embarcadero

17th Embarcadero Developer Camp

【B3】Delphi/C++テクニカル セッション

「Windowsサービスアプリケーションの作成と連携アプリケーションへの応用」

株式会社シリアルゲームズ取締役
細川淳

embarcadero

17th Embarcadero
Developer Camp

Windows
サービスについて

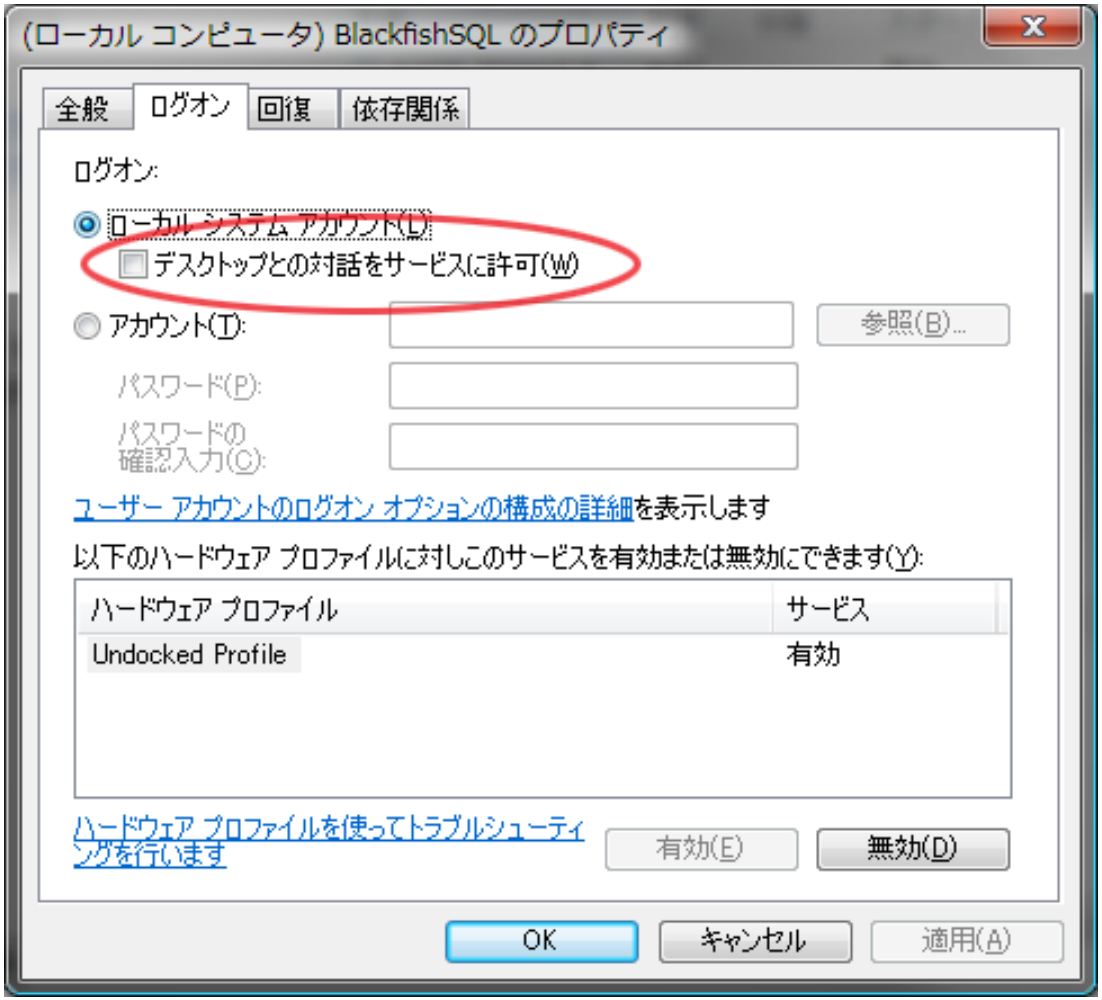
Windowsサービスとは

- ログインせずに動作できます
  - ユーザーは「LOCAL_SERVICE」など特別なユーザーとして実行されます。
- 自動的に起動できます
- バックグラウンドで動作し続けます

名前	説明	状態	スタートアップの種類	ログオン
##Id_String1.6844F930_1628_4223_B...	##Id_String2.6844F930_1628_4223_B5CC_5BB94B879762##		無効	Local System
Adobe Version Cue CS3 {ja_JP}	Adobe Version Cue CS3 {ja_JP}		手動	Local Service
Application Experience	起動するアプリケーションに対して、アプリケーションの互換性のキャッシュ要求を処理...	開始	自動	Local System
Application Host Helper Service	IIS に構成履歴やアプリケーション プール アカウントのマッピングなどの管理サービス...	開始	自動	Local System
Application Information	追加の管理者特権による対話型アプリケーションの実行を容易にします。このサービスが...		手動	Local System
Application Layer Gateway Service	インターネット接続共有に使用する、サード パーティーのプロトコル プラグイン用のサ...		手動	Local Service
Application Management	グループ ポリシーで展開されるソフトウェアに対する、プロセスのインストール、削除お...		手動	Local System
ASP.NET State Service	ASP.NET のアウト プロセス セッションに対するサポートを提供します。このサービスが...		手動	Network Service
Ati External Event Utility		開始	自動	Local System
Background Intelligent Transfer Service	アイドル状態のネットワーク帯域幅を使ってバックグラウンドでファイルを転送します。...	開始	自動 (遅延開始)	Local System
Base Filtering Engine	ベース フィルタ エンジン (BFE) は、ファイアウォールとインターネット プロトコル セ...	開始	自動	Local Service
BlackfishSQL	CodeGear NDataStore リレーショナル SQL データベースです。	開始	自動	Local System
Block Level Backup Engine Service	データのブロック レベルのバックアップおよび回復を実行するためのエンジン		手動	Local System
Certificate Propagation	スマート カードの証明書を伝達します。		手動	Local System
CNG Key Isolation	CNG キー分離サービスは、LSA プロセスでホストされています。このサービスは、Com...		手動	Local System
COM+ Event System	サポート システム イベント通知サービス (SENS) は、イベント通知先として登録された ...	開始	自動	Local Service
COM+ System Application	コンポーネント オブジェクト モデル (COM)+ ベース コンポーネントの構成と追跡を管理...		手動	Local System
Computer Browser	ネットワーク上のコンピュータの最新の一覧を管理し、その参照者として指定されたコン...	開始	自動	Local System
Cryptographic Services	提供される管理サービスは、次の 4 つです。カタログ データベース サービス: Windows ...	開始	自動	Network Service
DCOM Server Process Launcher	DCOM サービスを起動する機能を提供します。	開始	自動	Local System
Desktop Window Manager Session Man...	デスクトップ ウィンドウ マネージャのスタートアップおよびメンテナンス サービスを提...	開始	自動	Local System
DFS Replication	ローカル またはワイド エリア ネットワーク (WAN) のネットワーク接続全体にわたって...		手動	Local System

