はじめてのDataSnap

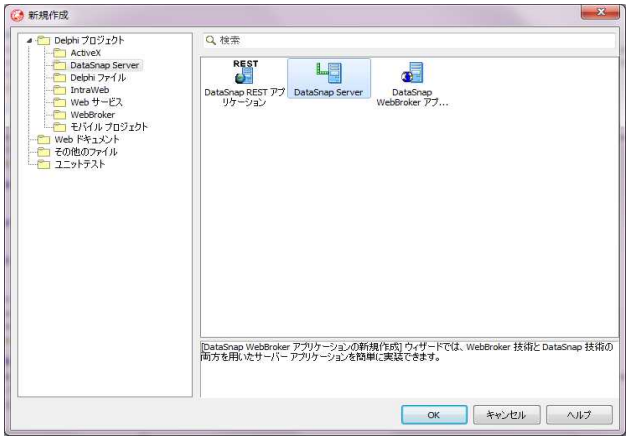
Windowsサービスを使ってサーバーを作成

Visual N@VI
Oracle Object Management Tool

新規プロジェクトを作成

ウィザードを使って新規プロジェクトを作成する

・[ファイル | 新規作成 | その他...] メニューを選択すると、下の画面が表示される
・左のペインから「DataSnap Server」、右のペインから「DataSnap Server」を選択し「OK」ボタンを押す



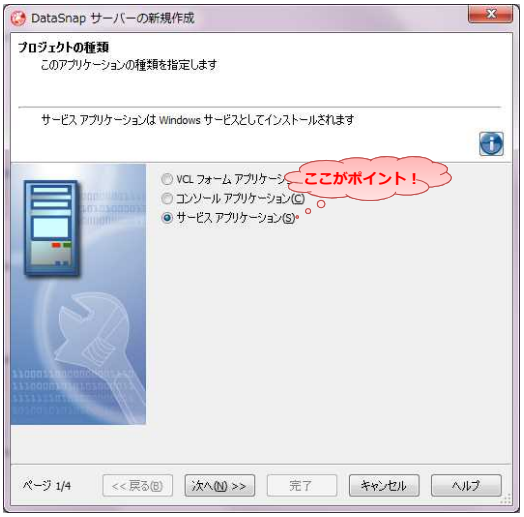
プロジェクトの種類を選択

DataSnapサーバーの種類を選択する

作成するアプリケーション・タイプを次の3種類から選択

・VCLフォームアプリケーション
・コンソールアプリケーション
・(Windows)サービスアプリケーション

ここでは、「サービスアプリケーション」を選択し「次へ」を押す



サーバーの機能を選択

DataSnapサーバーに追加する機能を選択する

・(通信)プロトコル
・認証
・サーバーメソッドクラス
・フィルタ
・JavaScriptファイル
・モバイルコネクタ

ここでは、標準設定のまま「次へ」を押す

ポート番号の指定

クライアントからのリスニングを行うポート番号を指定する

「サーバー機能の選択」で選択した「プロトコル」のポート番号を指定

ここでは、標準設定のまま「次へ」を押す

※[ポートのテスト]ボタンを押すと、指定のポートが使用可能かどうかを確認できる

Visual N@VI
Oracle Object Management Tool

サーバー・メソッド・クラスの上位クラスを指定

サーバーメソッド クラスの上位の型を指定する

- TComponent
- TDataModule
- TDSServerModule

ここでは、「TComponent」を選択し「完了」を押す

Visual N@VI
Oracle Object Management Tool

サーバープログラムの自動生成

ウィザードで次の2つのユニットが自動生成される

[ServerContainerUnit1]

```
unit ServerContainerUnit1;

interface

uses System.SysUtils, System.
  Vcl.SvcMgr,
  Datasnap.DSTCPServerTranspo
  Datasnap.DSServer, Datasnap
  Datasnap.DSAuth, IPPeerServ

type
  TServerContainer1 = class(T
    DSServer1: TDSServer;
    DSTCPServerTransport1: TD
    DSServerClass1: TDSServer
    procedure DSServerClass1G
      var PersistentClass: TP
    procedure ServiceStart(Se
  private
    { private 宣言 }
  protected
    function DoStop: Boolean;
    function DoPause: Boolean
    function DoContinue: Boo
    procedure DoInterrogate;
  public
    function GetServiceContro
  end;
   :
```

