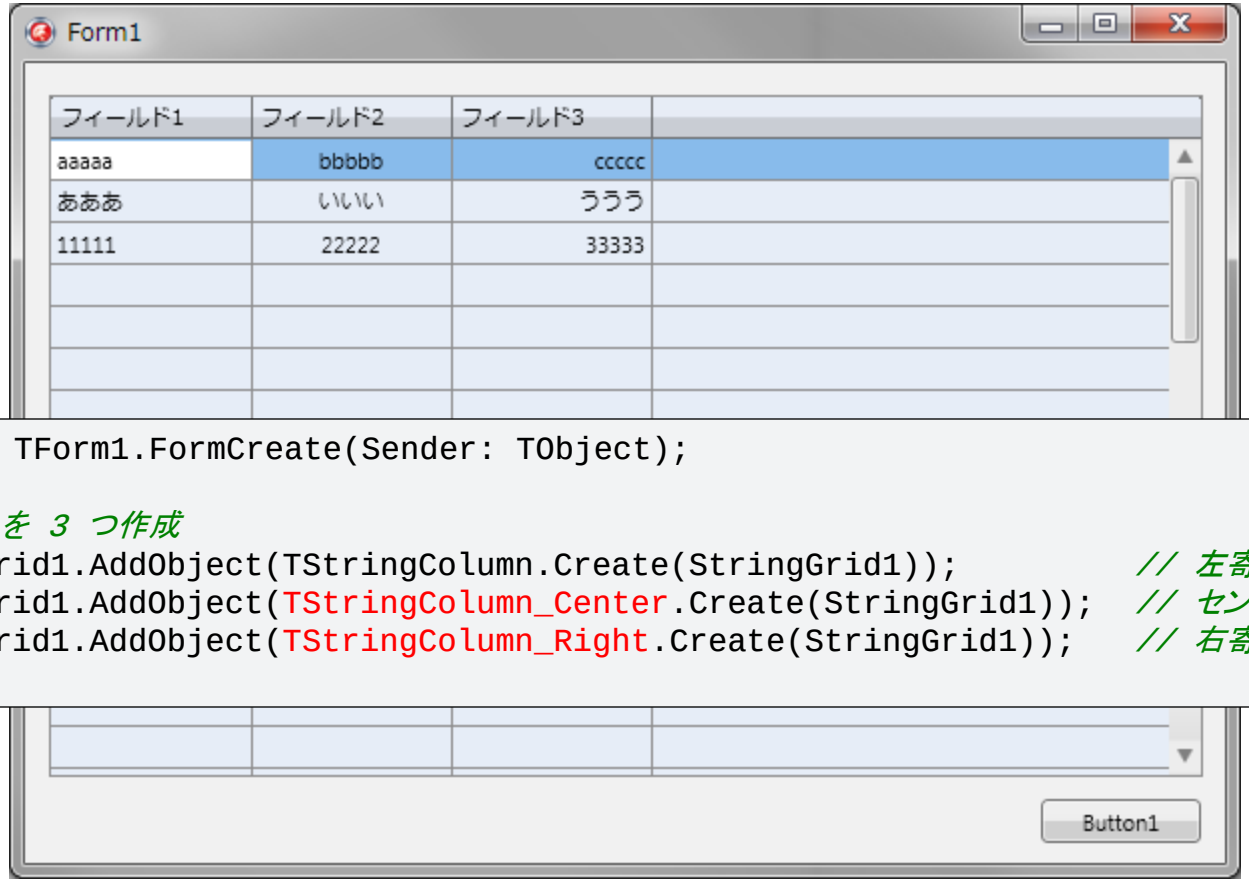
Assigned() を使ってセルコントロールの存在確認をする方法もあるが、
たかだか左/中央/右寄せのためにそんなにコード書かなくては
ならないのか？

TStringGrid (7)

別途カラムコントロールを作ってやれば解決する。ソースはこれだけ。

```
  { TStringColumn_Center }
  TStringColumn_Center = class(FMX.Grid.TStringColumn)
  protected
    function CreateCellControl: TStyledControl; override;
  end;

 { TStringColumn_Right }
  TStringColumn_Right = class(FMX.Grid.TStringColumn)
  protected
    function CreateCellControl: TStyledControl; override;
  end;
  ...

 { TStringColumn_Center }
function TStringColumn_Center.CreateCellControl: TStyledControl;
begin
  result := inherited;
  TEdit(result).TextAlign := TTextAlign.taCenter;
end;

{ TStringColumn_Right }
function TStringColumn_Right.CreateCellControl: TStyledControl;
begin
  result := inherited;
  TEdit(result).TextAlign := TTextAlign.taTrailing;
end;
```

TStringGrid (8)