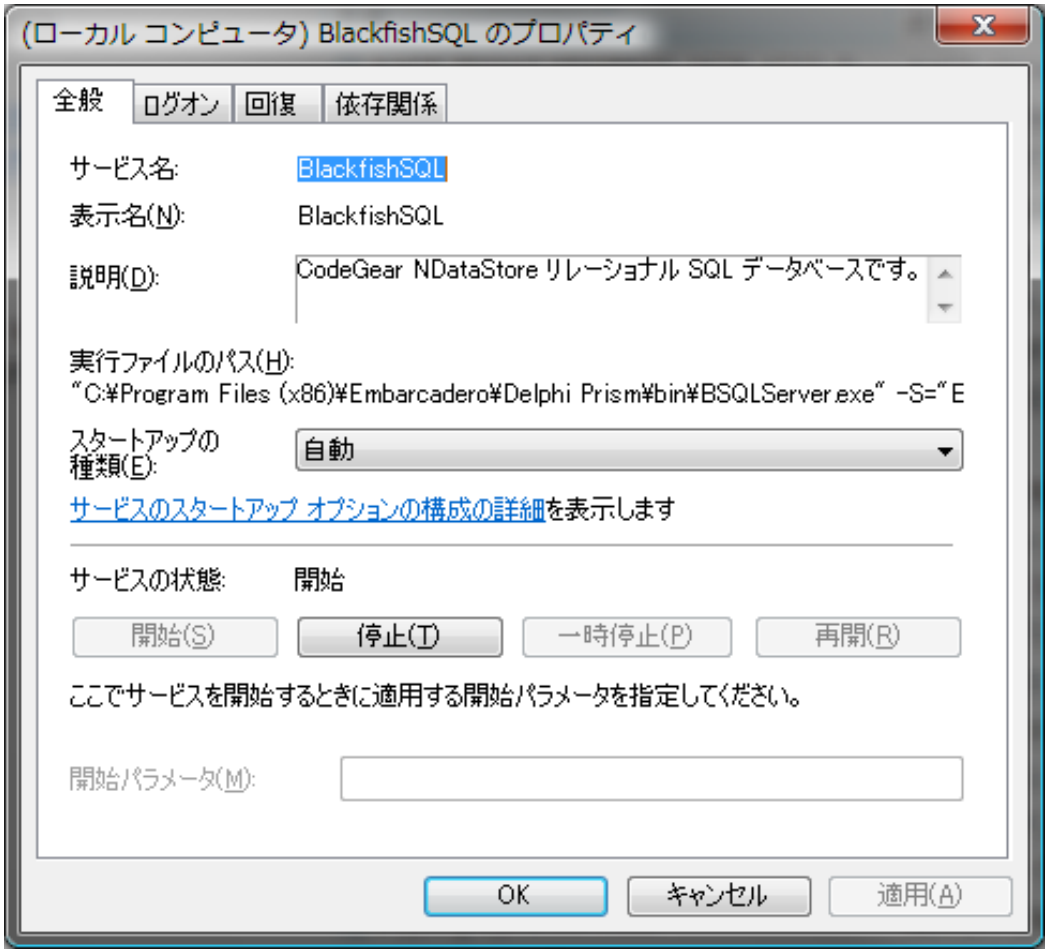
Windowsサービスとは

- UI を持たないアプリケーション
  - 「デスクトップとの対話を許可」にチェックをいれればGUIを表示することも可能です



Windowsサービスとは

- これらの特徴からウェブサーバなどのUIを必要とせず、かつ、必ず起動していなくてはならない用途に使われます。



例:
BlackfishSQL サービスのプロパティ

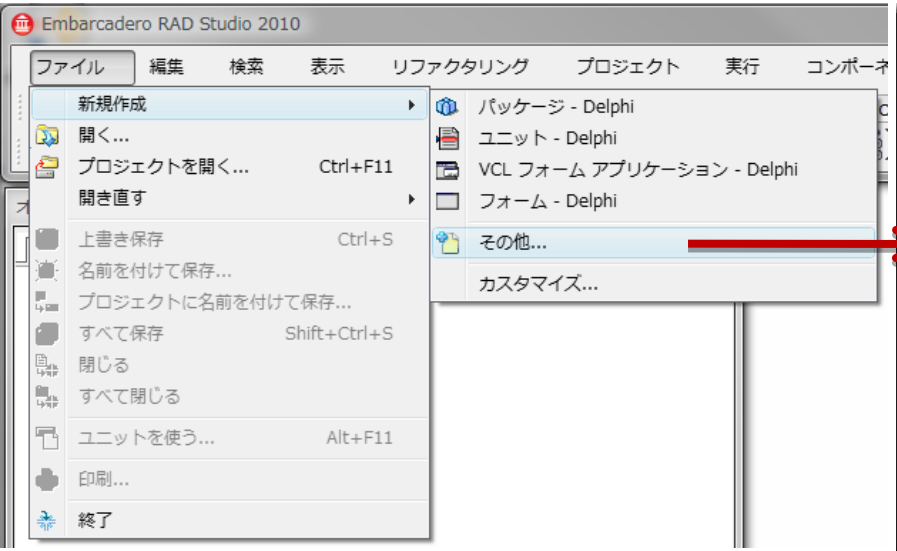
embarcadero

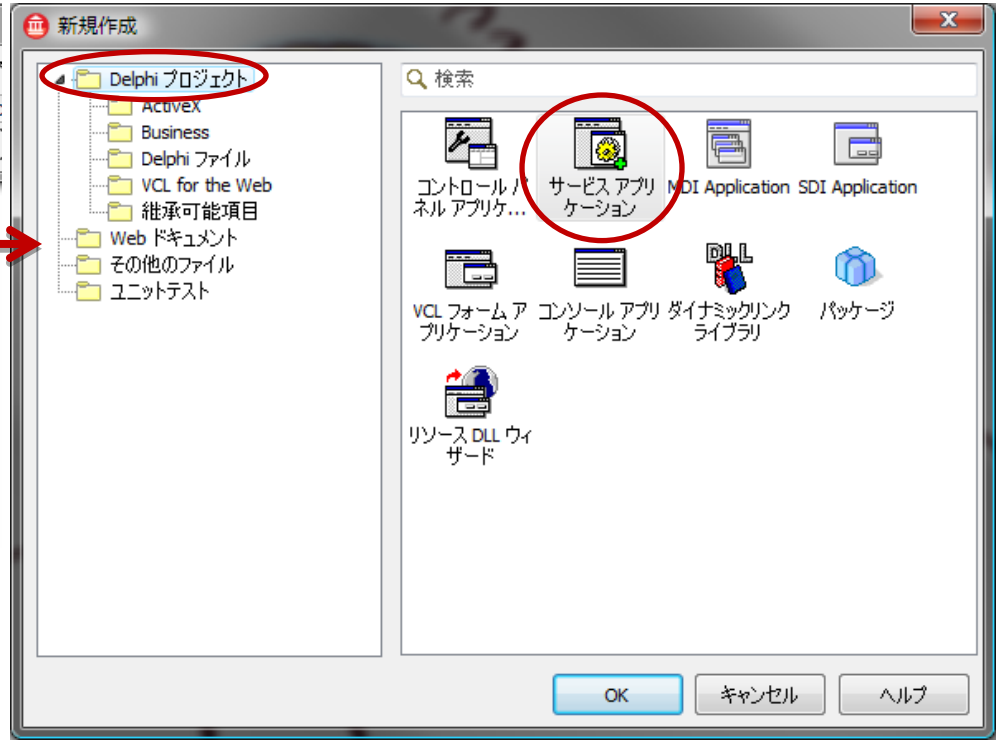
17th Embarcadero
Developer Camp

Delphi が