[ServerMethodsUnit1]

```
unit ServerMethodsUnit1;

interface

uses System.SysUtils, System.Classes, Datasnap.DSServer, Datasnap.DSAuth;

type
{$METHODINFO ON}
  TServerMethods1 = class(TComponent)
  private
    { private 宣言 }
  public
    { public 宣言 }
    function EchoString(Value: string): string;
    function ReverseString(Value: string): string;
  end;
{$METHODINFO OFF}

implementation

uses System.StrUtils;

function TServerMethods1.EchoString(Value: string): string;
begin
  Result := Value;
end;

   :
```

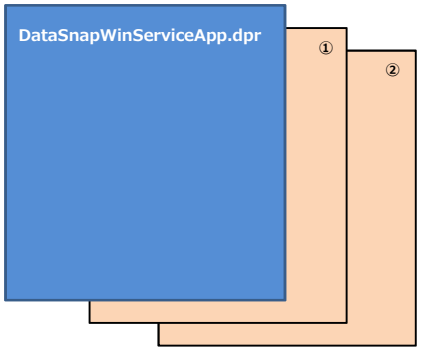

Visual N@VI
Oracle Object Management Tool

プロジェクトの保存とビルド

[ファイル | すべて保存]を実行する

・プロジェクトを「 DataSnapWinServiceApp.dpr」と言う名前で保存する
・他はそのままの名前で保存する

「ビルド」を実行する



① ServerContainerUnit1.pas
② ServerMethodsUnit1.pas

はじめてのDataSnap

生成されたユニットを見てみる

Visual N@VI
Oracle Object Management Tool

サーバー・コンテナ ユニットを見てみる（1/4）

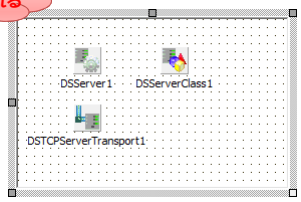
ウィザードで自動生成される「ServerContainerUnit1」には、3つ*1 のコンポーネントが配置されている

- **TDSServer :**
  すべてのDataSnapコンポーネントを結びつけるための
  メインのサーバー設定コンポーネント

- **TDSServerClass :**
  公開するクラスごとに必要なコンポーネント。
  publicなインターフェイスを持つクラスを参照する

- **TDSTCPServerTransport :**
  転送プロトコルと使用するTCP/IP等の設定を定義する
  コンポーネント

必ず配置される



*1 「サーバー機能の選択」でチェックした内容によって配置されるコンポーネントが異なる

サーバー・コンテナ ユニットを見てみる（2/4）

```
unit ServerContainerUnit1;

interface

uses System.SysUtils, System.Classes,
  Vcl.SvcMgr,
  Datasnap.DSTCPServerTransport,
  Datasnap.DSServer, Datasnap.DSCommonServer,
  Datasnap.DSAuth, IPPeerServer;

type
  TServerContainer1 = class(TService)
    DSServer1: TDSServer;
    DSTCPServerTransport1: TDSTCPServerTransport;
    DSServerClass1: TDSServerClass;
    procedure DSServerClass1GetClass(DSServerClass: TDSServerClass;
      var PersistentClass: TPersistentClass);
    procedure ServiceStart(Sender: TService; var Started: Boolean);
  private
    { private 宣言 }
  protected
    function DoStop: Boolean; override;
    function DoPause: Boolean; override;
    function DoContinue: Boolean; override;
    procedure DoInterrogate; override;
  public
    function GetServiceController: TServiceController; override;
  end;

var
  ServerContainer1: TServerContainer1;

implementation

uses Winapi.Windows, ServerMethodsUnit1;

{$R *.dfm}
```