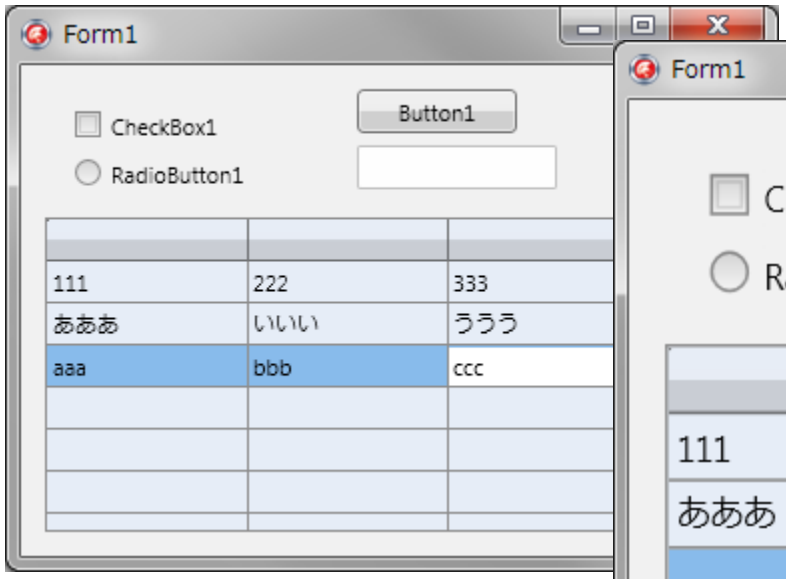


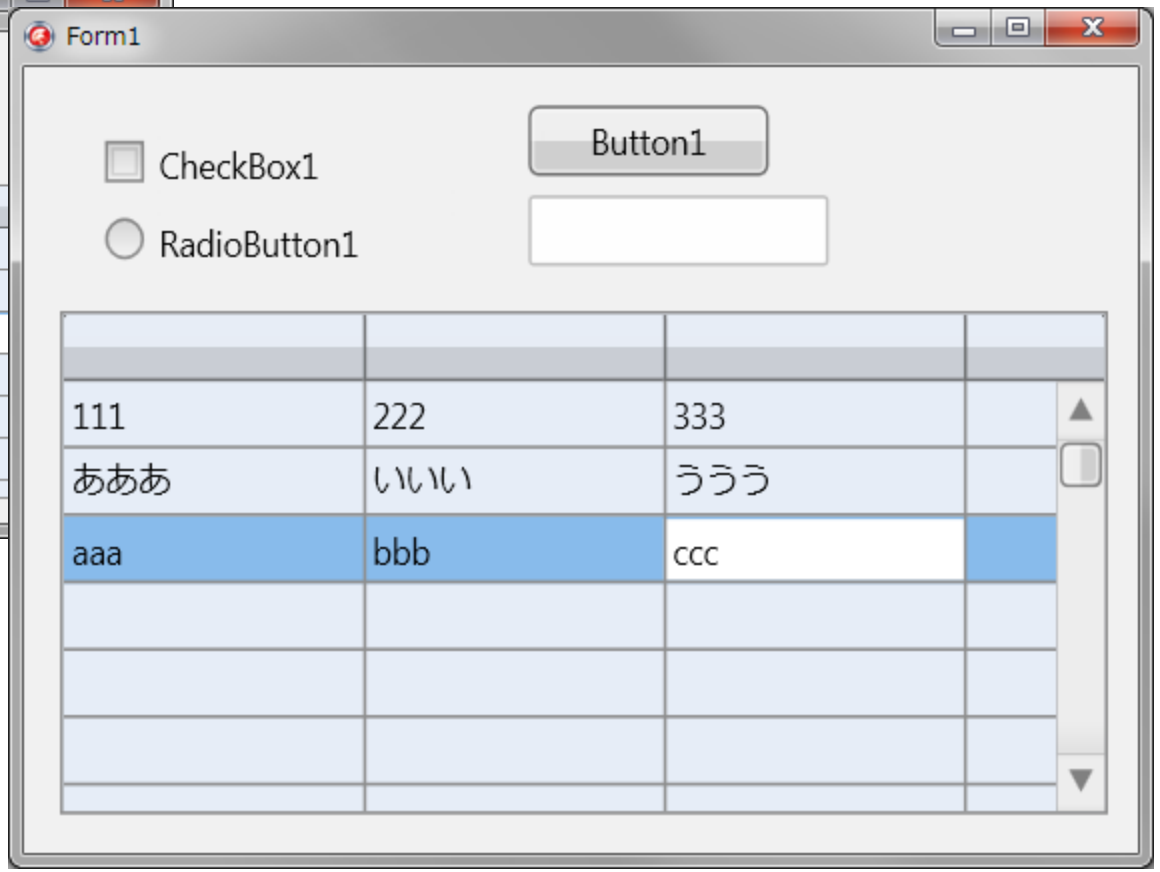

```
procedure TForm1.FormCreate(Sender: TObject);
begin
  // カラムを 3 つ作成
  StringGrid1.AddObject(TStringColumn.Create(StringGrid1));          // 左寄せ
  StringGrid1.AddObject(TStringColumn_Center.Create(StringGrid1));   // センタリング
  StringGrid1.AddObject(TStringColumn_Right.Create(StringGrid1));    // 右寄せ
  ...
```

TGrid も同様の考え方で機能拡張できる。

FireMonkey 固有のコンポーネント (1)

- FireMonkey HD アプリケーションのフォームにはとりあえず TScaledLayout を Align = alFit で貼ってみる。

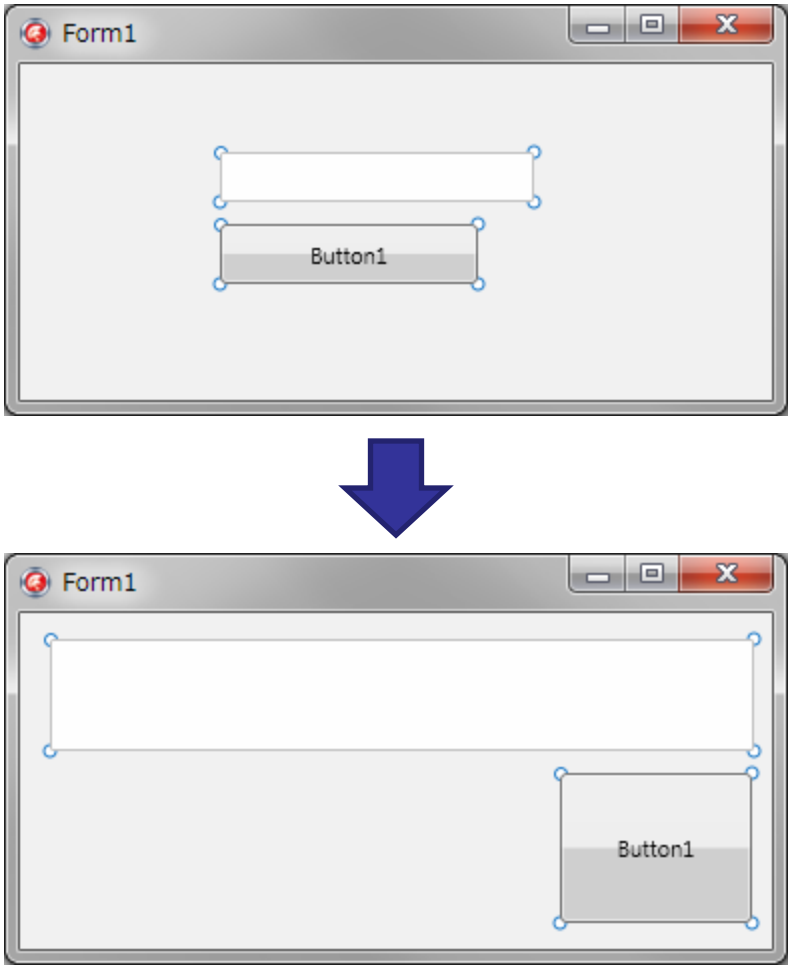




フォームをリサイズすると
コントロールサイズや、
グリッドの幅も追従する。

FireMonkey 固有のコンポーネント (2)

- TSelection を使うと簡単にフォームデザイナが作れる。



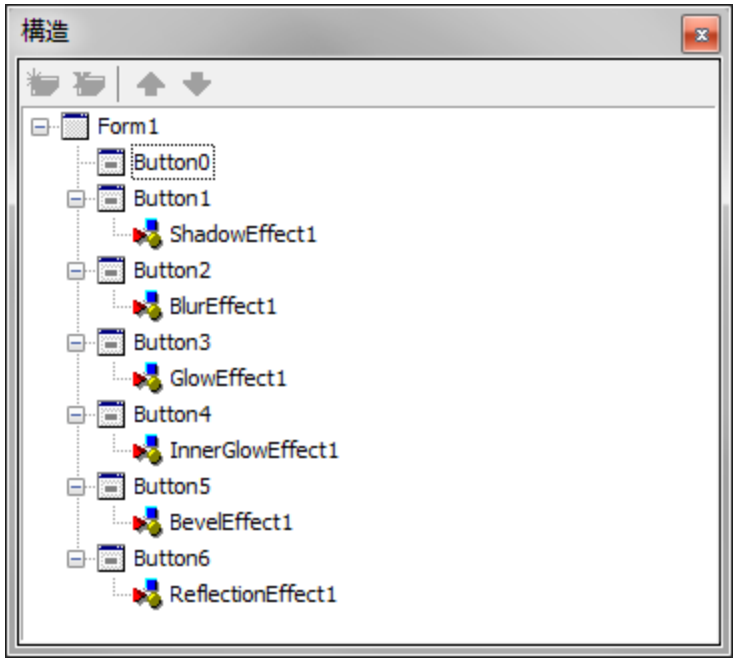
- 子コントロールの
  HitTest プロパティを False に設定。
- 子コントロールの
  Align プロパティを alClient に設定。

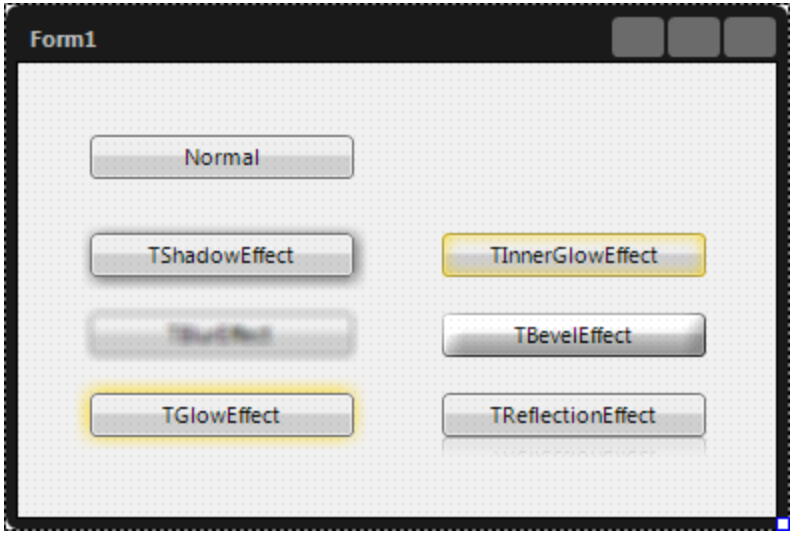
...たったこれだけ。

- TSelection の HideSelection
  プロパティを True に設定すると
  セレクションポイントが非表示になる。
- TSelection の Proportional
  プロパティを True に設定すると
  アスペクト比を保った拡大/縮小が
  可能になる。

