

[主页](#) [博客](#) [相册](#) | [个人档案](#) | [好友](#)

[查看文章](#)

Delphi2010强化的反射

2009-09-02 13:18

很多人可能都发现了，**Delphi2010**编译后的程序体积非常大，一个空的窗体居然达到接近**800K**。这些多出来的体积其实就是运行时信息，使用这些信息，我们可以轻松的获取到有关的类内容，并在不知情的情况下使用它们。这个特性对于实现多层的架构有着非同一般的意义。

Delphi2010提供了一个**Rtti**单元，用于实现运行时信息的**Get, Set**。

在其中能找到一个名为**TRttiContext**的**Record**，它就是本文的主角了。**TRttiContext**提供了一个静态的构造方法，也就是说，我们无需手工的创建它。新建一个窗体，然后在**public**下写入 **Ref: TRttiContext;**即可完成声明，以后就能直接使用了。（此处的**Ref**可以按需求修改成任何的合法变量名）

在此我写了一个类，用来做测试，如下（**Delphi2010**已不再要求用**{\$MethodInfo}**来标识用于反射的信息）

```

unit untTest;

interface

uses
  Classes, SysUtils, Dialogs;

type
  TTestClass = class
  private
    FX: Integer;
    FY: Integer;
    FStr: string;
    function GetXY: Integer;
  public
    function DoAdd(x,y: Integer): Integer;
    procedure ShowMsgBox;
  public
    property X: Integer read FX write FX;
    property Y: Integer read FY write FY;
    property STR: string read FStr write FStr;
    property XAY: Integer read GetXY;
  end;

implementation

function TTestClass.DoAdd(x, y: Integer): Integer;
begin
  Result := x + y;
end;

function TTestClass.GetXY: Integer;
begin
  Result := FX + FY;
end;

procedure TTestClass.ShowMsgBox;
begin
  ShowMessage('Test Call');
end;

end.

```

接着就是用代码来获取类的信息了，我在窗体上放了一个TMemo并命名为mm，用于输出信息。如下：

```

procedure TFormMain.ReflectClass(AClass: TRttiInstanceType);
var
  fields: TArray<TRttiField>;
  field: TRttiField;
  methods: TArray<TRttiMethod>;
  method: TRttiMethod;
  params: TArray<TRttiParameter>;
  param: TRttiParameter;
  props: TArray<TRttiProperty>;
  prop: TRttiProperty;
  s: string;
begin
  mm.Lines.Add(AClass.Name);
  fields := AClass.GetFields;
  methods := AClass.GetMethods;
  props := AClass.GetProperties;
  mm.Lines.Add('Fields:');
  for field in fields do
    mm.Lines.Add(Format('    %s: %s;', [field.Name, field.FieldType.Name]));
  mm.Lines.Add('Methods:');
  for method in methods do
    begin
      params := method.GetParameters;
      s := EmptyStr;
      for param in params do
        s := s + Format('%s, ', [param.ToString]);
      s := Copy(s, 1, Length(s) - 2);
      if method.ReturnType = nil then
        s := Format('    procedure %s(%s);', [method.Name, s])
      else
        s := Format('    function %s(%s): %s;', [method.Name, s, method.ReturnType.Name]);
      mm.Lines.Add(s);
    end;
  mm.Lines.Add('Properties:');
  for prop in props do
    begin
      s := EmptyStr;
      if (prop.IsReadable) and (not prop.IsWritable) then
        s := 'ReadOnly';
      if (not prop.IsReadable) and (prop.IsWritable) then
        s := 'WriteOnly';
      mm.Lines.Add(Format('    %s: %s; %s', [prop.Name, prop.PropertyType.Name, s]));
    end;
  end;
end;

