ATTRIBUTES AND RTTI

CODE FASTER!

ATTRIBUTES

"Attributes are a language feature in Delphi that allows annotating types and type members with special objects that carry additional information.

This information can be queried at run time. Attributes extend the normal *Object-Oriented* model with *Aspect-Oriented* elements."

- Delphi documentation

ATTRIBUTES

- Code annotation
  - Programmer annotates the code
- Run time querying
  - Program reads those annotations and reacts accordingly

EXAMPLE

```
type
  TMyClass = class
    [MyAttribute]
    FField: string;
  end;
```

IMPLEMENTATION

- Attribute = class
  - Descending from `TCustomAttribute`
  - Traditionally **not** starting with a T and ending in an `Attribute`

```
type
  MyAttribute = class(TCustomAttribute)
  end;
```

SIMPLIFICATION

- `Attribute` part of the name can be removed in the annotation
  - Standard practice

```
type
  TMyClass = class
    [MyAttribute]
    FField1: string;
    [My]
    FField2: integer;
  end;
```

RTTI

```
ctx := TRttiContext.Create;
try
  t := ctx.GetType(aClass);
  for f in t.GetFields do
    for a in f.GetAttributes do
      …
finally ctx.Free; end;
```

- Attributes exist only when observed!

USEFUL HELPERS

- RTTIUtils
  - Robert Love
- DSharp.Core.Reflection
  - Stefan Glienke

PARAMETERS

- Attributes can accept parameters

```
type
  TMyClass = class
    [StoreAs('ExternalField')]
    FField: string;
  end;
```

CONSTRUCTOR

- Parameters are passed to an appropriate constructor

```
type
  StoreAsAttribute = class(TCustomAttribute)
  public
    constructor Create(externalName: string);
  end;
```

OVERLOADED ATTRIBUTES

- Attribute can accept different parameter types or variable number of parameters

```
type
  TMyClass = class
    [StoreAs('ExternalField1')]
    FField1: string;
    [StoreAs('ExternalField2', true)]
    FField2: string;
  end;
```

OVERLOADED CONSTRUCTORS

```
type
  StoreAsAttribute = class(TCustomAttribute)
  public
    constructor Create(
      externalName: string); overload;
    constructor Create(externalName: string;
      storeAsAttribute: boolean); overload;
  end;
```

DEFAULT VALUES

```
type
  StoreAsAttribute = class(TCustomAttribute)
  public
    constructor Create(externalName: string;
      storeAsAttribute: boolean = false);
  end;
```

ADVANCED ATTRIBUTES

- Multiple attributes are allowed on one element

```
[Serialize]
[RootNode('Options')]
TewOptions = class

[Serialize] [RootNode('Optons')]
TewOptions = class

[Serialize, RootNode('Options')]
TewOptions = class
```

ATTRIBUTABLE ENTITIES

- Attributes can be applied to
  - classes, records
  - fields, properties
  - methods, method parameters
  - non-local enumeration declaration, variable declarations
- Even to entities that are not accessible with RTTI!

PROBLEMS

- Not working on generic classes
- W1025 Unsupported language feature: 'custom attribute‘
- Meaning can be unclear

```
[GpManaged(true)]
FList: TObjectList;
```

PRACTICAL EXAMPLES

AUTOMATICALLY CREATED FIELDS

```
type
  TewOptions = class(TGpManaged)
  strict private
    FActiveLanguage: string;
    [GpManaged] FEditor : TewOpt_Editor;
    [GpManaged] FHotkeys: TewOpt_Hotkeys;
  public
    property ActiveLanguage: string read FActiveLanguage
                                     write FActiveLanguage;
    property Editor: TewOpt_Editor read FEditor;
    property Hotkeys: TewOpt_Hotkeys read FHotkeys;
end;
```

OBJECT SERIALIZATION - XML

```
type
  [XmlRoot('Person')]
  TPerson = class(TObject)
  …
  public
    [XmlAttribute('First_Name')]
    property FirstName: String read FFirstName write FFirstName;
    [XmlElement('LAST_NAME')]
    property LastName: String read FLastName  write FLastName;
    [XmlIgnore]
    property MiddleName: String read FMiddleName write
FMiddleName;
  end;
```

OBJECT SERIALIZATION - INI

```
type
  TConfigSettings = class(TObject)
  …
  public
    [IniValue('Database','ConnectString','')]
    property ConnectString: String read FConnString
                                    write FConnString;
    [IniValue('Logging','Level','0')]
    property LogLevel: Integer read FLogLevel write FLogLevel;
    [IniValue('Logging','Directory','')]
    property LogDirectory: String read FLogDir write FLogDir;
  end;
```

UNIT TESTING - DUNITX

```
type
  TMyExampleTests = class
  public
    [Test]
    [TestCase('Case 1','1,2')]
    [TestCase('Case 2','3,4')]
    [TestCase('Case 3','5,6')]
    procedure TestOne(param1: integer;
                      param2: integer);

    [Test]
    procedure TestTwo;
```

ORM MAPPING

```
type
  [TAttrDBTable('NONE')]
  TReportItem = class(TObject)
  protected
    [TAttrDBField(PP_VEHICLE_FIELD)]
    FVeiculoId: integer;
    [TAttrDBField(PP_DRIVER_FIELD)]
    FMotoristaId: integer;
end;
```

DATA VALIDATION

```
type
  TMyConfig = class
    [MustNotEmptyString] [FolderMustExists]
    property WorkingDir: string;

    [MinValue(1)] [MaxValue(7)]
    property DefaultDay: Integer;

    [MustNotEmptyString][LimitToTheseChars('xyz')]
    property DefaultCity: string;
  end;
```

COMMAND-LINE PARSING

```
type
  TCommandLine = class
  public
    [CLPDescription('Set precision'),
     CLPDefault('3.14')]
    property Precision: string read FPrecision
                              write FPrecision;
    [CLPPosition(1), CLPDescription('Input file'),
     CLPLongName('input_file'), CLPRequired]
    property InputFile: string read FInputFile
                              write FInputFile;
```

REST SERVER

```
[Authorize('User', TNCWAAuthorizer)]
[ExtraHeader('Access-Control-Allow-Headers',
             'sessionId')]
TFilesController= class(TRestPublicController)
  function Put(const fileName,
    [FromBody]content: string): TRestResponse;
  function Save(const fileName,
    [FromBody]content: string): TRestResponse;
```

CONCLUSION

PROS & CONS

+ Big code reduction
+ Self-describing code

- Meaning sometimes unclear
- RTTI can get complicated

LINKS

- Overview
  - http://docwiki.embarcadero.com/RADStudio/XE5/en/Overview_of_Attributes
  - http://www.tindex.net/Language/Attributes.html
  - http://www.thedelphigeek.com/2012/10/multiple-attributes-in-one-line.html
  - http://delphi.fosdal.com/2010/11/another-generics-rtti-bug-attributes.html
  - http://stackoverflow.com/q/6119986
- GpAutoCreate
  - http://www.thedelphigeek.com/2012/10/automagically-creating-object-fields.html
- Serialization
  - http://robstechcorner.blogspot.com/2009/10/xml-serialization-control-via.html
  - http://robstechcorner.blogspot.de/2009/10/ini-persistence-rtti-way.html
- DUnitX
  - https://github.com/VSoftTechnologies/DUnitX
  - http://www.finalbuilder.com/Resources/Blogs/PostId/697/introducing-dunitx.aspx
- ORM mapping
  - http://stackoverflow.com/a/14342203
- Validation
  - http://forum.codecall.net/topic/76628-using-attributes-for-validations/

QUESTIONS?