FireMonkey 固有のコンポーネント (3)

- [Effects] にあるコントロールは画像コントロールに限らず、
  任意の FMX コントロールに対して適用する事が可能。
  以下は TButton にエフェクトを適用した例。





- 各ボタンの子としてエフェクトコントロールが追加されている。

embarcadero
Developer Camp

FireMonkey 3D
コントロールの概要

[3D Shapes] - その1

-  **TCube**
  立方体 (直方体にもできる)。
  テクスチャは 6 面に貼られる。

-  **TRoundCube**
  角が丸い立方体。テクスチャは前面と背面をメインに貼られる。

-  **TPlane**
  板。テクスチャは前面に貼られる。

-  **TDisk**
  円盤。Updete 4 で追加された。

-  **TGrid3D**
  位置関係を示すスケールとして使う。
  色は LineColor で指定。
  マス目のサイズは Frequency で指定。
  Marks に指定したサイズの倍数の時にグリッドの線は細線になる。

-  **TStrokeCube**
  ワイヤーフレームの立方体。
  色は Material.Diffuse で指定。

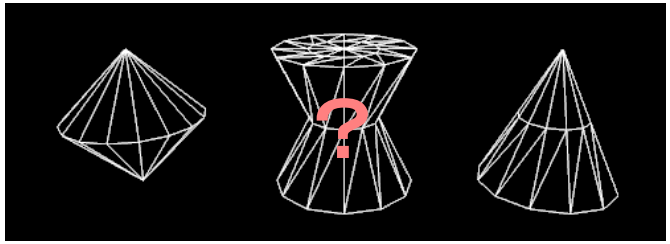
※TGrid3D 以外にはテクスチャを貼る事ができる。
但し、TStrokeCube のテクスチャは無視される。

[3D Shapes] - その2

- **TSphere**
  球体。テクスチャは球面に貼られる。

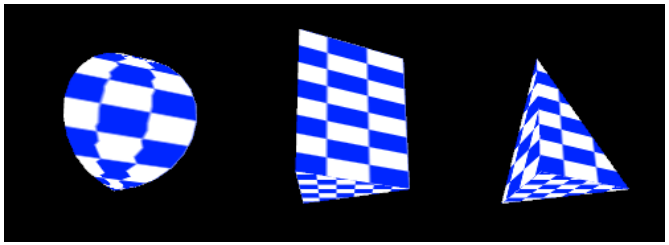
縦横比 1 : 2 のテクスチャが
使えます。

- **TCylinder**
  円柱。三角柱や、六角柱も作れる。

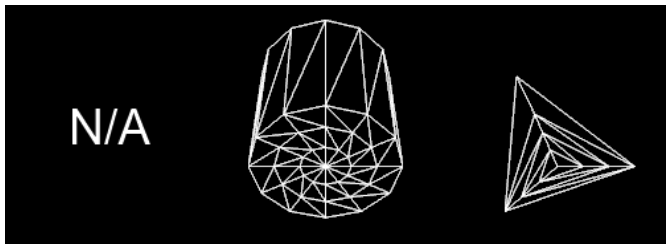


SubdivisionHeight は高さの
分割数を指定する (分割数 = 2)。

- **TCone**
  円錐。三角錐や、四角錐も作れる。



SubdivisionAxes は軸の
分割数を指定する (分割数 = 3) 。



SubdivisionCap はキャップ(底面)の
分割数を指定する (分割数 = 3) 。

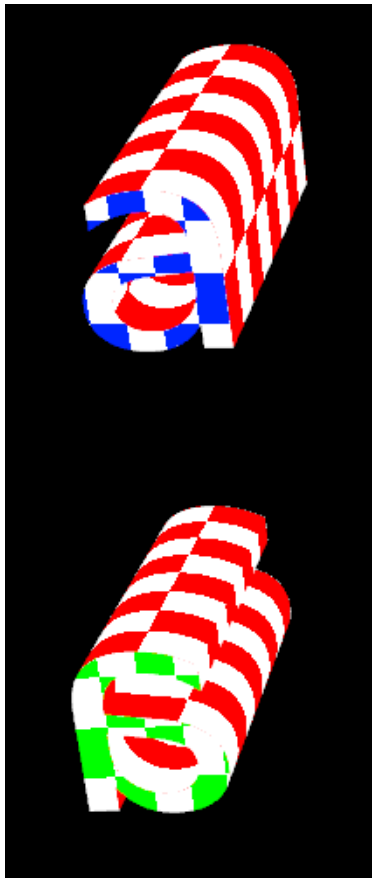
[3D Shapes] - その3

- **TEllipse3D**
  円を押し出した形状 (円柱) の 3D オブジェクト。

- **TRectangle3D**
  長方形を押し出した形状 (四角柱) の
  3D オブジェクト。

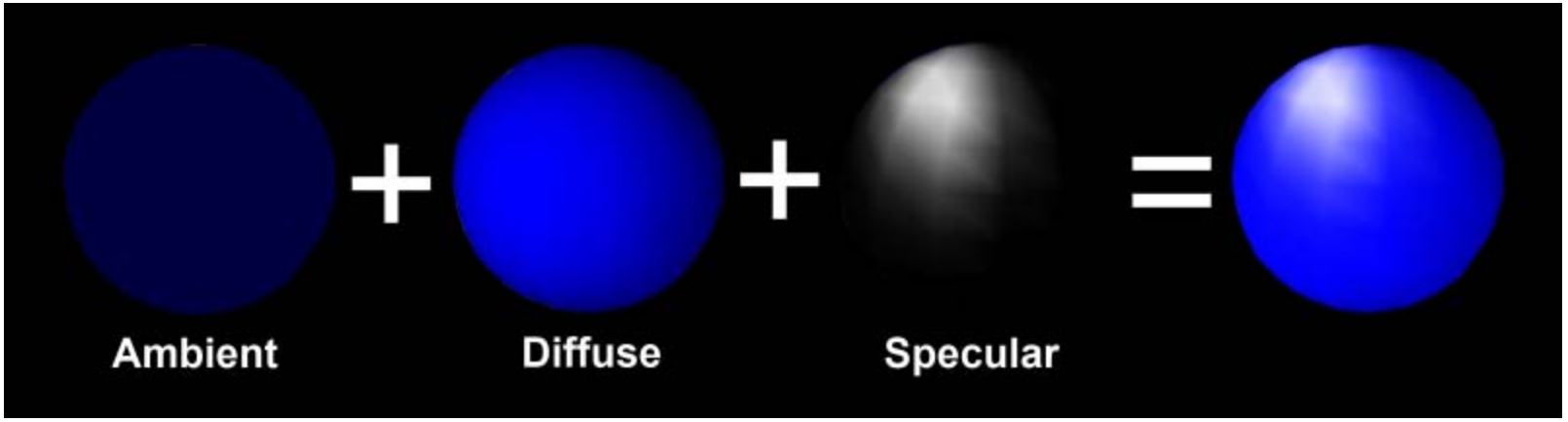
- **TPath3D**
  パスで描かれた図形を押し出した形状の
  3D オブジェクトを作り出す事ができる。