```

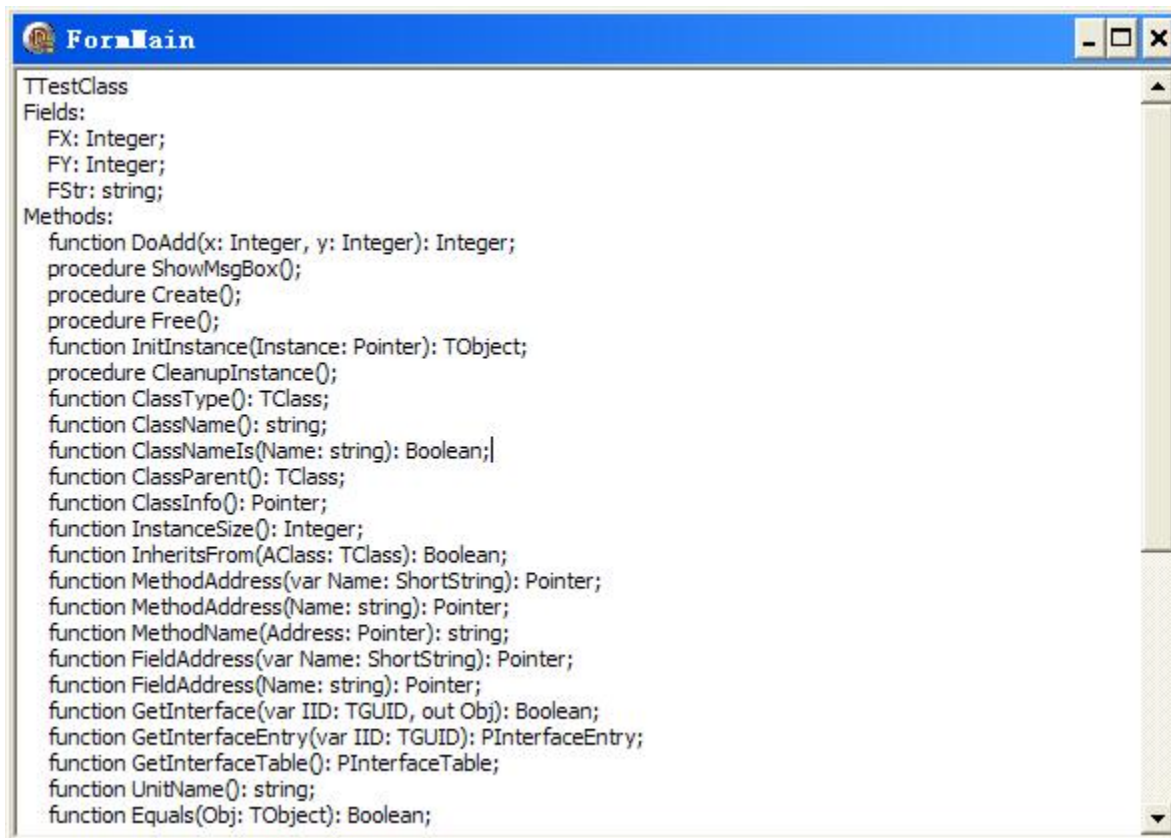
这是一个很通用的方法，对于任意的类都能调用这个方法显示它的有关信息。

在窗体的Create事件中，再写一点调用的代码：

```
procedure TFormMain.FormCreate(Sender: TObject);
var
    typ: TRttiType;
begin
    typ := Ref.GetType(TTestClass);
    ReflectClass(typ.AsInstance);
end;
```

到此为止我们先运行一下程序，看看效果，如图所示，TTestClass的信息已经全部输出了。

有人可能会问，为什么只写了两个方法，反射却得到那么多方法呢？那是因为反射机制会同时得到父类的内容。



注意到什么了吗？那三个Field，代码中是private下的，居然拿到了！而同在private下的GetXY()方法却没有得到。暂时还无法确定是否反射单元的bug，或是原本就如此设计。总之现在，我们该拿到的东西都拿到了，不该拿到的也有一部分拿到了。

好了，上面的部分只是“显示”出一个类的信息，对于反射机制来说，只是显示是完全不够的，还要能够调用。很幸运的是，Delphi2010不仅在获取上做了改进，而且调用也方便了很多。

以下的代码是调用TTestClass内的DoAdd() 方法:

```
procedure TFormMain.Button1Click(Sender: TObject);
var
    t: TTestClass;
    typ: TRttiType;
    mthd: TRttiMethod;
    value: TValue;
begin
    t := TTestClass.Create;
    typ := Ref.GetType(TTestClass);
    mthd := typ.GetMethod('DoAdd');
    value := mthd.Invoke(t, [1,2]);
    ShowMessage(IntToStr(value.AsInteger));
    t.Free;
end;
```

再来一个给成员变量赋值的代码:

```
procedure TFormMain.Button3Click(Sender: TObject);
var
    t: TTestClass;
    typ: TRttiType;
    fld: TRttiField;
    prop: TRttiProperty;
    value: TValue;
begin
    t := TTestClass.Create;
    typ := Ref.GetType(TTestClass);
    fld := typ.GetField('FX');
    fld.SetValue(t, 1);
    fld := typ.GetField('FY');
    fld.SetValue(t, 2);
    prop := typ.GetProperty('XAY');
    value := prop.GetValue(t);
    ShowMessage(IntToStr(value.AsInteger));
    t.Free;
end;
```

是不是觉得反射用起来很方便呢? 不仅仅是类, **Record**, **Set**, 有序类型和一些其他的東西在Delphi2010下也都拥有运行时信息, 可以被动态获取到。官方提供了一个名为rtti_browser的演示程序, 随Delphi2010分发给了最终用户, 有兴趣也可以参考一下。

最近读者:



[登录后](#), 您就
出现在这里。



[xmhelp](#)



[cpoor](#)



[ztf86781163](#)



[andyfurong](#)



[jakezon](#)



[美是简单](#)



[流星无语吧](#)



[wr96020](#)

[4](#)

网友评论:

1 网友:[lazarus](#) 2009-09-02 14:03 | [回复](#)
好文章。就是不贴代码贴图片的方式有点贰。

2  2009-09-02 14:11 | [回复](#)
我也没有办法, 百度对delphi代码的支持不好, 会吃掉格式, 还是图片看着清晰些
[rarnu](#)

