第八讲 Delphi程序调试及异常处理

吕松茂

lsmtech@163.com

内容提要

- 程序调试
- 异常处理

内容提要

- 程序调试
  - 异常处理

一、程序调试

- 错误种类
- 使用断点
- 设置程序的执行方式
- 监视变量和相关数据的值

一、程序调试

1.1 错误种类

编写程序时所出现的错误一般分为三种：语法错误（Syntax Error）、运行时期错误（Run-Time Error）和逻辑错误（Logic Error）。这些错误都可以通过Delphi开发环境所提供的强大的集成调试器（Integrated Debugger）来找出并修正。

一、程序调试：语法错误

1. 语法错误

语法错误就是在编写程序代码时没有遵守Object Pascal语言的语法规则，在程序编译时，只要有这种错误，Delphi的调试器就会自动找出，并把出错的语句突出显示，同时在代码编辑器的下方给出相应的出错提示。

2. 运行错误

程序语法正确，能正确编译，但程序在执行的过程中发生了错误，这种错误称为运行错误。遇到这种错误，操作系统会自动中止程序的运行，并给出相应提示。

一、程序调试：语法错误

3. 逻辑错误

逻辑错误是最难找的一种错误，表现为程序语法正确，编译运行也没有出现任何异常，但程序运行后产生的结果与编程者所设想的不一样。

一般从以下三点出发，发现程序执行到哪条语句开始出错，从而找出错误根源。

- 猜测出程序可能出错的地方，并在此设置“断点”，
- 让程序执行到“断点”停止运行，观察所有中间变量值及对象内容
- 让程序单步运行，同时观察每一个变量值及对象内容的变化

1.2 使用断点

Delphi的集成调试器可以让用户在程序代码的某些行上设置标记，使程序执行到这些行时暂停执行，这些标记就称为断点。

一、程序调试：使用断点

1. 断点的设定与取消

- 在代码编辑器中，把光标移到某一行上，按下F5键或用鼠标点击该行的左边区域，可以为该行代码设断点。
- 在已设断点的行上重复以上所述设置断点的操作，则取消该行的断点设置。
- 断点必须设置在可执行的代码行上，如果把断点设在注释、空行、变量声明等非执行行上，调试器将认为该断点无效。

一、程序调试：使用断点

2. 设置断点属性

选择执行【View】→【Debug Windows】→【Breakpoints】菜单项，或按下Ctrl+Alt+B组合键，将打开断点列表窗口，窗口中包含了所有断点信息。

❁使断点有效与无效

在断点列表窗口中用鼠标右键单击所选的断点，在弹出菜单中选取【Enable】项，使其左方的“√”出现（断点有效）或消失（断点无效）。

一、程序调试：使用断点

- 设置断点条件

在断点列表窗口中右键单击所选的断点，在弹出菜单中选取【Properties】项，在弹出的源断点属性窗口上做出相应设置。

1.3 设置程序的执行方式

1. 单步执行

单步执行就是一次执行一行语句，当程序遇到断点暂停后，可以选择这种方式跟踪程序的执行。通过选择执行【Run】→【Step Over】菜单项，或按下F8键实现程序单步执行。

一、程序调试：设置程序的执行方式

2. 跟踪执行

跟踪执行和单步执行类似，只是当执行到含有过程或函数调用的行后，执行点将进入过程或函数内部。跟踪执行程序的方法是选择执行【Run】→【Trace Into】菜单项，或按下F7键。

3. 执行到光标所在处

如果希望程序在没有设置断点的行上暂停，可以把光标停在这行上，选择执行【Run】→【Run to Cursor】菜单项，或按下F4键，程序就会直接执行到光标所在行上，然后暂停，等待用户作进一步操作。

1.4 监视变量和相关数据的值

1. 提示文本

当程序暂停运行后，在代码编辑窗口中把鼠标移到有关变量上，这时集成调试器就会把该变量的值以一个提示文本（Hint）的方式显示出来。

一、程序调试：监视变量和相关数据的值

2. Watch List窗口

在调试程序的过程中，还可以利用Watch List窗口监视多个变量或表达式的值。选择执行【View】→【Debug Windows】→【Watches】菜单项，或按Ctrl+Alt+W组合键，打开Watch List窗口。在Watch List窗口中添加需要监视的对象。

一、程序调试：监视变量和相关数据的值

3. Evaluate/Modify窗口

使用Evaluate/Modify窗口查看或修改数据的步骤如下：

❁选择执行【Run】→【Evaluate/Modify…】菜单项，或按下Ctrl+F7组合键打开如图5-13所示的Evaluate/Modify窗口；

❁在窗口的Expression编辑框输入变量名、对象属性或表达式；

❁选取Evaluate按钮，则Expression中数据项的当前值就显示在Result框中；

❁在New Value编辑框中输入新的值，选取Modify按钮，则Expression中变量的值就被改成新输入的值。

内容提要

- 程序调试
- 异常处理

二、异常保护及处理

- 异常的概念及异常类
- 异常保护和处理
- 自定义异常类及其应用

二、异常保护及处理：异常的概念及异常类

程序运行期间产生的不可预料的错误称为异常，它干扰正常的程序流程。存在许多可能导致异常发生的情形，例如，内存申请失败，浮点运算的溢出，文件I/O的各种异常，以及打印异常等等。

二、异常保护及处理：异常的概念及异常类

在SysUtils单元中定义的RTL异常类和在Controls单元中定义的VCL异常类，都由Exception类派生而来，在Exception类中定义了进行异常处理的基本属性和方法。这些异常类的定义，一方面归纳总结了Delphi应用程序可能出现的异常，另一方面对异常进行了内置的保护。

二、异常保护及处理：异常保护和处理

Delphi提供了异常保护和处理的程序控制结构。在保留字try与end之间的一段代码称为保护块。在Delphi中，异常是一个对象，它包含了异常的原因、位置和类型的信息。当保护块内程序的运行发生异常时，将自动创建一个相应的异常类对象，程序可以捕获这个异常类对象，以确保程序的正常结束和资源的释放，并可对异常做出响应，或调用系统的默认处理过程。

二、异常保护及处理：异常保护和处理

1. 异常保护与try ... finally ... end;语句

该语句的格式如下：

```
try
  ..... // 被保护的代码块
finally
  ..... // 处理语句
end;
```

二、异常保护及处理：异常保护和处理

2. 响应异常和try ... except ... end;语句

该语句结构提供了一个可以根据需要进行自定的异常处理的机制。其一般格式如下：

二、异常保护及处理：异常保护和处理

```
try
  // 以下为保护代码块
  if <异常条件> raise <异常对象>
except
// 以下为异常处理块
  on <异常类1> do <处理过程1或语句1>
// 捕获异常为异常类1的处理
  on <异常类2> do <处理过程2或语句2>
 // 捕获异常为异常类2的处理
  on ……
  else <其他处理过程或语句>    // 该子句可以缺省
end;
```

二、异常保护及处理：自定义异常类及其应用

通过继承类exception，可以自定义新的异常类。

定义一个新异常的形式如下：

```
type
  异常类名 = class(Exception);
```

或者如下：

```
type
  异常类名 = class(Exception)
    类成员   // 数据域或方法
    ...
  end;
```

总结

- 程序调试
- 异常处理

谢谢大家