Code Contracts for .NET Editor Extensions

Mike Barnett and Daryl Zuniga

# Introduction

The package Code Contracts Editor Extensions is a Visual Studio 2010 editor extension that provides contract support for programmers as they edit code. It is part of the [Code Contracts for .NET](http://msdn.microsoft.com/en-us/devlabs/dd491992.aspx) project available at [DevLabs](http://msdn.microsoft.com/en-us/devlabs/default.aspx). The support consists of three features:

* Tooltip augmentation
* Inheritance adornments
* Metadata files

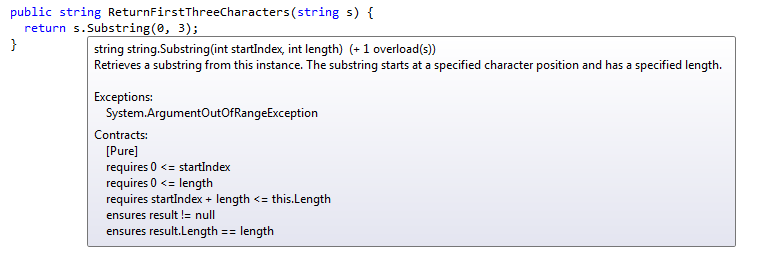
It works within Visual Studio 2010 and currently works only for C# projects. It helps you program by showing you relevant preconditions and postconditions for methods in your code.

This is still under development so please let us know about things you like or don’t like!

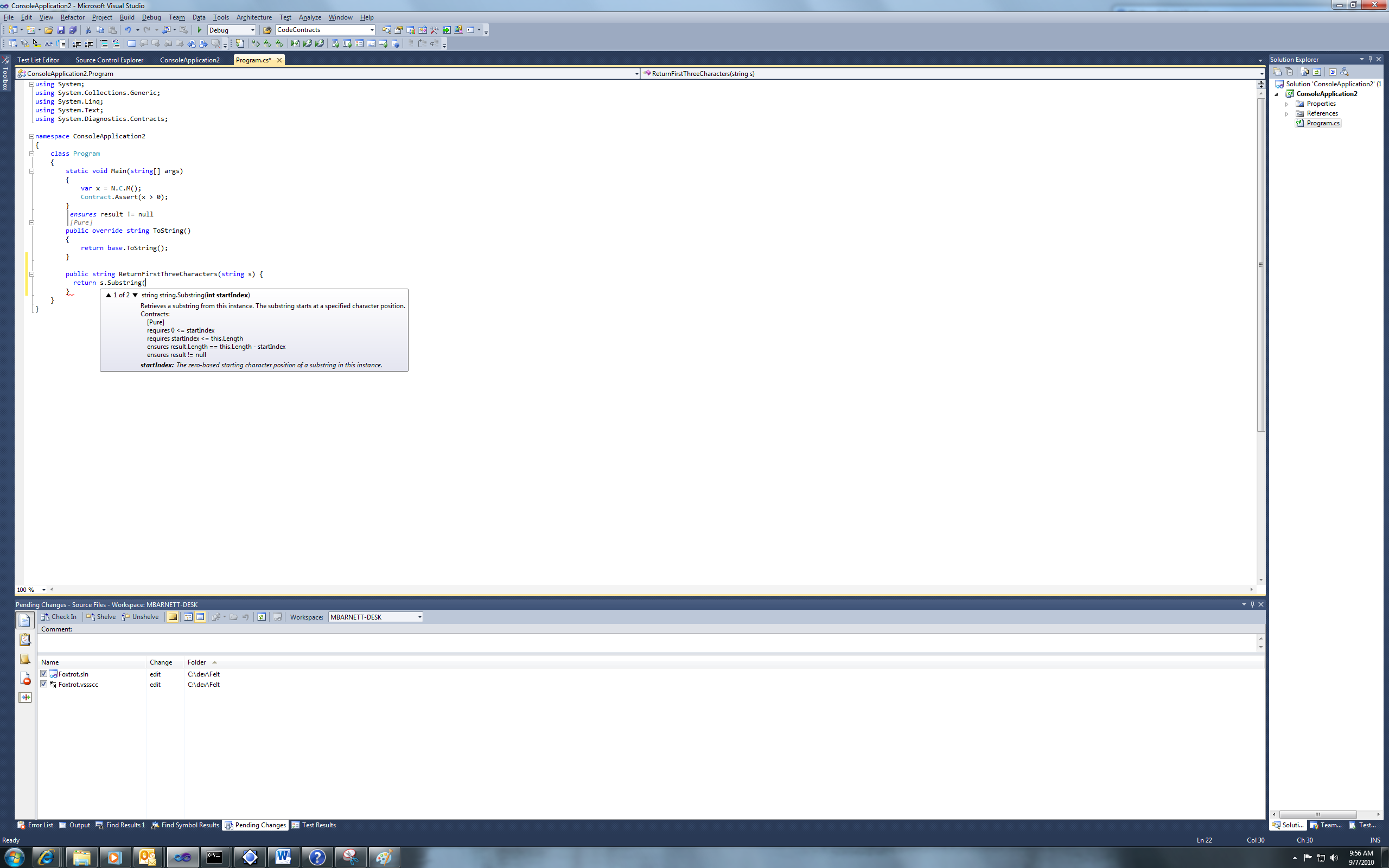
The extension shows the different contracts with keywords (“requires”, “ensures”) instead of the method calls that you use when actually writing contracts (“Contract.Requires”, “Contract.Ensures”). It also replaces the method-call encodings for the return value of a method (“Contract.Result<T>”) with the keyword “result” and the encoding for pre-state values (“Contract.OldValue”) with the keyword “old”.