- **TText3D**
  文字列を押し出した形状の 3D オブジェクト。



※前面 / 背面 / 側面 (シャフト) にテクスチャを貼る事ができる。

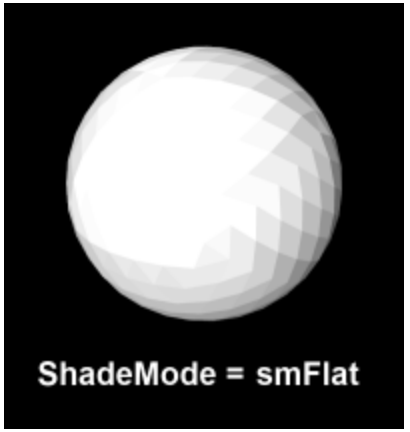
[3D Shapes] - その4 (TMaterial)



反射色なので、ライトが必要。Update 3 以前では　Specular は効果がない。

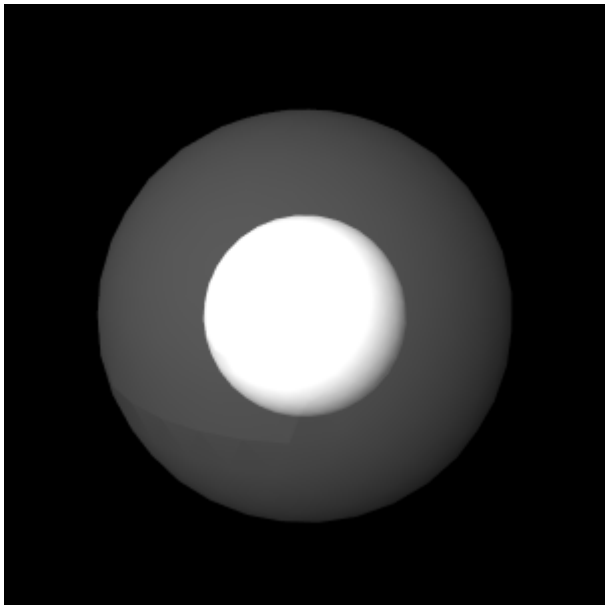
鏡面反射の強さは Shinness で指定する。

- Emissive
  (自己) 発光色を指定する。
- Modulation
  - tmModulate
    シェイプの色とテクスチャの色をミックスする。
  - tmReplace
    テクスチャの色のみが使われる。
- Texture
  テクスチャを指定する。



[3D Shapes] - その5 (半透明の表現)

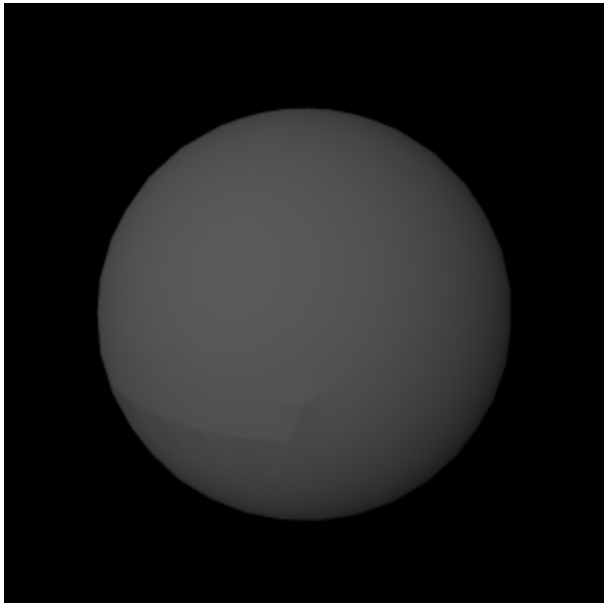
半透明のシェイプを作るには、Diffuse にアルファカラーを設定する。
以下は 2 重の Sphere で、外側の Sphere を半透明にしてみた所。



- Ambient: #FF404040
- Diffuse: #00404040
- 作成順：内側→外側

- 半透明のオブジェクトは後で描画するようにしなくてはならない。
- 複数の半透明オブジェクトがあるのなら、遠くのものから描画しなくてはならない。

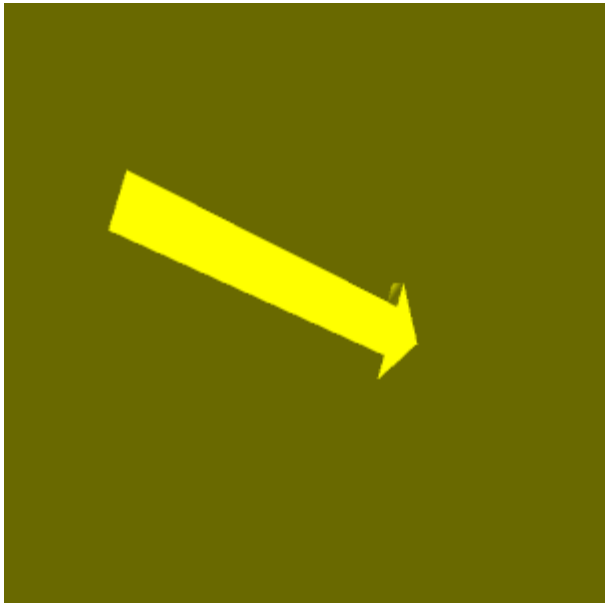
- Ambient: #FF404040
- Diffuse: #00404040
- 作成順：外側→内側



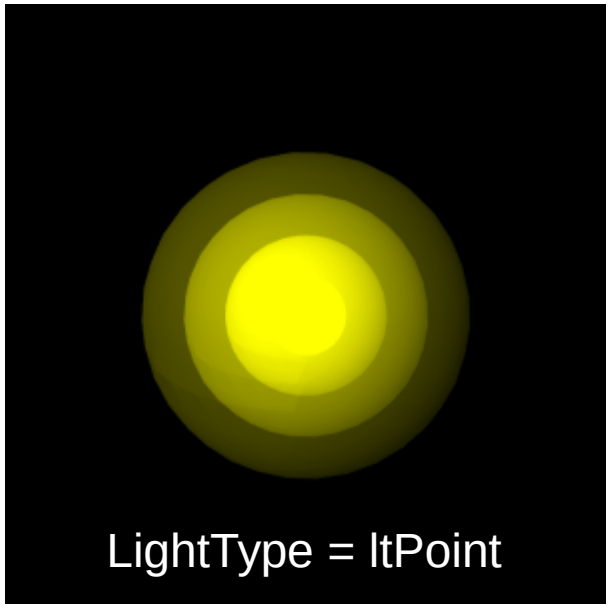
※Update 4 からは Opacity でも半透明の指定が可能。

[3D Scene] - その1 (ライト)

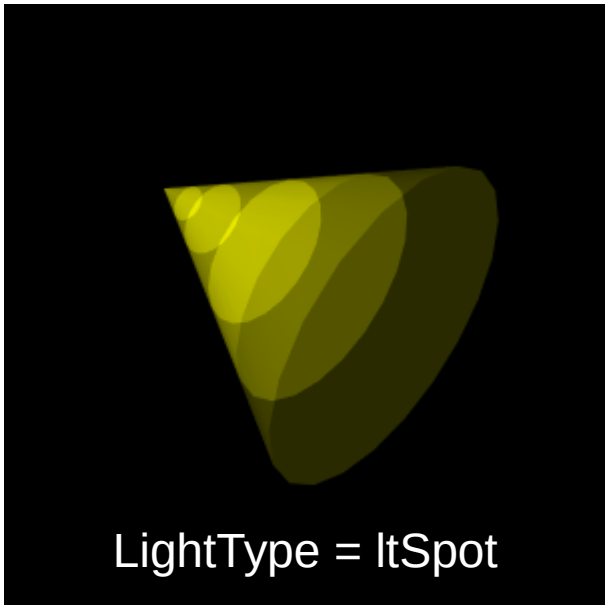
-  TLight は外部光。3D Shape の反射光のプロパティに影響を及ぼす。



