Blog

Entries

Summary

Listed by:

Date

February 2010

January 2010

December 2009

November 2009

October 2009

September 2009

August 2009

July 2009

June 2009

May 2009

April 2009

March 2009

February 2009

January 2009

December 2008

November 2008

October 2008

September 2008

August 2008

< Previous Next >

August 25

Delphi 2009测试版的新语法特性

Delphi 2009测试版的新语法特性

结束了对到Delphi 2007的语法回顾，终于正式进入到最激动人心的2009新语法部分了。

String的变化与增强了的Exit

为全面支持Unicode，Delphi 2009中的所有跟String相关的部分都由原来的AnsiString变成了UnicodeString。这自然也就意味着，原来一些用String声明的函数现在可能会有一些问题（好在我都不厌其烦的用AnsiString和WideString）。这同时意味着Char已经是WideChar而不再是AnsiChar、PChar也是PWideChar而不再是PAnsiChar了。在使用D2009编程时一定要时刻小心和注意这个地方。
而Exit则变成了类似C的return的作用，不过退出参数是可选的，这样才能兼容以前的代码和Result关键字。虽然说这是一个小改进，但是减少了每次不厌其烦的

```
  if(not True)then
  begin
    Result  := nil;
    Exit;
  end;
```

而只需

```
  if(not True)then Exit(nil);
```

即可了。

匿名方法引用（reference to）

以前我们创建一个方法引用的时候会很麻烦，尤其是在类中，需要跳出去在别的地方写一段函数，然后再回来继续写。新的语法reference to避免了这种情况的发生，尤其是许多时候其实我们的方法实际上只有一两句的时候，它可以大大加快开发的速度，就像前面的Exit语法加强一样贴心。不过遗憾的是，这个类lamda方法的语法糖还不够甜。

```
type
  TFoo = reference to function(num: Integer): Integer; // reference to Method
var
  func: TFoo;
  n: Integer;
begin
  func := function(a: Integer): Integer // *NOTE*: Do NOT Write ';'
  begin
    Result := a * a;
  end;
```

July 2008
June 2008
May 2008
April 2008
March 2008
February 2008
January 2008
December 2007
November 2007
October 2007
September 2007
August 2007
July 2007
June 2007
May 2007
April 2007
March 2007
February 2007
January 2007
December 2006
November 2006
October 2006
September 2006
August 2006
July 2006
June 2006
May 2006

```
n := func(10);
```

增强了的反射单元ObjAuto

这个是RTTI里的，按说不算语法上的更新，但涉及到编译器，又在RTTI方面非常有用，所以这里我还是把它拿出来说了。看过李维的《Inside VCL》的应该都知道TypInfo单元在RTTI中的重要性，而ObjAuto可以说是TypInfo的增强版。
先来点儿预备知识：一般情况下，class声明中的默认区域是public，但TPersistent指定了{$M+}编译器参数，使得包括其继承类（所有VCL控件）的默认区域都成为了published。以前我们可以通过TypInfo获取published区域的方法信息，成为了许多控件自动化功能的重要组成部分。而在2009中又增加了{$METHODINFO ON}/ {$METHODINFO OFF}编译器选项，使ObjAuto单元能够获取public区域中的方法信息。具体的示例请看这个链接：http://www.cnblogs.com/del/archive/2008/08/16/1269359.html

泛型

终于到了最激动人心、让人欢喜让人忧的泛型了。大家都知道C++难，异常难，用了十几二十年的人严谨点儿的话也不敢说自己非常懂C++。除了各种cast、实例的创建外，操作符的重载、泛型的支持使得C++变成一个自由度极高的语言，而复杂度也因此上了一个量级。有人说C++的精华就在于Templates，也就是泛型的支持，以我对C++的浅见，窃以为此话还是有一定道理的。在Delphi引入泛型的支持后（还好Delphi的操作符是关键字比较多，不适合重载），许多人担心的是，编译速度会不会变慢。在用了D2009beta版之后，至少目前几个测试项目还体会不到速度的明显降低，好在Delphi里没有C/C++那么复杂的预编译宏。不过就我目前的测试来看，Delphi 2009对泛型的支持还不是那么好，比如class<T>中内嵌type定义record中如果使用了类型为T的泛型成员的话，编译是会挂掉的。虽说有还不算太麻烦的用class代替并且不用默认方式的构架/析构过程的trick来避免这一问题（直接用class模拟record会导致如泛型包含string之类时会发生leak），但不够强大的泛型支持还是比较麻烦的事。
泛型的语法并不复杂，类似C++，在元素名称后使用“<>”包围泛型列表，如“Arr<T>=Array of T”“<TA, TB>”等，在其对应的作用域中可见。可以使用泛型的只有两种情况，一种是新定义一种类型的时候，这个类型可以是record、class、interface、array等，另外一种情况是一个class的方法、类方法可以单独定义泛型，但普通函数/过程不能定义泛型。这里就不具体举例了，有兴趣的可以参考Delphi 2009中自带的Generics.Collections单元中的几个基本类型。

其他

由于对Unicode支持是一个大局大变化，所以由此带来的Windows单元、VCL组件单元带来的变化也不小，其他相关的支持单元也加入了许多元素和进行了一些较大的调整。比如新加入的TEncoding和对TStrings、TStream系列的增强，以及TStringBuilder这种.Net中的概念元素等。还有一些变化使得以前版本的控件无法很方便的移植到D2009中，而Delphi如果缺少第三方组件的支持，可以说吸引力会大打折扣。
对于IDE等方面的变化和VCL控件方面的改进等这里就不介绍了。总之，非常期待正式版的出现，这是Delphi近年来最值得期待的一个版本。同时，Delphi 2009正式版中对泛型的支持我仍然要打一个问号，不知能否改进对使用泛型的class内部record的支持，或者出于编译速度的考虑无法达到十全十美，但愿今天的CodeGear会有一个更令人满意的答复。

Back: 引子

9:58 PM | Blog it | 计算机与 Internet

Comments

Trackbacks

The trackback URL for this entry is:
http://egust.spaces.live.com/blog/cns!98563909D0913C04!549.trak
Weblogs that reference this entry

- None