

专注IT软件技术！别人放弃我坚持！

碧水航工作室

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

[查看文章](#)

D2010 RTTI + Attribute 简单实现ORM

2009-11-30 23:34

还记得**David I** 今年四月来盛大时，被问及“反射机制能再做得好一点吗？我们想放弃RTTI”，**David I** 回答“这的确是需要考虑的地方，当然RTTI我们不会放弃的”。（这个白胡子的老哥哥还真很可爱，当年Borland几经起落，唯一能看得顺眼的就是**David I**）。我还以为RTTI在D2010最多只是改良，炒冷饭而已。没想到，RTTI不仅能反射Public、protected、Private里的信息，还能动态执行该类里的方法，更惊奇的是，还支持Attribute。D2010 New RTTI 在某种程度上，比肩Unicode，在扩展框架上有无限的遐想空间。下面说一下 D2010 RTTI + Attribute 简单实现ORM。

1、支持ORM，最基础的两个信息是表的信息和字段信息。这两个信息，如果用Attribute 来辅助，代码更简洁和可读性更好。可以把属性名当做真实字段名，也可以将特性里的属性当成真实姓名，再加上字段标题（可以当成注释）、必填字段、是否为主键、显示格式等等，如果没有Attribute ，类、属性的辅助信息必须用其他信息来描述，非常麻烦。

```
uses
SysUtils, RTTI, TypInfo, Types;

type
Table = class(TCustomAttribute)
private
```

```

    FName: string;
    FTitle: string;
published
public
    constructor Create(ATableName, ATitle: string);
    property Name: string read FName write FName;
    property Title: string read FTitle write FTitle;
end;
```

2、有了这两个Attribute，我们必须创建一个解析属性和Attribute的类，并且能解析Insert、update、delete、select等SQL语句。我们姑且叫TStorable。这个类可以根据需要扩展你所想要的东西。目前只实现了Insert方法，其他的方法，留给勤奋的人去遐想。

```
public
//插入SQL语句
function Insert: string;
//获取字段标题
function GetFieldTitle(const AFieldName: string): string;
//设置
//function SetAttributeValue(const PropName, AttributeValue: string): Boolean;
end;
```

```

        Break;
    end;
end;
end;
finally
    Context.Free;
end;
end;

function TStorable.Insert: string;
var
    Context:TRttiContext;
    Prop:TRttiProperty;
    typ:TRttiType;
    A1,A2:TCustomAttribute;
    Sqls,Fields,Values,Value:string;

begin
    Context := TRttiContext.Create;
    try
        Sqls := "";
        Fields := "";
        Values := "";

        typ := Context.GetType(ClassType);
        for A1 in typ.GetAttributes do
            begin
                if A1 is Table then
                    begin
                        Sqls := 'Insert Into '+Table(A1).Name; //获取Insert表名
                        for Prop in typ.GetProperties do
                            begin
                                for A2 in Prop.GetAttributes do
                                    begin
                                        if A2 is FieldInfo then    //AHa
                                            begin
                                                Fields := Fields + ','+ FieldInfo(A2).FieldName ;
                                                // the value of the attribute
                                                Value := Prop.GetValue(Self).ToString;
                                                //根据数据类型对属性值加以编辑
                                                case Prop.GetValue(Self).Kind of
                                                    tkString, tkChar, tkWChar, tkWString, tkUString:
                                                        Value := QuotedStr(Value);
                                                    tkInteger, tkInt64, tkFloat:
                                                        Value := Value;
                                                    else
                                                        Value := QuotedStr(Value);
                                                end;
                                                Values := Values + ',' + Value ;
                                            end;
                                        end;
                                    end;
                                end; //for A2 in Prop.GetAttributes
                            end;
                        end; //enf of for Prop
                        Delete(Fields,1,1);
                    end;
                end;
            end;
        end;
    except
        Context.Free;
    end;
end;

```

```

        Delete(Values,1,1);

        Sqls := Sqls + ' (' + Fields + ') VALUES (' + Values + ');';

        Result := Sqls;

        end; //if A1 is Table then
    end; //for A1 in typ.GetAttributes do

finally
    Context.Free;
end;
end;
```

```

constructor FieldInfo.Create(AFieldName, ATitle: string);
begin
    FFieldName := AFieldName;
    FTitle := ATitle;
end;
```

3、有了上面的解析类和SQL基础，我们必须创建一个实体类。属性名是否为中文，可以有不同的说法。偶目前栖身在一个医疗行业公司，医疗专业英语术语又臭又长，奥巴马未必能拼写出几个术语。如果用属性名用中文描述，将其真实的字段名放在**Attribute** 里，或许更能提高程序的可读性和维护性。

```

unit uContact;

interface
uses SysUtils,uAttribute;

type
[Table('CONTACTS','联系人信息')]
TContact = class(TStorable)
private
    FName: string;
    FAge: integer;
    F电话: string;
published
public
    [FieldInfo('NAME','名称')]
    property Name: string read FName write FName;
    [FieldInfo('AGE','年龄')]
    property Age: integer read FAge write FAge;
    [FieldInfo('电话','联系电话')]
    property 电话:string read F电话 write F电话; //尝试一下中文字段名，习惯就好
end;
implementation

end.
```

4、调用示例就很简单了：

```

procedure TForm4.btn1Click(Sender: TObject);
var
    Contact:TContact;
begin
```

```
Contact := TContact.Create;  
Contact.Age := 32;  
Contact.Name := 'TinTin';  
Contact.电话 := '135*****918';//你还会记得918的屈辱吗?
```

```
ShowMessage(Contact.Insert);
```

```
ShowMessage(Contact.GetFieldTitle('Age'));  
Contact.Free;  
end;
```

5、综述：

ORM确实在对象映射上使用起来非常方便，但并非万能，如果过分依赖于ORM，不仅不能了解数据库表与业务的关系，而且还容易写出低效的SQL查询语句。Update语句，须谨记，字段值变化才去更改，否则，会增加数据库的数据不一致风险及其增加数据库日志开销。Delete语句，配合有关键字信息的Attribute，必要时，还要校验是否影响单条或多条记录。

这只是一个简单的例子，离真正的生产力还差一步，为了执行SQL语句，你可以在TStorable 实现数据集的读写，然后才调用执行SQL语句。



类别：delphi技术 | 添加到收藏 | 浏览(250) | 评论 (4)

上一篇：D2010 New RTTI 妙解 Xml DataBi...

最近读者：



[登录](#)后，您就出现在这里。



网友评论:

1



[rarnu](#)

2009-12-01 10:10 | [回复](#)

不介意我转个帖吧?

2



[TinTinSoft](#)

2009-12-01 10:44 | [回复](#)

任何人都可以，橙子转我的文章更是我的荣幸。

3

网友:偶是马
甲

2009-12-01 11:00 | [回复](#)

2010居然连attribute都有了。。。

4



[hacklxx](#)

2009-12-02 11:12 | [回复](#)

cool~

2010, good!

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