提供する機能

Delphi サービスプロジェクト

- 「ファイル」→「新規作成」→「その他」→「Delphiプロジェクト」→「サービス アプリケーション」としてサービスプロジェクトを作成可能です。

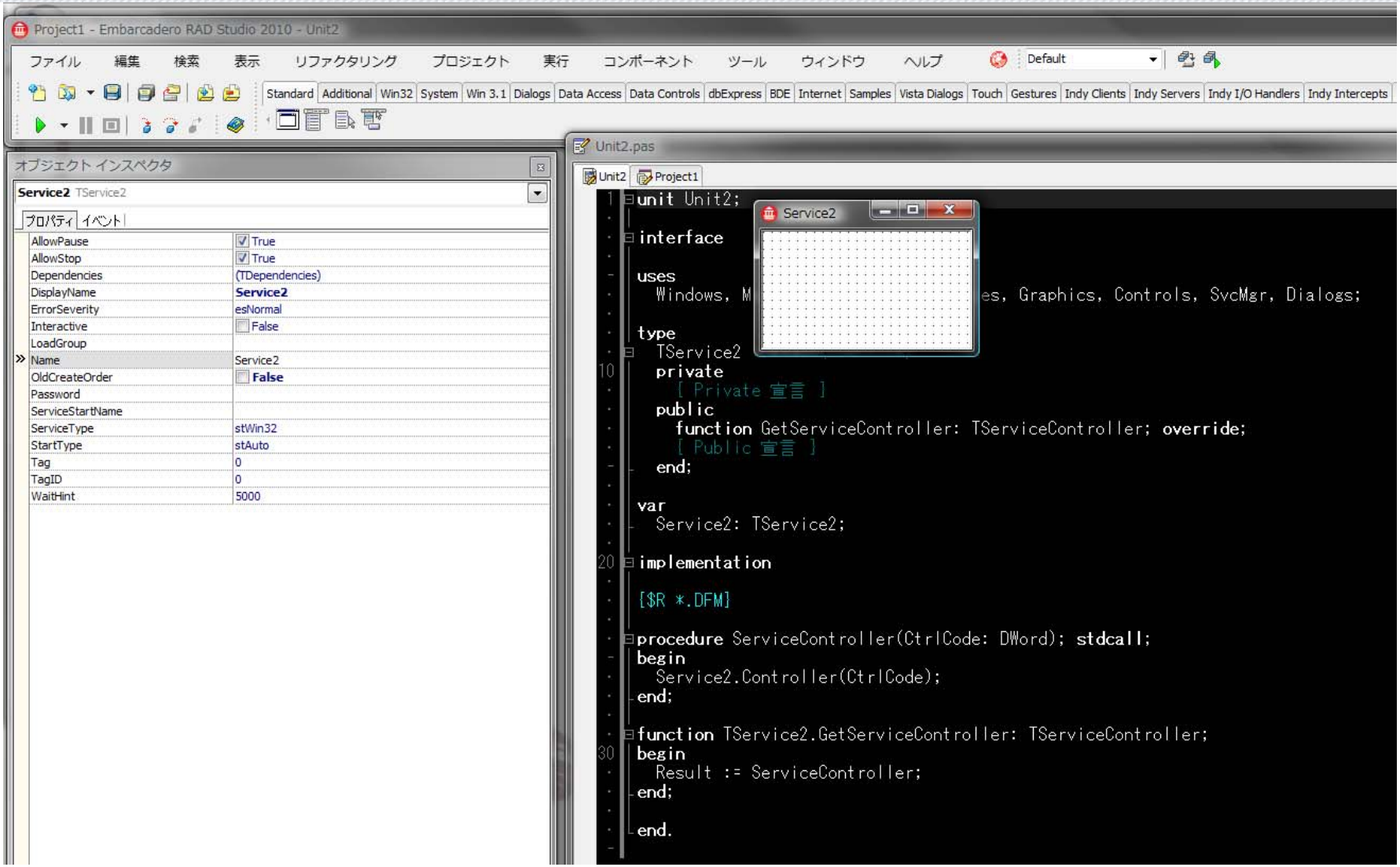




Delphi サービスプロジェクト

- プロジェクトを作成すると見慣れたフォームのような物が表示されますが、これは、TDataModule を継承した TService です。
- TDataModule を継承したクラスは RAD スタイルで開発可能です。
  - たとえば、非ビジュアルコンポーネントをドラッグ &ドロップして貼り付けておけます。
  - 今回作成するクラスでも TIdTCPServer を使用しています。