並行光源。どこに配置しても光の強さは変化しない。
光を当てる方向はライトの向きで変わる。



点光源。無指向性なので放射状に照らす。ライトからの距離で光の強さが変化。
光を当てる方向はライトの位置で変わる。



スポット光源。指向性があるため、円錐状に照らす。
ライトからの距離で光の強さが変化。
光を当てる方向はライトの向きと位置で変わる。

※Update 4 からは 反射色の指定が可能。

[3D Scene] - その2 (カメラ)

-  TCamera は視点を切り替えるのに使う。

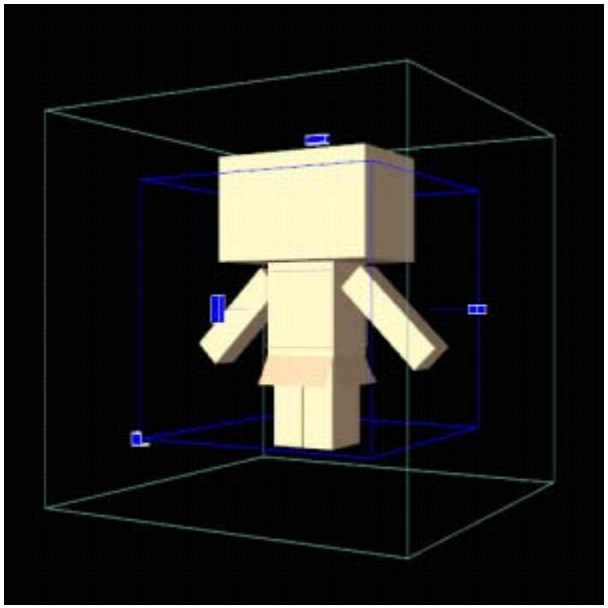
- TForm3D または TViewPort3D の Camera プロパティに TCamera を設定し、UsingDesignCamera を False にする事でカメラの視点に変更できる。

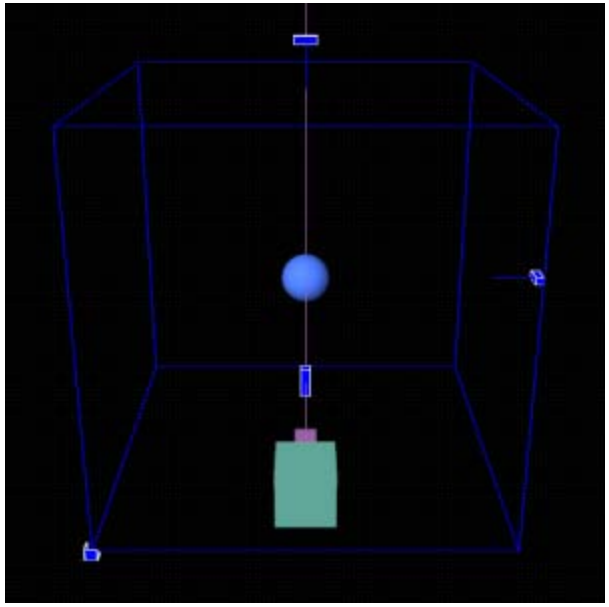
- 実行時にコードでカメラを切り替えた場合には、
Repaint しなくてはならない。

- 3D シェイプの内側にカメラを設置する事も可能。
但し、3D シェイプの TwoSide プロパティを True に設定しないと
内側にテクスチャは貼られない。
また、貼られたテクスチャは鏡像になる。

[3D Scene] - その3 (ダミーオブジェクト)

- TDummy は、見えないダミーのオブジェクト。





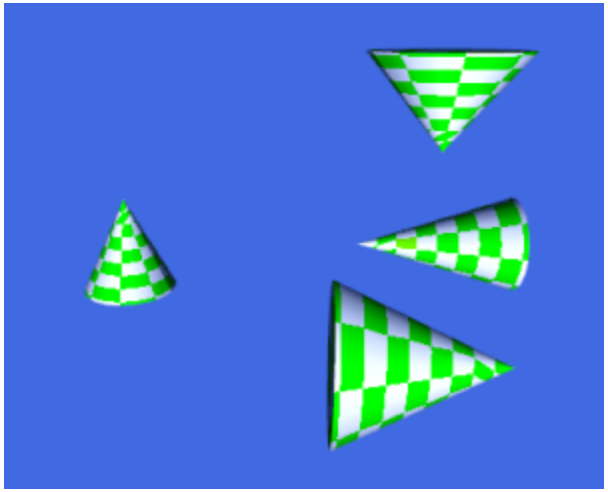
複数のオブジェクトを TDummy の子として配置すれば、TDummy を移動 / 回転するだけで、すべてのオブジェクトが "配置された状態で" 移動 / 回転する。

カメラを子としてオフセット配置し、
TDummy を回転させれば、
オブジェクトを周回するカメラになる。
カメラを単純に座標計算で移動しても、
オブジェクトを捉えない事に注意。

※落書きダメ、ゼッタイ。

[3D Scene] - その4 (代理オブジェクト)

-  TProxyObject はその名の通り代理オブジェクト。



左の TCone が実体で、右の 3 つは代理オブジェクト。
TCone にはテクスチャが貼ってあるが、
これを 4 つの TCone で表現しようとすると
4 倍のテクスチャが必要になる。

テクスチャを貼った TCube を一つ用意し、
それを参照する TProxyObject を迷路状に
敷き詰めれば、3D ダンジョンの出来上がり。
ファン〇シースター（初代）みたいなのが作れる。

スー〇ーマ〇オのブロックでもいいけれど。

- 困った事に、TProxyObject が実体を指す事がある (参照ではなくて)。
- 誰もが真っ先に思いつくであろう TModel3D の代理オブジェクトは
  XE2 Update 4 の時点では不可能 (QC#103139)。

TModel3D [3D Shapes] - その1

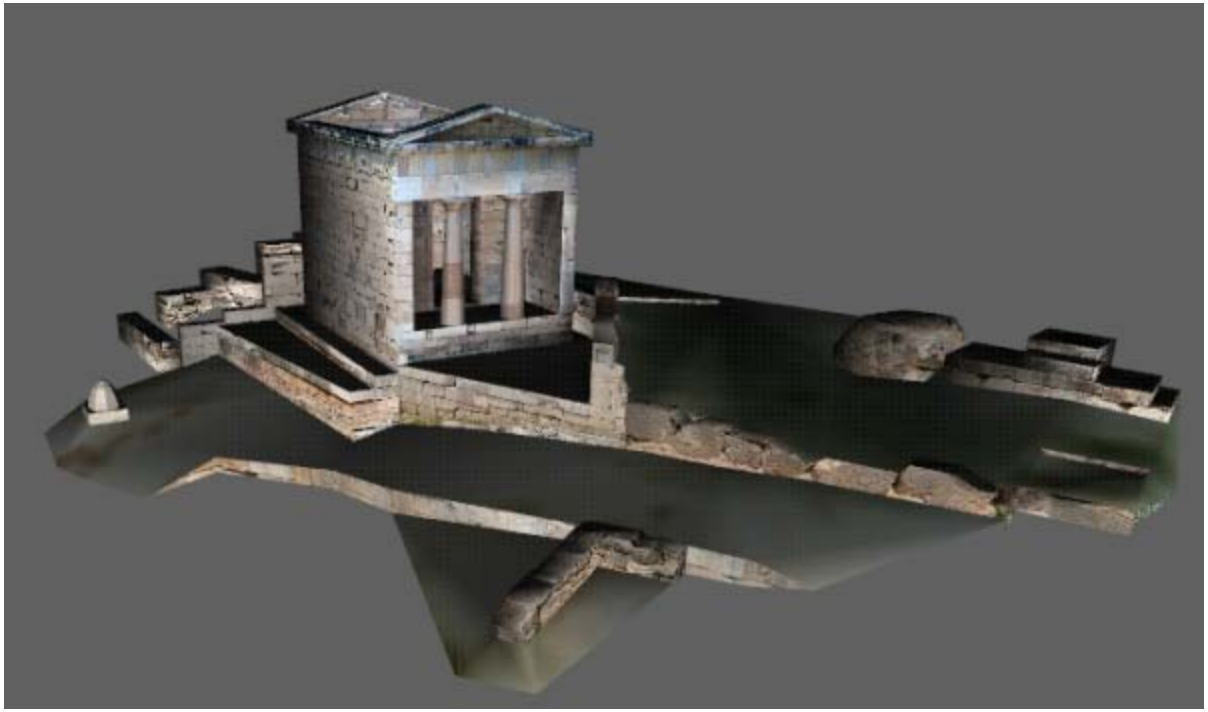
-  TModel3D には、各種モデルデータが読み込める。

- TMeshCollection には以下の形式のモデルデータが読み込める。
  - *.ase (Ascii Scene Export)
  - *.dae (COLLADA)
  - *.obj (Wavefront)

