Microsoft .NET

Using .NET Classes

Introduction

.NET Classes (strictly known as *Types*) are implemented in stand-alone DLLs known as assemblies. Types are typically organised into collections of classes known as *namespaces*. This term is analogous to, but not identical to, the Dyalog namespace.

To create an instance of a .NET Class, you execute an expression of the form:

```
Netclass.New <params>
```

where `Netclass` is the name of the Class.

Note that `.New` is just syntax that tells Dyalog to create a new instance of the specified Class.

`params` refers to whatever arguments the Class requires in order to create an instance. For example, the following expression creates an an instance of the `DateTime` Class:

```
mydt←DateTime.New 2001 4 30
```

Now, because Classes are organised into namespaces and implemented in DLLs, it is necessary to tell Dyalog where to look for them. This is done by the `⎕USING` system variable.

`⎕USING` is a vector of character vectors each of which contains two parts separated by a comma. The first part specified the name of a .NET Namespace, and the second part specifies the name of a DLL.

For example:

```
⎕USING←'System,mscorlib.dll'
```

How it Works

When APL encounters a reference to a name that does not exist (i.e. would otherwise generate a `VALUE ERROR`), and the `⎕USING` system variable is not empty, APL interprets the name to be the name of a .NET Class. Using the standard language interfaces provided by .NET, APL loads the Class, searching the .NET namespaces and DLLs in the order prescribed by `⎕USING`. Once located, the name is then associated with that Class so that subsequent references are very much faster.

Dyalog Ltd
South Barn
Minchens Court
Minchens Lane
Bramley
Hampshire
RG26 5BH
United Kingdom

Phone:
+ 44 (0) 1256 830 030

Fax:
+ 44 (0) 1256 830 031

e-mail:
sales@dyalog.com

Microsoft .NET

Using .NET Classes

The `.New` in the expression is special syntax that tells APL to create an instance of the Class to the left of the dot. APL does this by calling the appropriate *Constructor* for the Class. If the Class provides more than one constructor (this is common), APL matches the number and types of the elements in the argument `params` to the number and types of parameters required by each form of the constructor, and then calls the appropriate constructor.

The result of the expression is reference to a new namespace that represents the newly created instance of the Class.

Properties and Methods

Once you have a created a reference to an instance of a Class, you may access the Properties and Methods exposed by the Class using standard *dot* notation. For example, the DateTime Class has a DayOfWeek property that reports which day of the week the instance represents.

```
      mydt.DayOfWeek
1
```

The DateTime Class provides a IsLeapYear method that reports whether or not a specified year is a leap year. Calling a method is equally simple.

For example:

```
      mydt.IsLeapYear 1999
0
      mydt.IsLeapYear 2000
1
```

Methods behave just like regular APL functions, so the ¨ (each) operator works as expected:

```
      mydt.IsLeapYear¨1999 2000
0 1
```

Dyalog Ltd
South Barn
Minchens Court
Minchens Lane
Bramley
Hampshire
RG26 5BH
United Kingdom

Phone:
+ 44 (0) 1256 830 030

Fax:
+ 44 (0) 1256 830 031

e-mail:
sales@dyalog.com

Microsoft .NET

Using .NET Classes

Language Extensions

Many .NET Classes support a range of common methods that implement various arithmetic and Boolean operations such as *plus*, *minus*, *greater-than*, *equals* and so forth. For example, the DateTime Class provides Boolean methods to compare one DateTime with another. Where these exist, Dyalog calls them through the equivalent APL primitive function.

For example:

```
      DT1←DateTime.New 2001 4 30
      DT2←DateTime.New 2001 1 1

      DT1=DT2  ⍝ Is DT1 equal to DT2 ?
0
      TS←TimeSpan.New 1 1 1
      TS
01:01:01

      DT1
30/04/2001 00:00:00
      DT1+TS   ⍝ What is TS added to DT1 ?
01/05/2002 00:00:00
```

... and **yes**, you can manipulate arrays of instances of .NET Classes like any other APL array.

```
⍋DT1 DT2 DT3  ⍝ Sort an array of 3 DateTime instances
2 1 3
      ⌊/DT1 DT2 DT3 ⍝ Which is the least (earliest)
01/01/2001 00:00:00
```

Dyalog Ltd
South Barn
Minchens Court
Minchens Lane
Bramley
Hampshire
RG26 5BH
United Kingdom

Phone:
+ 44 (0) 1256 830 030

Fax:
+ 44 (0) 1256 830 031

e-mail:
sales@dyalog.com