Delphi サービスプロジェクト



TService のプロパティ

- TService は、TForm と同じようにビジュアルコンポーネントなので、当然オブジェクトインスペクタ上でプロパティを設定できます。
- 順に見ていきましょう。

TService のプロパティ

- AllowStop: Boolean
  - サービスが停止可能かどうかを設定します。
- AllowPause: Boolean
  - サービスが一時停止可能かどうかを設定します。
- Dependencies: TDependencies
  - サービスの依存関係を設定します。
  - たとえば、ローカルネットワーク上のコンピュータのリストを必要とするサービスは「コンピュータブラウザ」サービスを必要とします。
- DisplayName: String
  - サービスの一覧に表示される名前です。

TService のプロパティ

- ErrorSeverity: TErrorSeverity
  - サービスが起動に失敗した場合の動作を定義します。
  - esIgnore　エラーを記録して動作を続けます
  - esNormal　エラーを記録しメッセージを表示しますが動作は続けます。
  - esSevere　エラーを記録し最後に適切であった環境設定が起動中のみ動作を続けます。それ以外の場合は最後に適切だった環境設定が起動されます。
  - esCritical　エラーを記録し最後に適切であった環境設定を代わりに起動します。現在の環境設定が最後だった場合は、起動に失敗します。

TService のプロパティ

- Interactive: Boolean
  - デスクトップとの対話が必要な場合に True にします。
- LoadGroup: String
  - ロードの順序をしめすグループの名前です。
  - 依存関係があるサービス同士の起動の順序を規定します。
- Password: String
  - サービスを実行するユーザーのパスワードです。
  - SYSTEM や、LOCAL_SERVICE といったビルトインユーザーの場合は必要ありません。
- ServiceStartName: String
  - サービスを実行するユーザーの名前です。

TService のプロパティ

- ServiceType: TServiceType
  - サービスのタイプです。
  - stWin32　　Win32 サービス
  - stDevice　　デバイスドライバ
  - stFileSystem　　ファイルシステムドライバ
- StartType: TStartType
  - stBoot　　ブートローダによって起動されます。デバイスドライバなどです。
  - stSystem　　システムの初期化後に起動されます。ファイルシステムなどです。
  - stAuto　　Windows 起動後、自動的に起動します。
  - stManual　　ユーザーが手動で起動します。
  - stDisabled　　無効。サービスは起動しません。
- TagID: DWord
  - LoadGroup で使用される一意な値です。

TService のイベント

- OnPause
  - 一時中断時に呼ばれます
- OnContinue
  - サービスの再開時に呼ばれます
- OnStart
  - サービスの起動時に呼ばれます
- OnStop
  - サービスの停止時に呼ばれます
- OnShutdown
  - サービス終了時に呼ばれます。

TService のイベント

- OnExecute
  - サービスを実現するスレッドが開始されたときに呼ばれます。
  - 基本的にはメッセージループを記述する場所です。
  - ただし、本来はこのハンドラを記述する必要はありません。
  - TServiceThread がサービス関連のメッセージを適切に処理します。
  - このハンドラを記述する必要があるのは、独自のロギング機構を組み込む必要がある場合などです
    - *Windows からサービスの開始や停止などの際に、独自にログを出力したりできます。*