TServiceから派生

自動追加されている

サーバー・コンテナ ユニットを見てみる（3/4）

```
procedure TServerContainer1.DSServerClass1GetClass(
  DSServerClass: TDSServerClass; var PersistentClass: TPersistentClass);
begin
  PersistentClass := ServerMethodsUnit1.TServerMethods1;
end;

procedure TServerContainer1.ServiceStart(Sender: TService; var Started: Boolean);
begin
  DSServer1.Start;
end;

function TServerContainer1.GetServiceController: TServiceController;
begin
  Result := ServiceController;
end;

procedure ServiceController(CtrlCode: DWord); stdcall;
begin
  ServerContainer1.Controller(CtrlCode);
End;
```

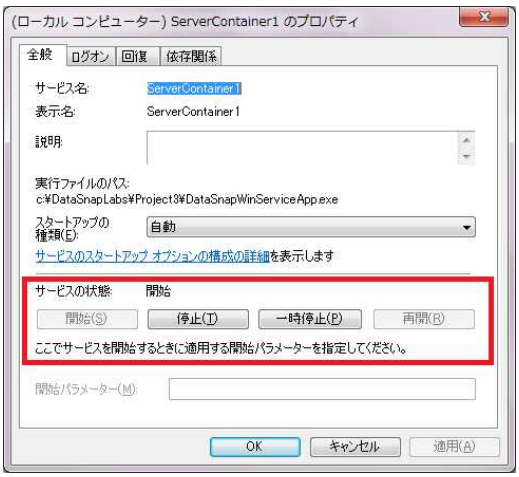
```
procedure TServerContainer1.DoInterrogate;
begin
  inherited;
end;

function TServerContainer1.DoStop: Boolean;
begin
  DSServer1.Stop;
  Result := inherited;
end;

function TServerContainer1.DoPause: Boolean;
begin
  DSServer1.Stop;
  Result := inherited;
end;

function TServerContainer1.DoContinue: Boolean;
begin
  Result := inherited;
  DSServer1.Start;
end;

end.
```

Visual N@VI
Oracle Object Management Tool

サーバー・コンテナ ユニットを見てみる（4/4）

ウィザードで自動生成される「ServerContainerUnit1.pas」には、サービス・アプレット内の操作（赤枠）で呼び出される4つのメソッドが生成される

- **DoInterrogate :**
  「開始」ボタンが押されたときに呼び出される
- **DoStop :**
  「停止」ボタンが押されたときに呼び出される
- **DoPause :**
  「一時停止」ボタンが押されたときに呼び出される
- **DoContinue :**
  「再開」ボタンが押されたときに呼び出される



Visual N@VI
Oracle Object Management Tool

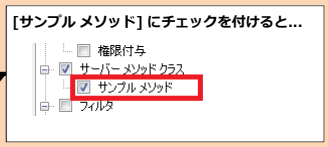
サーバー・メソッド ユニットを見てみる

ServerMethodsUnit1には、サーバー・クラスとサーバーメソッドが自動的に追加されている

```
unit ServerMethodsUnit1;

interface

uses System.SysUtils, System.Classes, Datasnap.DSServer, D

type
{$METHODINFO ON}
  TServerMethods1 = class(TComponent)
  private
    { private 宣言 }
  public
    { public 宣言 }
    function EchoString(Value: string): string;
    function ReverseString(Value: string): string;
  end;
{$METHODINFO OFF}

implementation

uses System.StrUtils;

function TServerMethods1.EchoString(Value: string): string;
begin
  Result := Value;
end;

function TServerMethods1.ReverseString(Value: string): string;
begin
  Result := System.StrUtils.ReverseString(Value);
end;

end.
```

Server Class

[サンプル メソッド] にチェックを付けると...



