岡本安晴 2001.1.23;2001.3.3

Object Pascal 入門

その８

実行時例外

８-１　try..except 文

８-２　try..finally 文

８-３　例外の生成

8　実行時例外

プログラムの実行時にエラーがあると、例外というオブジェクトが生成される。生成された例外は、プログラム中で処理することができる。この処理によって、プログラムの実行時にエラーが発生しても、プログラム内で対応することができる。生成された例外の処理を行う文として、try..except 文と try..finally 文がある.

8-1　try..except 文

try..except 文は、

```
try
    文；　・・・；　文；
except
    例外の処理
end;
```

の形で用いる。try と except の間にある文の実行において実行時エラーが発生すると、except と end の間に記述されている例外の処理が実行される。例えば、次のリスト 8.1 の場合、

リスト 8.1　try..except 文の例

```
ck:=0;
while true do
  begin
    try
      x:=StrToFloat(InputBox('','x =','?'));
      y:=sqrt(x);
      ShowMessage(FloatToStrF(x,ffGeneral,5,1)+'の平方根 = '+
                  FloatToStrF(y,ffGeneral,5,1));
    except
      ck:=1;
    end;
    if ck <> 0 then goto 0;
  end;
0 : ShowMessage('例外が生成されました');
```

InputBox によって入力された文字列が StrFoFloat の処理に不適切な場合、あるいは変数 x に設定された値が負の数であるなどのため sqrt による計算ができないときは実行時エラーとなり例外が生成される。例外が生成されると、直ちに except と end の間の処理が実行される。上の場合、

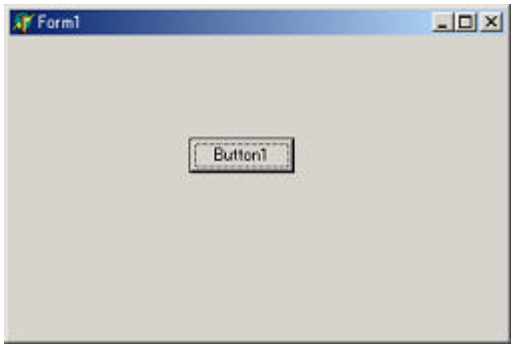
```
ck:=1;
```

が実行される。この except と end に挟まれた例外の処理の部分は、try と except の間の文の実行において例外が生成されない（実行時エラーが発生しない）ときは実行されない。したがって、

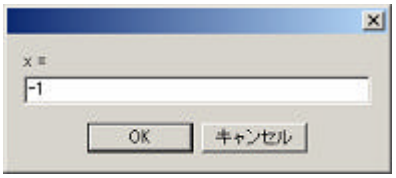
```
if ck <> 0 then goto 9;
```

における「goto 9」は、例外が生成されたときのみ実行される。すなわち、上のプログラムでは、実行時エラーが生じて例外が生成されたときに while 文の外に出ることができる。

