NOKIA Developer

Dynamically executing mathematical formulas on Windows Phone using Jace.NET

This article explains how to use Jace.NET on Windows Phone. Jace.NET is a high performance calculation engine for the .NET platform that can dynamically interpret and execute strings containing mathematical functions.

Introduction



In this article, I will explain **Jace.NET** (https://github.com/pieterderycke/Jace), an OSS framework I have developed in my free time. Jace.NET is a high performance calculation engine for the .NET platform that can dynamically interprete and execute strings containing mathematical functions. These functions can rely on variables. If variables are used, values can be provided for these variables at execution time of the mathematical function. Jace.NET is available for the various .NET flavors: .NET, WinRT, WP7 and WP8.

I will explain the architecture of Jace and how to build a calculator for Windows Phone using it, but the possibilities reach much further (payroll applications, simulation applications, banking and insurance applications).

How does Jace work?

Jace.NET has an architecture similar to the one of modern compilers: interpretation and execution are performed in a number of steps. Each step focuses on one aspect of the parsing and interpretation of the formula. This keeps the overall complexity manageable.



The process starts with the tokenizing phase. During this phase the input string is converted into the various allowed tokens: integers, doubles, operations and variables. If a part of the input string contains text that does not match with any type of token, an exception is thrown and Jace will halt.

When tokenizing is successfully finished, an abstract syntax tree (AST) is constructed. This abstract syntax tree is a tree like data model that unambiguously represents the mathematical formula in memory. All mathematical precedence rules are taking into account when constructing the abstract syntax tree. Jace uses an algorithm inspired by the shunting-yard algorithm of Dijkstra to create this AST.



After AST creation, the optimizer will try to simplify the abstract syntax tree: if a part of the formula does not depend on variables but solely on constants. This part of the tree is already calculated and replaced by a constant in the tree.



The final phase is the OpCode generation. During this phase, a .NET dynamic method is created and the necessary MSIL is generated to execute the formula. This dynamic method is cached in memory. If the same formula is executed again in the future with other values for the variables. The interpretation steps are skipped and the dynamic method is directly executed. If the formulas of the calculations are frequently reoccurring, Jace.NET has near compiled code performance.

Building the calculator

The demo WP8 application we are going to build is a calculator that allows users to freely enter mathematical formulas. It consists of two textboxes (formulaTextbox and resultTextBox) and a button (calculateButton). The screenshot below shots the UI layout:

MY CALCULATOR		1.2) 7:4
Calcu	lator	
Formula:		
	C	
	Calculate	
Result:		

The easiest way to add Jace.NET to the solution is by using NuGet. Right-click on the project and choose "Manage NuGet Packages...".





	DemoCalculator - Manage NuGet Packages	? ×
Installed packages	Include Prerelease * Sort by: Relevance *	jace 🗙 🗸
 Online NuGet official package source Search Results 	Jace.NET Jace.NET is a high performance calculation engine for the .NET platform that can dynamically interprete and exec	Created by: Pieter De Rycke Id: Jace Version: 0.8 Last Published: 2/27/2013
▶ Updates	 A commandline utility accelerator for "Tasks" style utilities. NDatabase - C# Lightweight Object Database C# Lightweight Object Database MoreLINQ's GroupAdjacent for C# Sequences (Source) Enhances LINQ to Objects with the method GroupAdjacent (4 overloads) 	Downloads: 241 View License Terms Project Information Report Abuse Description: Jace.NET is a high performance calculation engine for the .NET platform that can dynamically interprete and execute strings containing mathematical functions. These functions can rely on variables. If variables are used, values can be provided for these variables at execution time of the mathematical function. Dependencies: No Dependencies
Each package is licensed to you by its owner. Microsoft is not responsible for, nor does it grant any licenses to, third-party packages.	1	
Settings		Close

Next we call Jace to execute the formula provided in the formulaTextBox when calculate button gets clicked. Results are displayed in the resultTextBox.

```
public partial class MainPage : PhoneApplicationPage
{
    private readonly CalculationEngine engine;
    public MainPage()
    {
        InitializeComponent();
        this.engine = new CalculationEngine();
    }
    private void calculateButton_Click(object sender, RoutedEventArgs e)
```

	{		Printed or
		resultTextBox.Text = "" + engine.Calculate(formulaTextBox.Text);	
	}		
}			

Now we have a calculator for Windows Phone:

Formula: 4/(7*34+sin(76)) Calculate		LQ1	8:00
Formula: 4/(7*34+sin(76)) Calculate	Calculator		
Formula: 4/(7*34+sin(76)) Calculate Result:	Calculator		
4/(7*34+sin(76)) Calculate	Formula:		
Calculate	4/(7*34+sin(76))		
Result:	Calculate		٦
A 4467669 44999969	Result:		
0.016766841022062	0.016766841022062		

You can also download the source: File:Jace calculator source.zip

Next Steps

The "Calculate" method is the simplest way to call Jace. Jace also supports the creation of Func delegates and also provides a fluent API. For more information please see: https://github.com/pieterderycke/Jace/wiki