3 网友:偶是马
甲 2009-09-02 14:38 | [回复](#)
支持啊, 盼星星盼月亮终于盼到了RTTI增强了

4  2009-09-02 15:27 | [回复](#)
[g62100299](#) 橙子排版的挺好,很负责 🍊

5 网友:猪头没 2009-09-03 09:30 | [回复](#)
橙子, Method和property默认不生成private和protected

```
{ These constants represent the default settings built into the compiler.  
  For classes, these settings are normally inherited from TObject. }  
DefaultMethodRttiVisibility = [vcPublic, vcPublished];  
DefaultFieldRttiVisibility = [vcPrivate..vcPublished];  
DefaultPropertyRttiVisibility = [vcPublic, vcPublished];
```


-
- 6  [rarnu](#) 2009-09-03 11:01 | [回复](#)
是的, 已经看了, 可以用以下方式开启对**method**和**property**的**RTTI**信息记录
`{$RTTI EXPLICIT METHODS([vcPrivate..vcPublished]) PROPERTIES([vcPrivate..vcPublished])
FIELDS([vcPrivate..vcPublished])}`
-
- 7 网友:偶是马甲 2009-09-03 12:56 | [回复](#)
这么说, 就是那些讨厌程序变大的人, 也可以用这个方法让程序变小了
-
- 8 [匿名网友](#) 2009-09-04 12:01 | [回复](#)
`type := Ref.GetType(TTestClass);`
既然都可以得到**TTestClass**了, 为什么还要使用反射?
能否通过“**TTestClass**”名字来反射出类??
-
- 9  [rarnu](#) 2009-09-04 13:29 | [回复](#)
如此写只是为了实现一个**Demo**, 你可以设想从**Server**传来一个**Client**没有的类
当然了, 以此情况下**Client**接收到的只不过是**TObject**, 但是你必须从这个**TObject**获取到该类原本的真实信息。

通过名字得到类也是可以的, 有一个**FindType()**方法, 即要求传入一个字符串的类名, 并返回**RttiType**
-
- 10  [drroc](#) 2009-09-04 17:46 | [回复](#)
橙子能讲一下**lazarus**的**rtti**吗? **delphi 2010**的基本了解了, 很方便


我想在**lazarus**实现上面那种调用**DoAdd**的写法
-
- 12  [rarnu](#) 2009-09-04 18:18 | [回复](#)
Lazarus似乎还不能很好的支持**RTTI/Reflect**
参考其**wiki**, 它只能够存取**published**下的属性, 对于方法或是其他区分符下的属性, 成员都无能为力
wiki上也只有只字片语, **RTTI**被一言带过了
http://wiki.lazarus.freepascal.org/RTTI_controls
-
- 13 网友:Delphi Fans 2009-09-04 18:23 | [回复](#)
lazarus的**rtti**很弱的, 不支持这样做
-

14 网友:Delphi Fans 2009-09-04 18:23 | [回复](#)
原来橙子已经回复过了，那我路过~

15 [匿名网友](#) 2009-09-07 09:27 | [回复](#)
Rtti单元,我引用不到。能告诉我路径吗?

16 
[rarnu](#) 2009-09-07 09:32 | [回复](#)
在D2010安装路径下，lib目录内，有一个rtti.dcu

17 [匿名网友](#) 2009-09-07 09:39 | [回复](#)
.dcu的能直接引用吗???

18 
[rarnu](#) 2009-09-07 09:59 | [回复](#)
当然可以，其实delphi只引用dcu，如果你是pas，编译后也是先生成dcu再联编的

19 [匿名网友](#) 2009-09-07 10:06 | [回复](#)
回复rarnu：这个方法还没学会，能教一教吗?

20 [匿名网友](#) 2009-09-07 10:10 | [回复](#)
C++Builder2010不能用rtti?

21 [匿名网友](#) 2009-09-07 10:26 | [回复](#)
回复rarnu：在DELPHI2010中，要怎么引用.DCU呢？具体方法能说下吗?

22 
[rarnu](#) 2009-09-07 10:43 | [回复](#)
直接uses不就好了，没啥方法呀

- 23

匿名网友

2009-09-07 13:59 | [回复](#)
我直接USES了，认不到。
- 24

匿名网友

2009-09-07 14:45 | [回复](#)
DELPHI2010中有没有类似C#中的Activator.CreateInstance(a,bb)的方法，能创建对象的？不然还是不够灵活。
- 25


[rarnu](#)

2009-09-07 17:08 | [回复](#)
很早就有了，可以参考我以前写的DHibernate
在CnPack组件包里有
- 26

匿名网友

2009-09-08 11:39 | [回复](#)
回复rarnu：能给个例子吗？就是创建对象的方法，不用CREATE来创建的。我的邮箱：shijq19830923@163.com
- 27


[andyfurong](#)

2009-09-15 00:02 | [回复](#)
橙子兄：照这样的话，那么软件的安全性是否就会相应的除低了？

发表评论：

姓 名：

[注册](#) | [登录](#)

网址或邮箱：(选填)

:

内 容：

插入表情

验证码:

请点击后输入四位验证码，字母不区分大小写

©2009 Baidu