- LoadFromFile() でモデルデータを読み込む場合には
  モデル形式に応じて以下を **uses** する必要がある。
  - FMX.ASE.Importer
  - FMX.DAE.Importer
  - FMX.OBJ.Importer

- 検証した結果、Google SketchUp からの DAE エクスポートデータだと
  比較的問題なく取り込める事を確認した。

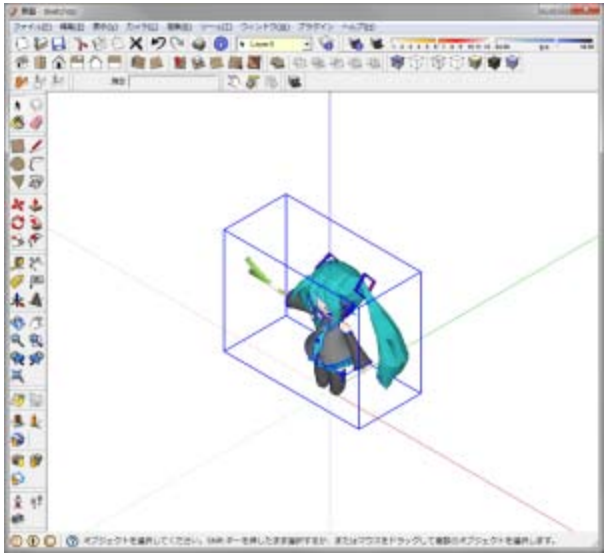
TModel3D [3D Shapes] - その2

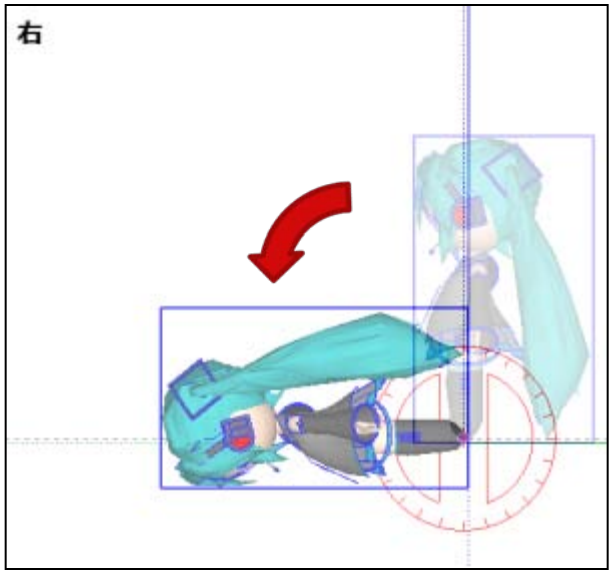


Delphi って事で Google 3D ギャラリーの神殿データを取り込んでみました。
[Θησαυρός των Αθηναίων, Δελφοί - Athenian Treasury, Delphi]
http://sketchup.google.com/3dwarehouse/details?mid=63e5f1f194d769c3623ac7e0d52d2192&prevstart=0

TModel3D [3D Shapes] - その3

- Google SketchUp に正常に取り込めたモデルデータは TModel3D にもほぼ確実に取り込める。
  1. Google SketchUp にモデルデータを読み込ませる。
  2. 表示を右側面あるいは左側面に切り替えて前に 90° 倒す。
  3. [ファイル | エクスポート | 3D モデル...] で *.dae でエクスポート。
  4. TModel3D. MeshCollection に読み込む。

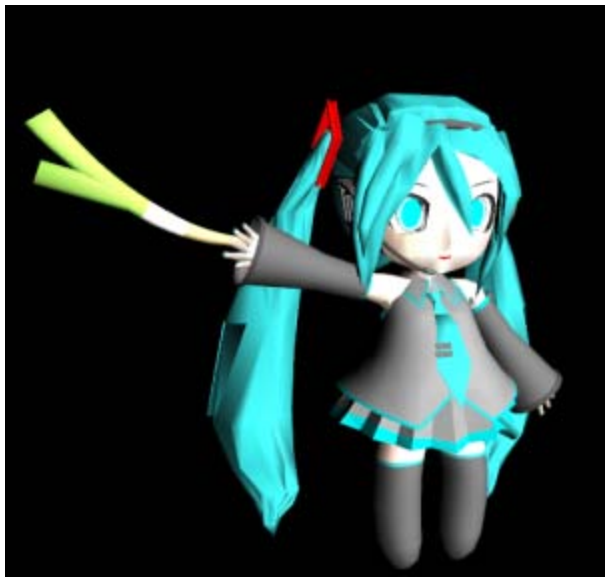




※DAE をインポートした場合には、
テクスチャ画像のフォーマットに注意。

TModel3D [3D Shapes] - その4

- TModel3D は  TMesh の集合体で、ボーン情報等は持っていない。
- つまり、読み込んだモデルそのものを簡単に形状変更する機能はない。
- TModel3D のモデルそのものをアニメーションする事はできない。



「ん？でもアナタ確かネギ振らせてましたよね？」

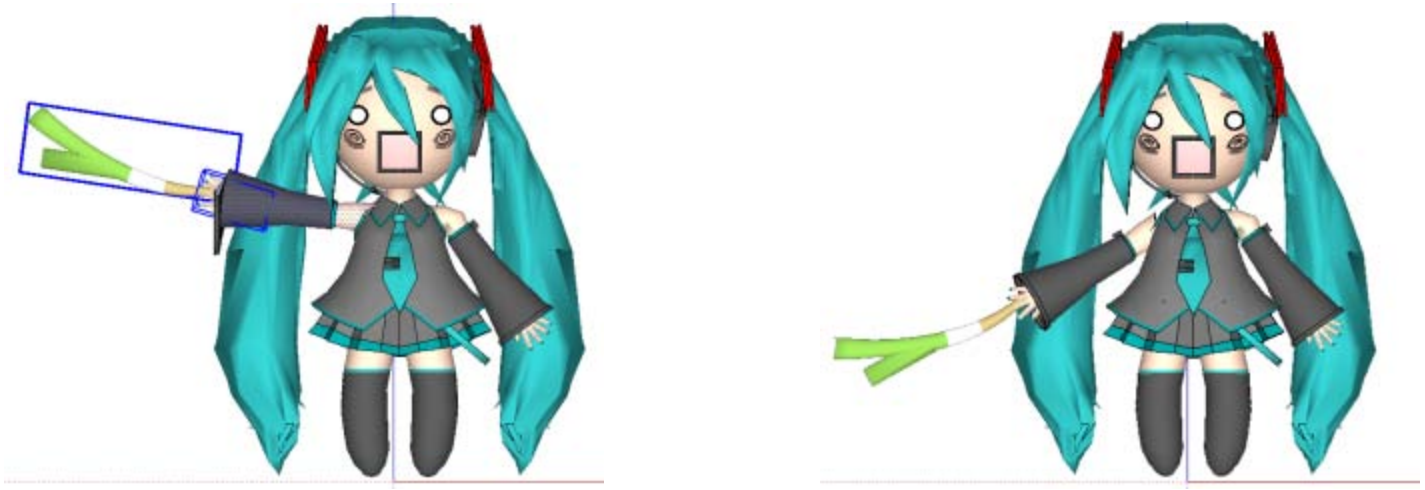
[FireMonkey [3D Shapes] Demo]
http://www.youtube.com/watch?v=UHmYKVz_oME