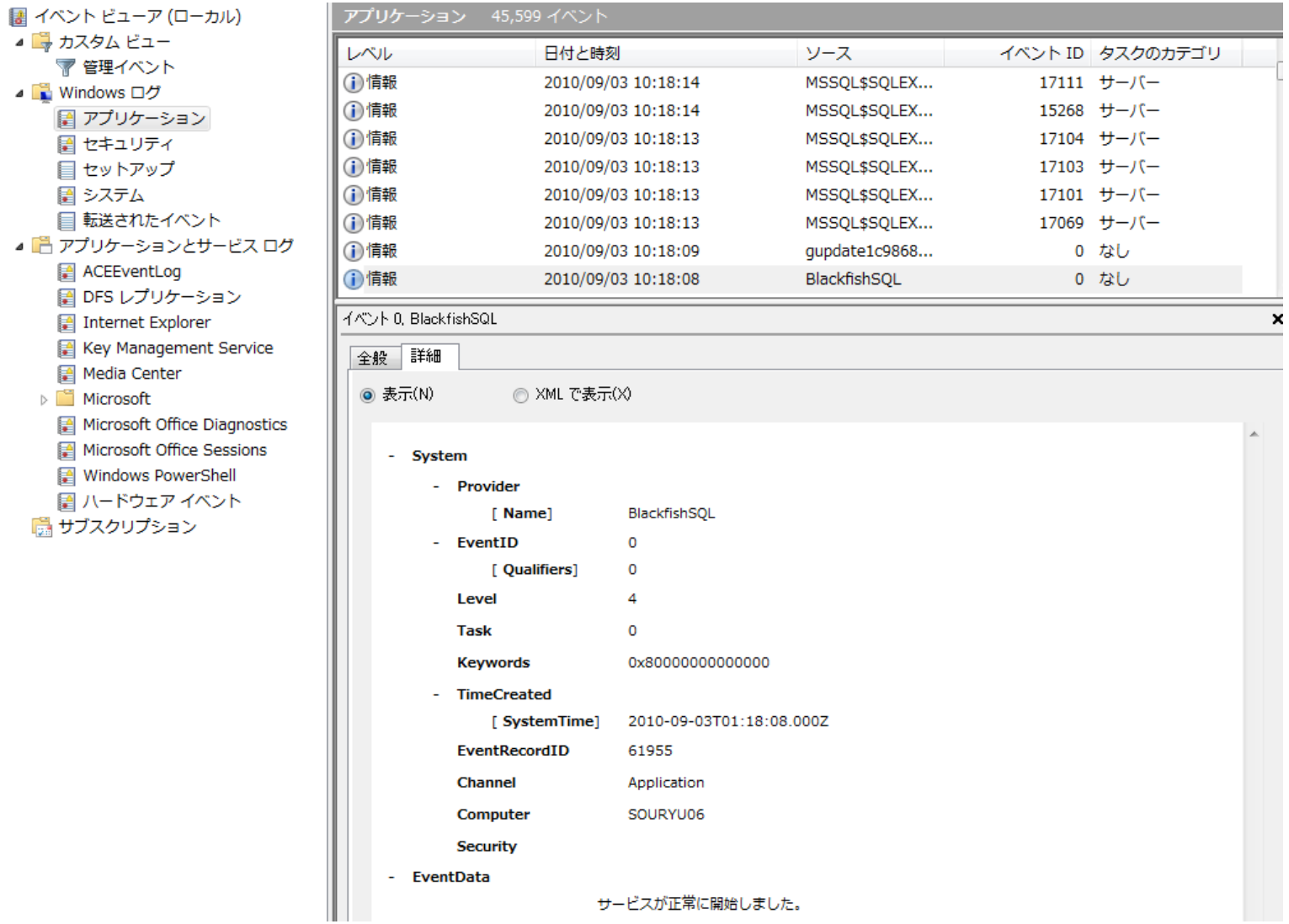
- これ以外に
  - OnCreate
  - OnDestroy

  などの普通のイベントも利用できます。

TService のメソッド

- TService のメソッドは基本的には呼ぶ必要がありません。
  - TService や TServiceApplication が Windows に対しての応答などを適切に処理します。
- ただし、1つ便利なメソッドがあります。
- LogMessage メソッドです。
  - このメソッドは、EventLog にログを書き込みます。
  - サービスは UI を持たないため、ログは非常に重要です。
  - 起動の状態などをこのメソッドを通じてイベントログにはき出せます。
  - try-except で補足したエラーをログにはき出せば、エラーも特定できます。
  - ただし、EventLog の形式で出力されるため、テキストエディタなどで簡単にみたり、ログを提出したりといった用途には向きません。

TService のメソッド



TServiceApplication

- プロジェクトソースを見ると、まるで普通のアプリケーションのようなコードになっています。

```
program BeepService;

uses
  SvcMgr,
  uMain in 'uMain.pas' {serviceBeep: TService},
  uLog in 'uLog.pas';

{$R *.RES}

begin
if not Application.DelayInitialize or Application.Installing then
    Application.Initialize;
  Application.CreateForm(TserviceBeep, serviceBeep);
  Application.Run;
end.
```

TServiceApplication

- しかし、この Application 変数は通常の TApplication ではなく、TServiceApplication のインスタンスです。
- TServiceApplication は、通常の TApplication と同様にアプリケーションの制御を司ります。
- Run メソッドで TService のインスタンスを起動し実行します。
- この機構により、プログラマはサービスの起動などのルーチンワークを組む必要なく、サービスの根幹のみをコーディングできます。

embarcadero

17th Embarcadero
Developer Camp

サービスの権限

サービスの権限

- サービスはビルトインユーザーとして実行されることが、ほとんどです。
- サービスのユーザーを指定するには、ドメインとユーザー名を指定します。
- ビルトインアカウントは NT_AUTHORITY というドメインに属しています。

表示名	System	LocalSystem	NetworkSystem
実際の名前	NT_AUTHORITY¥SYSTEM	NT_AUTHORITY¥LOCAL_SYSTEM	NT_AUTHORITY¥NETWORK_SYSTEM
権限	Administrators グループと同じ	Users グループと同じ	

embarcadero

17th Embarcadero
Developer Camp

サービスとの通信方法

通信方法

- サービスとの通信にはいくつかの方法があります
  - 通常のプロセス間通信の仕組みが全て使えます。

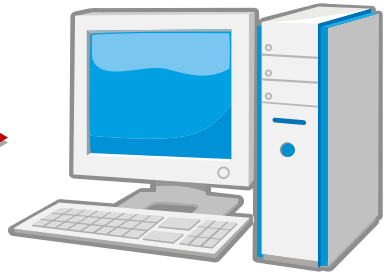
- RemoteProcedureCall
- 名前付きパイプ
- ソケット

- などなどです。

通信方法ー注意点

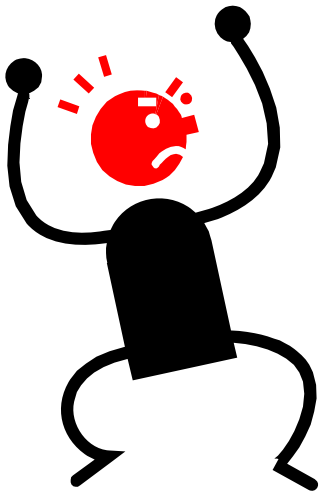
- サービスとクライアントプログラムは権限が違う
  - サービスは SYSTEM や LOCAL_SERVICE といった高い権限をもって動作します
  - サービスと接続するプログラムは通常、ログインしているユーザーの権限で動作します
  - 通常はサービスの権限が高いため問題にはなりませんが、本来は取得できないハズの情報にまでアクセスできてしまうので注意が必要です。
  - そのような場合は、次スライドの名前付きパイプを使うか、パスワードなどでアクセスしてきたプログラムの認証をする必要があります。

セキュリティの違うユーザーが
高いセキュリティを持ったサービスと通信することで
取得できないハズのデータを取得できてしまう！

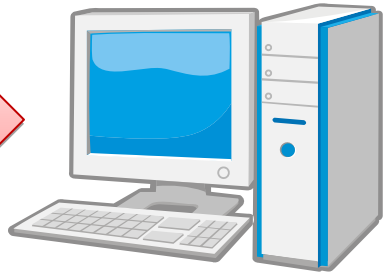


通信方法ー注意点

- ユーザー権限が必要な場合
  - 名前付きパイプを使うとパイプサーバ（今回はサービス）は、クライアントの権限で API を呼び出すことができます（**Impersonate**）。
  - セキュリティ以外にも、ユーザーのプロファイルが必要な場合などに利用できます。

