Worksheet – Dynamic Language Runtime

Lernziele:

Sie kennen die Vor- und Nachteile von dynamic

Sie können dynamische Sprachen in C# mit Hilfe der Dynamic Language Runtime nutzen

Aufgabe 1 – Dynamic typing

Sie finden für die folgende Aufgabe das File WSDynamicStud.zip mit einem VS Projekt auf dem AD. Das Projekt enthält drei leere Methodenrümpfe zur Berechnung der Fakultät einer Zahl, die Sie in den folgenden Aufgaben ausfüllen sollen. Das Programm ruft die jeweiligen Berechnungsmethoden auf und misst die Dauer der Berechnungen.

a) Implementieren Sie die Fakultät-Berechnung schrittweise auf drei verschiedene Arten:

1. Als Methode mit dem Datentyp `long`:

```
long fact(long n)
{
    if (n == 0)
        return 1;
    else
        return n * fact(n - 1L);
}
```

2. Als Methode mit dem Datentyp `dynamic`.

   Wie unter 1, jedoch mit dem Datentyp dynamic.

3. Als Python Skript: Das Skript `pythonDemo.py` ist bereits im Projekt integriert.
   Wichtig: Überprfen Sie, ob die Properties des Fies in VisualStudio bei "Copy to Output Directory" auf „Copy if newer“ gesetzt sind.
   Für die Ausführung müssen Sie zunächst IronPython installieren. IronPython steht unter http://ironpython.net/ zum Download zur Verfügung. Anschliessend fügen Sie ihrem Projekt via NuGet die Packete IronPython und System.CodeDom hinzu.

   Zusätzlich müssen Sie die beiden folgenden Namespaces importieren:

```
IronPython.Hosting
Microsoft.Scripting.Hosting
```

   Mit diesen Zeilen führen Sie das Skript aus:

```
var engine = Python.CreateEngine();
dynamic result = engine.Execute("import pythonDemo\n"
                                + "pythonDemo.factorial(1000)");
```

b) Führen Sie die drei Methoden nacheinander aus und messen Sie die dafür benötigte Zeit. Die Methode EvaluateFactorialCalculation ruft die Berechnungsmethoden auf und misst die Zeit. Was stellen Sie fest? Versuchen Sie das Verhalten zu erklären.

c) Führen Sie nun die Berechnung der Fakultät wiederholt aus (durch Erhöhung der "loops" Variable" auf 10) und messen Sie die Zeiten. Was stellen Sie nun fest? Erklären Sie das Verhalten.

Aufgabe 2 – DynamicObject Implementierung

Hinweis: Diese Aufgabe müssen Sie in einem eigenen .NET Framework 4.7 Projekt lösen. Die Sprachausgabe mit .net Core ist noch nicht lokal möglich.

Erstellen Sie eine neue Console-Applikation (.net C#) mit einer von `DynamicObject` abgeleiteten Klasse `Speaker`, welche dynamisch aufgerufene Methoden so implementiert, dass der Methodenname akustisch vorgelesen wird.
Mit andern Worten: Beim Aufruf einer Methode soll der Methodenname aus dem Lautsprecher tönen. Zum Vorlesen von Text können Sie die Klasse `SpeechSynthesizer` aus dem Assembly `System.Speech.dll` verwenden.

Zusatzaufgabe: Wie müssen Sie die Implementation anpassen, damit folgender Aufruf möglich wird?

```
dynamic speaker = new Speaker();
speaker.Im().Just().Demonstrating().This().Feature();
```