振ってましたね、ネギ。しかも回転しながら。

TModel3D [3D Shapes] - その5

「複数のモデルを交互に表示 / 非表示しているの？」

場合によってはそれもアリかもしれませんが、
パタパタアニメになってしまいます。

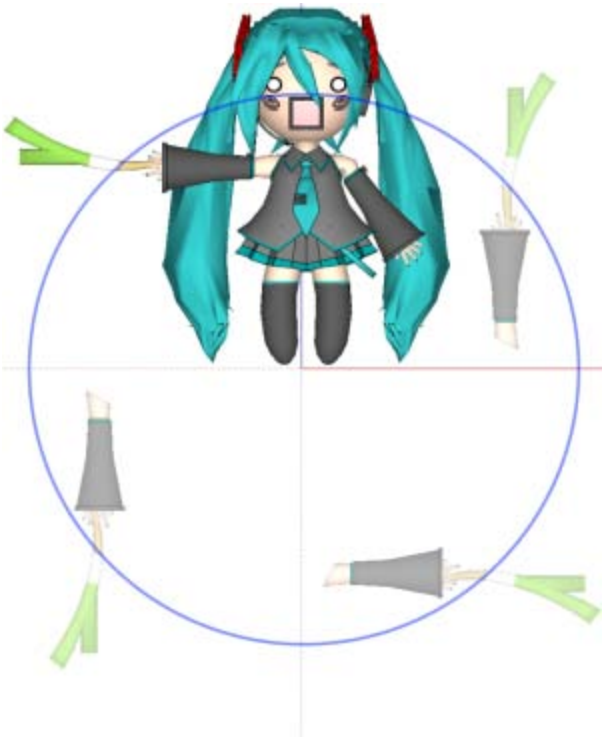


動画のネギはスムーズに上下しているように見えます。

TModel3D [3D Shapes] - その6

「じゃ、胴体と腕パーツに分けてあるんでしょ？」

ほぼ正解です。



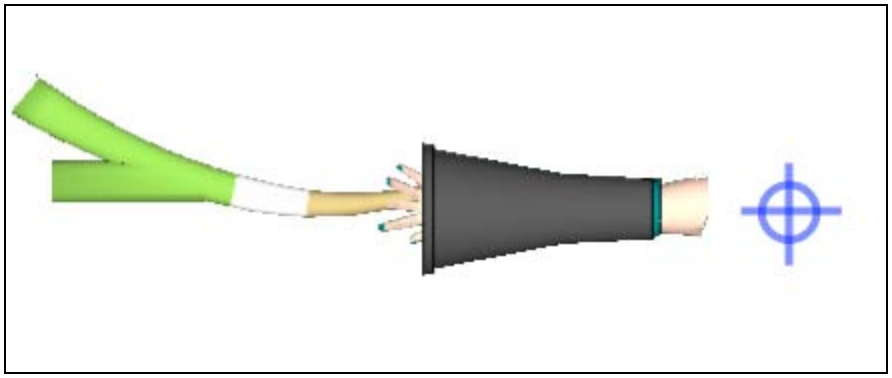
確かに胴体（TModel3D1）と腕パーツ（TModel3D2）を分け、TDummy でグループ化してあります。
これで TDummy を回転/移動させれば、胴体と腕は連動して回転/移動できます。

ですが、これだけではネギは振れません。
原点が足元にあるからです。

3D コントロールには、2D コントロールにあった RotationCenter プロパティが存在しないので、任意の 3D 座標での回転はできません。回転は原点で行われます。

※Update 4 には RotationCenter プロパティ があります。

TModel3D [3D Shapes] - その7



腕パーツのここに原点があれば
ネギを振れます。

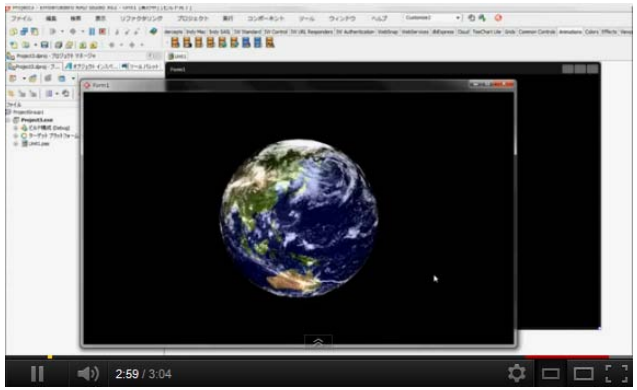
原点を変更する方法は 2 つあります。

1. モデルデータの時点で原点を変更し、TDummy の子として配置した時に
   位置を調整する。
2. モデルデータでは原点を変更せず、TDummy の子として
   さらに TDummy を配置し、そこに腕パーツを配置し、位置を調整する。

要はどうにかして関節部分に原点を持ってくればいいのです。
動画のものは前者を採用していますが、あらゆる箇所を動かすのであれば、
ネストした TDummy に分割したパーツを配置する必要があります。

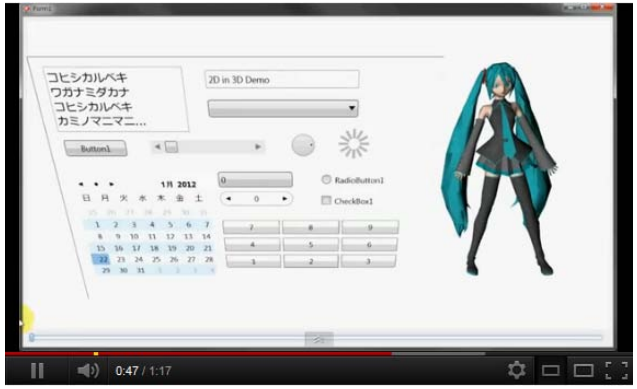
YouTube に動画をアップしてあります。

※ Update 3 で
作られたものです。



FireMonkey で作る地球儀
～ 多分、4 分以内に ～

FireMonkey で
PMD モデルを読み込んでみる



FireMonkey TLayer3D デモ

FireMonkey [3D Shapes] Demo

※ タイトル名で検索してみて下さい。

embarcadero
Developer Camp

Tips 的なもの

TLabel の色を変えるには？ (1)

- カスタムスタイルを設定する。

TLabel の色を変えるには？ (2)

1. TLabel を右クリックして、ポップアップメニューから
   [カスタムスタイルの編集] を選択。

2. スタイルデザイナの右ペインのツリーで "text: TText" を選択。

3. オブジェクトインスペクタで、Fill.Color を変更。

4. スタイルデザイナの右ペインの [適用して閉じる] ボタンを押下。

スタイルを変更すると、TStyleBook が自動的に
生成されます。
2 つ目以降の TLabel は、StyleLookup プロパティに、
最初の TLabel と同じ物を指定すれば OK です。
何度もスタイルデザイナで編集する必要はありません。

TComboBox / TListBox のフォントサイズを変えるには？

- TComboBox / TListBox には Font プロパティがないが、
  [項目エディタ] で作れる TListBoxItem は Font プロパティを持っている。

- TComboBox も TListBox もアイテムは TListBoxItem なので、
  TListBoxItem のカスタムスタイルを作ればいい。

- 現状、TComboBox からでは TListBoxItem のカスタムスタイルは作れない。
  TListBox で TListBoxItem のカスタムスタイルを作り、TComboBox で利用する。