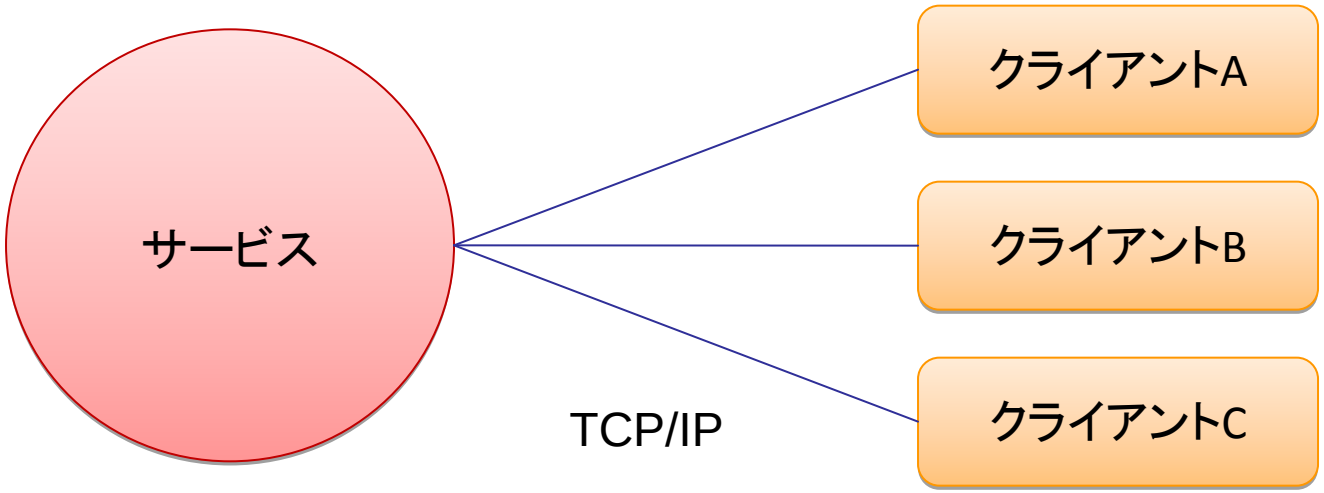


名前付きパイプを使うとクライアントの権限で API を呼び出すため安全です



通信方法

- Delphi には幸いなことに Indy がありますので、ソケットを使う方法が一番簡単です。
- 次に簡単なのは名前付きパイプでしょう。
  - しかし、名前付きパイプのクラスを作る必要があります。
- 今回は、もっとも簡単に使える TIdTCPServer と TIdTCPClient を使ってプログラムします。



embarcadero

17th Embarcadero
Developer Camp

実際のプログラム

作成するサービスについて

- 今回作成するサービスは以下のような動作とします

- TIdTCPServer で接続を待ち受けます
- TIdTCPServer で 'on' を受け取ると Beep 音を鳴らします
- TIdTCPService で 'off' を受け取ると Beep 音を消します

- ただこれだけの機能です。
- 権限は SYSTEM とします。
- 今回は敢えて OnExecute を実装し、サービスの動作に迫ります。

作成するサービスについて

- また、せっかくなので WinRing0 を使いハードウェアを直接叩いて Beep を鳴らします。
  - WinRing0 は、**Crystal Dew World** の hiyohiyo さんの著作物です。
  - ちなみに VCL の Beep関数 や Win32 API の Beep 関係の関数は、サウンドボードでエミュレーションされている場合があり、実際にマザーボード上のスピーカーから音が出ないことがあります。
  - そのため、このサービスでは WinRing0 を使用して直接ハードウェアを叩くことによって Beep 音を実現しています。

OnCreate, OnDestroy


```
procedure TserviceBeep.ServiceCreate(Sender: TObject);
var
  i: Integer;
begin
  FParams := TList.Create;

  ReadIni;

  FLog := TLog.Create(FLogDir, 'BeepService');
  FLogAdapter := TLogAdapter.Create(FLog, '');

  FLogAdapter.AddInfo('BeepService Created. ' + FExeName);

  for i := 0 to FParams.Count - 1 do
    with PBeepParam(FParams[i])^ do begin
      FLogAdapter.AddInfo(
        Format('Param%d  Freq = %d  Duration = %d', [i, Freq, Duration]));
    end;
end;

procedure TserviceBeep.ServiceDestroy(Sender: TObject);
begin
  Clear;
  FParams.Free;

  FLogAdapter.AddInfo('BeepService Destroyed.');
end;
```

OnCreate で初期化します。
OnStart でも初期化できますが、OnStart はサービスの開始ごとに呼ばれるため注意が必要です。

ここでは設定をファイルから読み込み、独自のログシステムを生成しています。

また、同様に OnDestroy で終了処理を行います。

OnExecute


```
procedure TserviceBeep.ServiceExecute(Sender: TService);
begin
  FLogAdapter.AddInfo('BeepService Executed.');

  InitializeOls;
  try
    FLogAdapter.AddInfo('WinRing0 Status: ' + IntToStr(GetDllStatus));

    tcpServer.DefaultPort := FPort;
    tcpServer.Active := True;
    try
      while (not Terminated) do begin
        if (ProcessMessages) then
          Break;

        if (not Beep) then
          Sleep(200);
      end;
    finally
      tcpServer.Active := False;
    end;
  finally
    DeinitializeOls;
  end;
end;
```

OnExecute のイベントハンドラです。
ここではメッセージループを作成し、サービスが終了するまでループしています。

ProcessMessages


```
function TserviceBeep.ProcessMessages: Boolean;
const
  ActionStr: array[1.. 5] of String =
    (SStop, SPause, SContinue, SInterrogate, SShutdown);
var
  Msg: TMsg;
  OldStatus: TCurrentStatus;
  ErrorMsg: String;
  ActionOK: Boolean;
begin
  Result := PeekMessage(Msg, 0, 0, 0, PM_REMOVE);

  if (not Result) then
    Exit;
```