サンプルプログラム PSample81.dpr は、上のプログラムを実行するものである。このプログラムを実行すると、次のフォームが表示される。

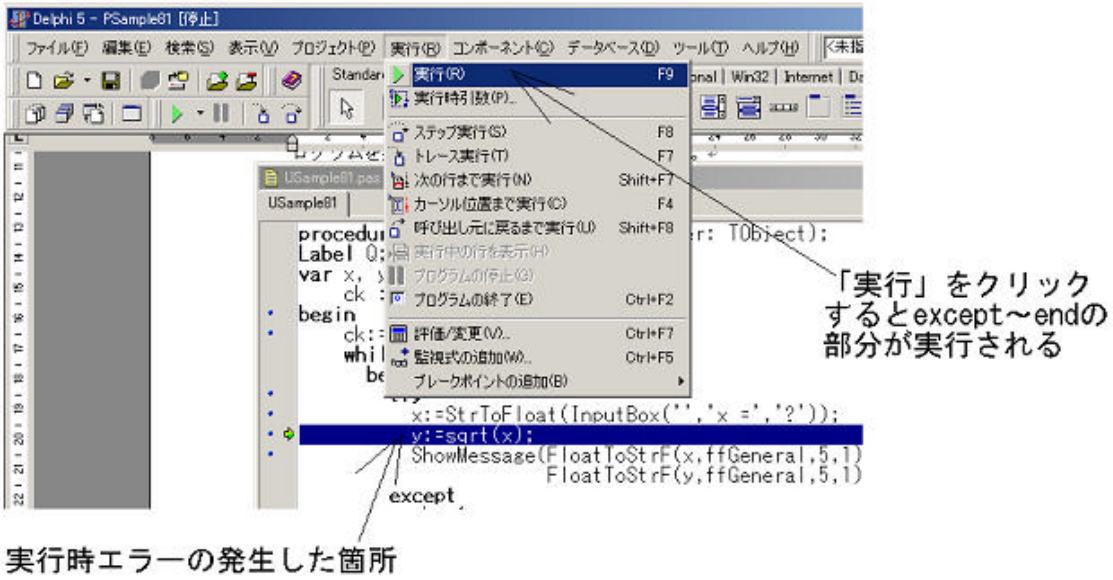


ボタンのクリックで関数 InputBox が呼び出されて、下図のようにダイアログボックスが表示される。



ここで、「－１」のように実行時エラーの発生する値を入力して「OK」ボタンをクリックすると、下図のようなエラーメッセージが表示される。

このエラーメッセージは Delphi によって表示されるもので、エラーメッセージの下の「OK」ボタンをクリックしてから Delphi のメニュ「実行｜実行」をクリックする。「実行｜実行」のクリック後、except～end の例外の処理の部分が実行される。コンパイル済みの*.exe ファイルをダブルクリックなどにより直接実行したときは、実行時エラーで例外が生成されると直ちに except～end の部分が実行される。Delphi の統合環境下でプログラムを実行しているときは、エラーの原因となった行が次の図のように表示される。



リスト 8.1 では

```
except
  ck:=1;
end;
```

となっているので、例外が生成されると ck の値が 1 となり、続く

```
if ck <> 0 then goto 9;
```

の実行により while 文を抜け出して、ラベル 9 の付いた

Q : ShowMessage('例外が生成されました');

が実行される。この ShowMessage の実行により次のメッセージボックスが表示される。

図 8.1　ShowMessage による表示

サンプルプログラム PSample81.dpr は、リスト 8.1 を実行するものである。このプログラムの場合、ShowMessage の次に Close が呼び出される。したがって、図 8.1 のメッセージボックスの「OK」ボタンをクリックすると ShowMessage の実行が終了して次の Close の呼び出しとなり、フォームが閉じられてプログラムの実行終了となる。

8-2　try..finally 文

実行時エラーが生じたとき、プログラム実行中に使用したリソースなどの処理を完了してからプログラムの実行を中断したいことがある。このとき、try..finally 文を用いているときは、try と finally の間の文で実行時エラーが発生しても、finally と end の間の文は必ず実行される。例えば次のリスト 8．2 の場合、

リスト 8.2　try..finally 文の例

```
ck:=1;
try

    try
        a:=StrToFloat(InputBox('','x = ','?'));
        b:=sqrt(a);

        // a あるいは b への代入文で例外が生成されると実行されない
        ck:=0;
    finally
        ShowMessage('ck = '+IntToStr(ck));
        if ck <> 0 then
          begin
            ShowMessage('例外が生成されました');
          end;
    end;

except
    ShowMessage('例外が再生成されていました');
end;

if ck = 0 then
  ShowMessage(FloatToStrF(a,ffGeneral,5,1)+
              'の平方根 = '+FloatToStrF(b,ffGeneral,5,1));
Close;
```

ａへの代入文、あるいはｂへの代入文において実行時エラーが生じても、エラーの生じた（例外の生成された）文から必ず finally 以降の部分に飛ぶ。エラーが生じなかった場合も、finally 以降の部分はそのまま実行される。

例外が生成されて finally～end の部分が実行されたとき、finally～end 部の実行後、例外が再生成される。したがって、上のプログラムのように try..except 文の中に try..finally 文が埋め込まれていると、例外の生成によって finally～end の部分が実行さ

れた後、except～end の部分が実行される。

サンプルプログラム PSample82.dpr は、リスト 8.2 を実行するものである。実行時の Delphi によるエラーメッセージの表示は PSample81.dpr の場合と同様であるが、このエラーメッセージの表示はａあるいはｂへの代入文でエラーが発生した場合で、finally～end の実行後に例外が再生成されるときには Delphi によるエラーメッセージの表示はない。

8-3　例外の生成

例外は、プログラムの実行において任意の箇所で生成することができる。したがって、プログラムの実行中に必要があれば、例外を生成して実行時エラーの発生とすることができる。例外はオブジェクトであるが、一番の親は Exception 型である。Exception 型の子孫として他の型の例外を宣言することができる。例えば、

```
Type  EObj = class(Exception) end;
```

と宣言すると、EObj という型の例外をプログラムの実行において生成することができる。例外の生成はオブジェクトの生成として行うが、raise 文を用いて例えば次のように行う。

```
raise  EObj.Create(文字列):
```