- カスタムスタイルは TListBoxItem 毎に設定してやらなくてはならない*1。
  だったらカスタムスタイルを作らずに、最初からコードでフォントサイズを
  指定すればいいような気がする（いっそ、Scale プロパティで…^^;A）。

```
for i:=1 to 50 do
  ListBox1.Items.Add(Format('%.3d', [i]));

for i:=1 to 50 do
  ListBox1.ItemByIndex(i-1).Font.Size := 18;
```

Items.Add() ではなく、
AddObject() で
TListBoxItem を追加してもいい。

*1 スタイルデザイナから設定する方法を見つけられなかった。
スタイルファイルを直接いじればできるのかもしれないが、未確認。

雑多な情報 1

- フォームデザイナは埋め込みウィンドウである必要がある。
- dbGo (Windows のみ) / IBX / DBX4 は使える。BDE は使えない
- WebSnap / IntraWeb は使えるが、Windows 専用。
- Indy / TeeChart は使える。レポートツールは使えない。
- 入力系コントロールのパフォーマンスが悪い時は、DisabledFocusEffect プロパティを True にするか FMX.Types.GlobalDisableFocusEffect を True に設定する。
- データモジュールにコンポーネントを置けない場合には、ClassGroup プロパティを見直す (FMX.Types.TControl 等に変更)。
- クリップボード / ペーストボードは Platform 変数の GetClipboard / SetClipboard を使う。但しテキスト専用。

雑多な情報 2

- TListView のフォントサイズを変える方法は TComboBox / TListBox の時同様。TTreeViewItem が Font プロパティを持っている。
- フォームデザイナでコピー＆ペーストできない場合には、フォームデザイナまたは [構造] ペインで右クリックしてコピーしたものを貼りつけてみる。
- 最終手段はフォームを [エディタで表示] にして編集。ペーストはできる事が多いので、テキストエディタにフォームのソースを貼りつけておくといい。
- TViewPort3D に貼りつけた 3D コントロールが常に中央に来て困る場合には、ワーク用の TViewPort3D を貼りつけておき、そちらで作業したものをコピペで戻すと楽。FireMonkey 3D アプリケーションの場合は作業用のフォームを別途作っておくと楽になる。
- アニメーションさせるとコントロールがチラつく場合には、FMX.Types. AniFrameRate を変更してみるといいかもしれない (規定値 = 30)。

embarcadero
Developer Camp

ファーストインプレッション

ファーストインプレッション (ネガティブ編)

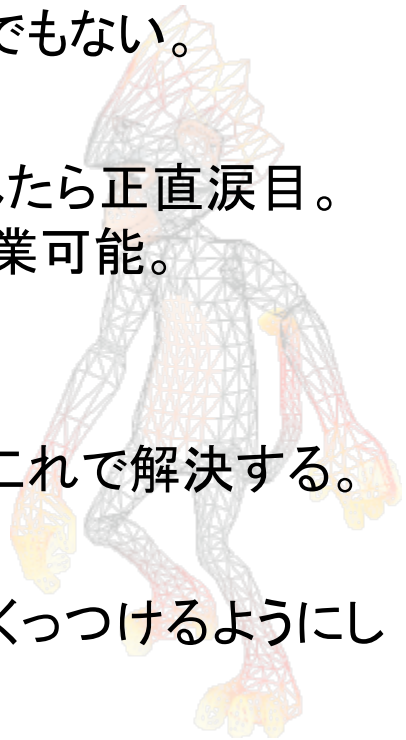
- フォームデザイナには結構イライラさせられる。
- 音声再生コントロールがあり、
  LiveBinding のノーティファイアがプロパティにあれば、
  ノーコーディングで Flash みたいな事ができるのだけれども...。
- TVideo / TVideo3D ...というか、動画再生コントロールも要るよね。
- 仮想キーボードはタッチ系のマシンには必須かも。
- IM (Input Method) の挙動が所々おかしい。
- ヘルプが貧弱。「サンプルデモで確認してくださいね」と
  書いてある割には、意味不明なデモになっているものが多い。

※意見には個人差があります。

ファーストインプレッション (ポジティブ編)

- 理屈的には、FireMonkey というフレームワークから逸脱しなければ、殆ど何も考えずにマルチプラットホームが可能 (iOS を除く)。
- 0 (できない) と 1 (できる) の間には大きな隔たりがある。
  BDS2006 で .NET Compact の開発をするような強引さでもない。
- 3D が簡単に使えるようになった。「自前でやれ」と言われたら正直涙目。
  モデルの生成に Google SketchUp とかが使えるので分業可能。
- スケール変更と回転は本当に便利。
  スレート PC プログラミングで苦労していた殆どの部分はこれで解決する。
- すべてのコントロールの OnChange にノーティファイアをくっつけるようにしておけば、UI デザイン (UI 制御) とロジックを分離できる。

※意見には個人差があります。



まとめ

- 現時点では確かに「足りない」と感じるものも少なくはない。
  ...ただ、いきなりフルセットを出されても使いこなせないであろう事は
  想像に難くないため、まずは "できること" から始めた方がいいように思う。

- VCL (Windows) アプリケーションとの "流儀の違い" を知るのが肝要。
  自分で認識している "常識" は、"Windows アプリの常識" かもしれない。

- VCL アプリケーションが得意とする分野と、FMX アプリケーションが得意とする分野は若干異なる。

- 加えて、なんでもかんでも単一バイナリで済ませようとしなければ、できる事の幅は格段に広くなる。

- Mac OSX で動作する ネイティブ IDE と VCL for Mac OSX があれば
  最高なんですけどね (^^;A

embarcadero
Developer Camp

Q & A

embarcadero
Developer Camp

Extras

FireMonkey 関連情報 &資料1

- [FireMonkey アプリケーション プラットフォーム (DocWiki)]
  http://docwiki.embarcadero.com/RADStudio/ja/FireMonkey_アプリケーション_プラットフォーム
- [Embarcadero Discussion Forums ≫ C++Builder ≫ FireMonkey]
  https://forums.embarcadero.com/forum.jspa?forumID=379
- [Embarcadero Discussion Forums ≫ Delphi ≫ FireMonkey]
  https://forums.embarcadero.com/forum.jspa?forumID=380
- [Embarcadero Technologies Channel (YouTube)]
  http://www.youtube.com/user/EmbarcaderoTechNet#g/u
- [Embarcadero Japan Channel (YouTube)]
  http://www.youtube.com/user/EmbarcaderoTechJapan#g/u

FireMonkey 関連情報&資料2

- [FireMonkey (Stack Overflow)]
  http://stackoverflow.com/questions/tagged/firemonkey

- [Delphi (FireMonkey) によるテクニック＆アルゴリズム]
  http://ht-deko.minim.ne.jp/techalgof.html

- [Delphi / C++ Builder 関連サイト内検索]
  http://ht-deko.minim.ne.jp/searchresult2.html

- [3Dグラフィックス・マニアックス （マイナビニュース)]
  http://news.mynavi.jp/column/graphics/index.html

- [Delphi Source Code (CodeCentral)]
  http://cc.embarcadero.com/ProdCat.aspx?prodid=1&catid=1
  ※ iOS 用のサンプルコードがあります。

FireMonkey 関連情報&資料3

- [Google SketchUp]
  http://sketchup.google.com/intl/ja/
- [Paint.NET]
  http://www.getpaint.net/
- [InkScape]
  http://inkscape.org/index.php?lang=ja
- [初音ミク　ネギ　ミニ]
  http://sketchup.google.com/3dwarehouse/details?mid=eea6a316ae7a45266d866ceb355191d
- [初音ミク　ネンドロイド普通(-"-)]
  http://sketchup.google.com/3dwarehouse/details?mid=dfc7bba7b1a561829fe3ed3275af47a7