ProcessMessages


```
with Msg do
    if (hwnd = 0) and (message = CM_SERVICE_CONTROL_CODE) then begin
      OldStatus := Status;
      try
        Result := False;
        ActionOK := True;

        case wParam of
          SERVICE_CONTROL_STOP: begin
            ActionOK := DoStop;
            Result := ActionOK;
          end;

          SERVICE_CONTROL_PAUSE: begin
            ActionOK := DoPause;
            Result := False;
          end;

          SERVICE_CONTROL_CONTINUE: begin
            ActionOK := DoContinue;
            Result := False;
          end;
```

ProcessMessages


```
      SERVICE_CONTROL_SHUTDOWN: begin
        DoShutDown;
        Result := True;
      end;

      SERVICE_CONTROL_INTERROGATE: begin
        DoInterrogate;
        Result := False;
      end;
    end;

    if (not ActionOK) then
      Status := OldStatus;
  except
    on E: Exception do begin
      if (wParam <> SERVICE_CONTROL_SHUTDOWN) then
        Status := OldStatus;

      if (wParam in [1.. 5]) then
        ErrorMsg :=
          Format(
            SServiceFailed,
            [ActionStr[Integer(wParam)],
            E.Message])
      else
        ErrorMsg := Format(SCustomError,[wParam, E.Message]);
```

```
          LogMessage(ErrorMsg);
        end;
      end;
    end
    else
      DispatchMessage(msg);
end;
```

OnConnect, OnDisconnect, OnException


```
procedure TserviceBeep.tcpServerConnect(AContext: TIdContext);
begin
  FLogAdapter.AddInfo('Connected by ' + AContext.Binding.PeerIP);
end;

procedure TserviceBeep.tcpServerDisconnect(AContext: TIdContext);
begin
  FLogAdapter.AddInfo('Disconnected by ' + AContext.Binding.PeerIP);
end;

procedure TserviceBeep.tcpServerException(AContext: TIdContext;
  AException: Exception);
begin
  FLogAdapter.AddWarning(AException.Message);
end;
```

OnExecute


```
procedure TserviceBeep.tcpServerExecute(AContext: TIdContext);
var
  Str: String;
begin
  if (AContext.Connection.IOHandler.Readable) then begin
    Str := AnsiLowerCase(Trim(AContext.Connection.IOHandler.ReadLn(#0)));

    FLogAdapter.AddInfo(Str + ' Read by ' + AContext.Binding.PeerIP);

    MonitorEnter(Self);
    try
      if (Str = 'on') then begin
        FEnabled := True;
        FIndex := 0;
      end;

      if (Str = 'off') then
        FEnabled := False;
    finally
      MonitorExit(Self);
    end;
  end;
end;
```

Beep


```
function TserviceBeep.Beep: Boolean;
var
  BeepHz: Integer;
begin
  Result := False;

  if (not FEnabled) or (FIndex < 0) then
    Exit;

  if (FIndex >= FParams.Count) then
    FIndex := 0;

  with PBeepParam(FParams[FIndex])^ do begin
    if (Freq <> 0) then begin
      BeepHz := 1193180 div Freq;

      WritePortByte($43, $B6);
      WritePortByte($42, BeepHz and $FF);
      WritePortByte($42, (BeepHz shr 8) and $FF);
      WritePortByte($61, ReadPortByte($61) or $03);
    end;

    Sleep(Duration);
  end;
```

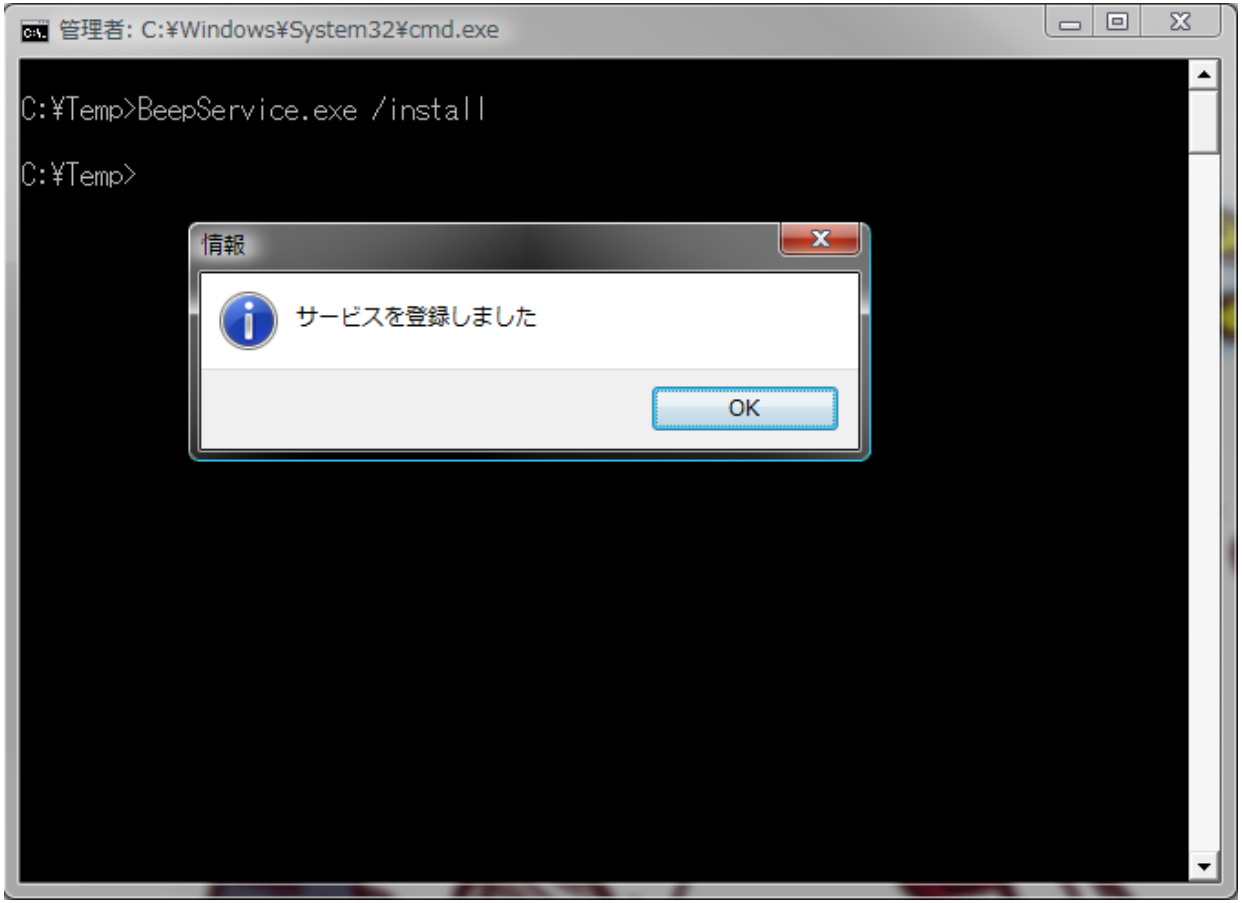
```
  WritePortByte($61, ReadPortByte($61) and $fc);

  Inc(FIndex);

  Result := True;
end;
```