サーバー・メソッドの追加

サーバー・クラス(TServerMethods1.pas)に、クライアントから呼び出されるメソッドを追加する

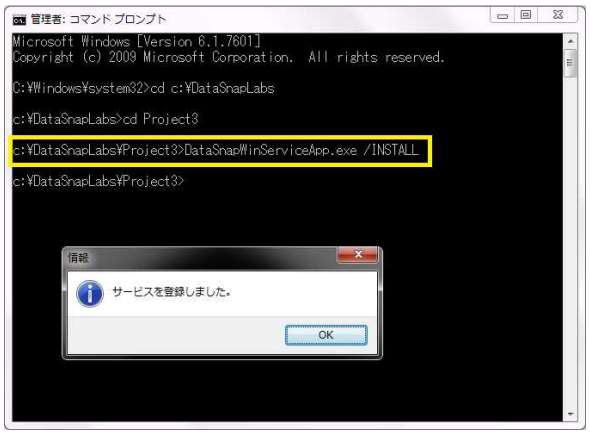
```
unit ServerMethodsUnit1;
...
{$METHODINFO ON}
  TServerMethods1 = class(TComponent)
      :
    function EchoString(Value: string): string;
    function ReverseString(Value: string): string;
    function Add(a, b: Double): Double;
    function Dic(a, b: Double): Double;
    function Mult(a, b: Double): Double;
    function Sub(a, b: Double): Double;
  end;
{$METHODINFO OFF}
...

function TServerMethods1.Add(a, b: Double): Double;
begin
  Result := a + b;
end;

function TServerMethods1.Dic(a, b: Double): Double;
begin
  Result := a / b;
end;

function TServerMethods1.Mult(a, b: Double): Double;
begin
  Result := a * b;
end;

function TServerMethods1.Sub(a, b: Double): Double;
begin
  Result := a - b;
end;
```

追加！

サービスのインストール

インストールは次の手順で行う

・管理者権限で「コマンド プロンプト」を起動する
・コマンドラインから次の通り入力し、 [Enter]キーを押す
［**パス名**］［**DataSnapサーバ名**］ **/INSTALL**

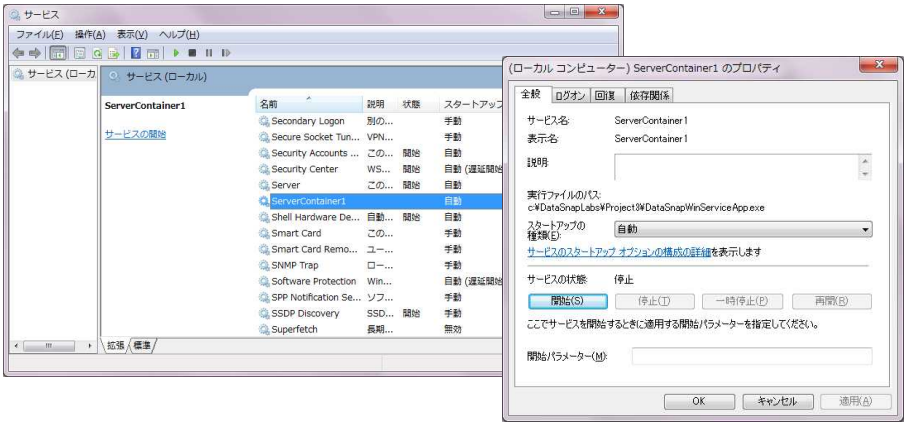


Visual N@VI
Oracle Object Management Tool

サービスの登録確認と起動

サービスが正しく登録されたかは、次の手順で確認することができる

・コントロールパネルを開き、[管理ツール][サービス]をクリックして「サービス」を起動する
・サービス名(**ServerContainer1**)が登録されているか確認
・状態が「開始」となっていない場合は、サービス名をダブルクリックし「プロパティ画面」から[開始]ボタンを押す



Visual N@VI
Oracle Object Management Tool

サービスの削除

登録されたサービスは、次の手順で削除することができる

・管理者権限で「コマンド プロンプト」を起動する
・コマンドラインから次の通り入力し、 [Enter]キーを押す
sc.exe DELETE [サービス名]