上の raise 文が実行されると、例外が生成されて実行時エラーの発生となり、Create の引数として設定された文字列を含む実行時エラーメッセージが表示される。あるいは、

```
raise  EObj.CreateFmt(文字列, [値リスト]);
```

の形式のときは、値リストにある値が文字列の中の書式に従って文字列中に表されたものを含む実行時エラーメッセージが表示される。

メソッド Create あるいは CreateFmt は、Exception 型に用意されているものである。

リスト 8.3 においては、例外の型 EObj1 と EObj2 が宣言されている。これらの例外は手続き proc の実行において生成される。生成される例外はパラメータ i の値によって決まる。

リスト 8.3　例外の宣言

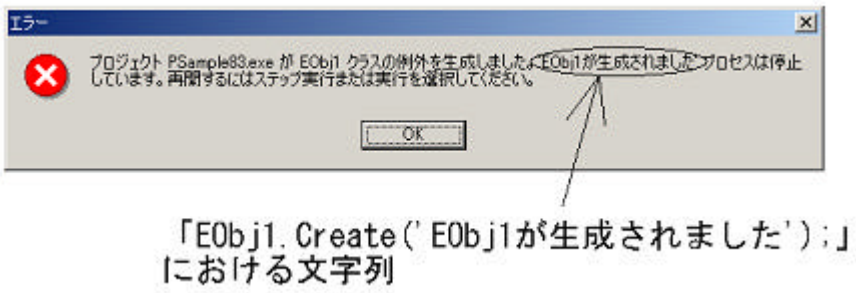
```
type
  EObj1 = class(exception) end;
  EObj2 = class(exception) end;

procedure proc( i : Longint );
  begin
      if i = 1 then
        raise EObj1.Create('Obj1が生成されました')
      else if i = 2 then
        raise EObj2.CreateFmt('i = %d', [i])
      else
        raise Exception.CreateFmt('i = %d @Exception', [i]);
  end;

procedure TForm1.Button1Click(Sender: TObject);
var ck, i : Longint;
begin
    ck:=0;  i:=1;
    repeat
        try
            proc(i);
        except
            on EObj1 do ShowMessage('EObj1が生成されました');
            on EObj2 do ShowMessage('EObj2が生成されました');
            else begin
                ShowMessage('Exceptionが生成されました');
                ck:=1;
            end;
        end;

        i:=i+1;
    until ck <> 0;

    Close;
end;
```

例外が生成されると try..except 文の except～end の部分が実行される。リスト 8.3 の場合、この部分は次の形になっている。

```
except
    on  例外の型  do  文；
      ・
      ・
      ・
    on  例外の型  do  文；
  else  文；  ・・・；  文；
end;
```

生成された例外と同じあるいはその祖先型の例外がonの後に指定されているとき、その「on 例外の型 do 文；」における文が実行される。該当する例外の型がいずれの on の後にも指定されていないときは、else に続く文の並びが実行される。該当する例外の型が何れの on の後にも指定されておらず、かつ else の部分もないときは、その try..except 文の外側の try～文に例外の処理が移されるが、外側の try～文がないときはプログラムの実行終了となる。

サンプルプログラム PSample83.dpr は、上のリスト 8.3 を実行するものである。まず、i の値が１の状態で proc(i)が実行されるので、EObj1 が生成され、次のようなエラーメッセージが表示される。



「ＯＫ」ボタンのクリック後、Delphi のメニュ「実行｜実行」をクリックすると例外の処理を行うために except～end の部分が実行される。例外の型 EObj1 に対応する

```
on EObj1 do ShowMessage('EObj1が生成されました');
```

における ShowMessage が実行され、次のメッセージが表示される。

例外 EObj1 の処理が終わると、i の値が 2 に設定されて proc(i)が実行される。このときは、例外 EObj2 が生成されるので、Delphi によるエラーメッセージ

と、例外の処理によるメッセージ

が順に表示される。

続いて、i の値が 3 に設定されて、proc(i)が実行される。このときは、例外 Exception が生成されるので、Delphi によるエラーメッセージ

と、続いて例外の処理によるメッセージ

が表示される。さらに、このときに ck の値は 1 に設定されるので、repeat～until 文の条

件

$$ck <> 0$$

が満たされて、repeat 文の実行も終了する。

参考文献

（1） Delphi 5オンラインヘルプ、Inprise Corporation, 1999.
（2） Delphi 5 Object Pascal 言語ガイド、インプライズ株式会社、1999.
（3） 岡本安晴「Delphi プログラミング入門」、CQ 出版社、1997.