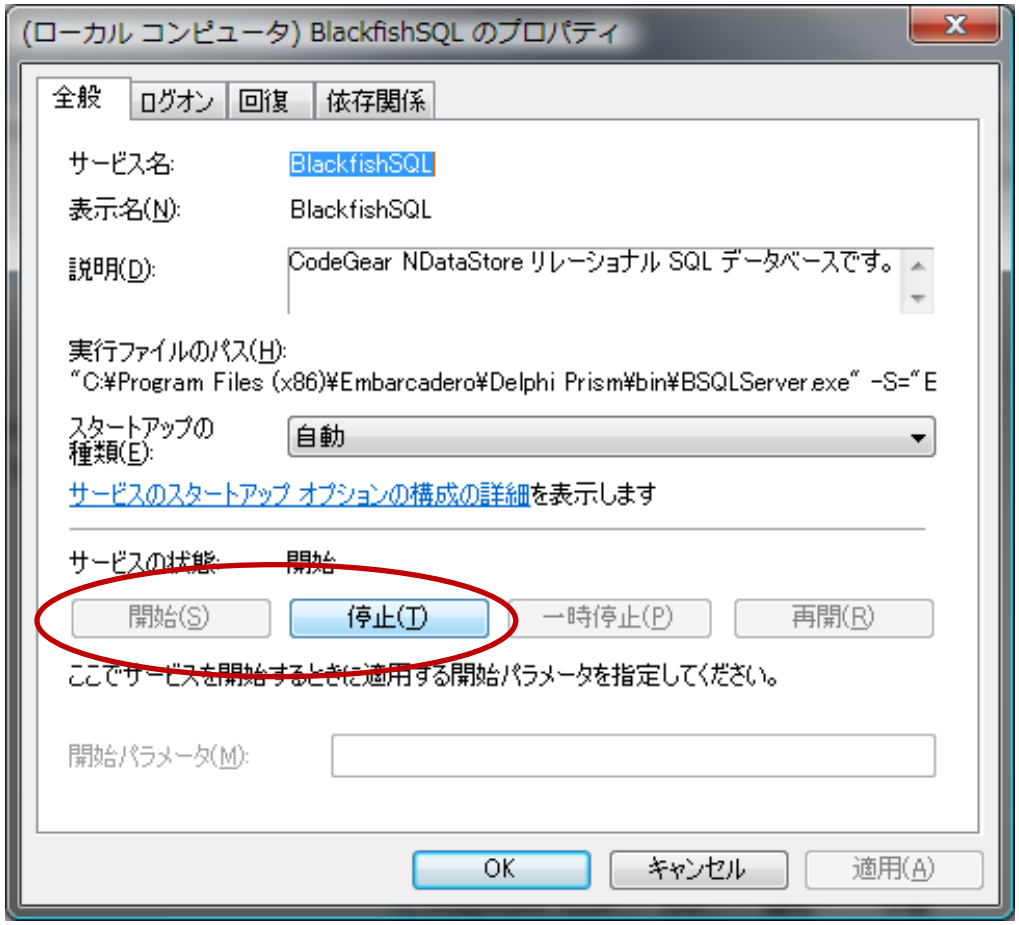
サービスのインストール・起動

- 作成したサービスは /Install でインストールできます。
- 同様に /Uninstall でアンインストールできます。
  - この機能は TServiceApplication が提供しています。



サービスのインストール・起動

- デバッグ中はマニュアルで開始・停止すると良いでしょう。
  - 実運用ではバッチファイルなので
  - "net start サービス名" などとして起動させます。



embarcadero

17th Embarcadero
Developer Camp

サービスのデモ

クライアントの作成

- クライアントは以下の動作とします

- サービスに TIdTCPClient を経由して接続します
- ボタンを押すとサービスに 'on' または 'off' を送信します。

クライアント側プログラム


```
procedure TForm1.Button1Click(Sender: TObject);
begin
  if (not tcpClient.Connected) then
    tcpClient.Connect;

  tcpClient.IOHandler.Write('on'#0);
end;

procedure TForm1.Button2Click(Sender: TObject);
begin
  tcpClient.IOHandler.Write('off'#0);

  if (tcpClient.Connected) then
    tcpClient.Disconnect;
end;
```

embarcadero

17th Embarcadero
Developer Camp

クライアントとの通信デモ

embarcadero

17th Embarcadero Developer Camp

まとめ

まとめ

- Delphi の提供しているプロジェクトを使えばサービスは簡単に作れます
- TService がサービスに必要な基本的な機能を提供しているため、プログラマはサービスの実際の動作に集中できます。
- サービスは高い権限で動作します。
- 作成したサービスはコマンドラインスイッチ Install, Uninstall でインストール・アンインストールできます。
- サービスとの通信は、ソケットや名前付きパイプを使います。
  - 権限が重要でない場合はソケット、重要である場合は名前付きパイプを使います

embarcadero

17th Embarcadero Developer Camp

Q & A