# Tooltip Augmentation

There are two distinct types of tooltip windows that are modified by the extension. The first is called the “quick info” window. This is what pops up when your mouse hovers over a method call in your code (you have to imagine the mouse is pointing to the call to Substring in the method below):



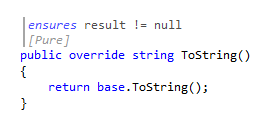
The second type of tooltip window is the “signature helper” window. This is the window that pops up while you are editing a method call and typing its arguments:



If you supply the first argument to the method and type a comma (to begin the second argument), then the contracts should change to match the overload displayed in the window. Please note that there is actually a third type of tooltip window that shows up if you type the method name but do **not** type the open parenthesis and then hover over the method with your mouse. The contracts are **not** added to that type of window.

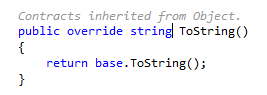
# Inheritance Adornments

Whenever a method inherits a contract, then the extension creates an “adornment” showing the inherited contract. This is a small graphic that may look like text, but there is no text added to your file:



In this case, the method is an override of Object.ToString which has a postcondition saying that the method may not return a null value. Method overrides inherit contracts from the superclass and interface implementations inherit contracts from the interface. Object invariants are not inherited and so never show up in the adornments.

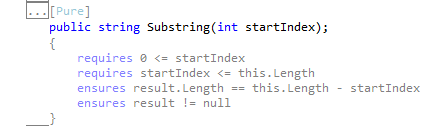
If you want to hide the adornment, then just click on it and it will collapse and replace itself with this:



If you click on it again, it will re-open. If you hover over it with the mouse while it is collapsed, it will bring up a tooltip window that shows the contract.

# Metadata Files

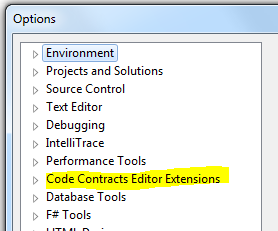
When you “go to definition” for a method/type that is defined in an external assembly (as opposed to an assembly belonging to a different project in your solution), then Visual Studio opens a new window called a “metadata window”. This window is non-editable and contains the type and all of its members. The extensions augment this file with contracts. Here is an example showing the System.String type from mscorlib:



You can turn off the contracts in the window by clicking the button “Hide all contracts” that is in the top-right of the window. Only method contracts are shown: any object invariants are private to the implementation of the type and so are not shown.

# Options

The options for the extensions are found under the Tools|Options menu:



The main option that you might want to turn off is the logging since that can both slow things down and take up a lot of disk space. (If you run into problems with the extension, please turn it back on so that you can send the log with your bug report.)

# FAQ

## I don’t see any of the stuff you’re describing.

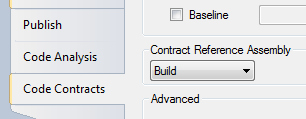
Look in your Output window and select the pull-down for “Code Contracts Editor Extensions”. If you see messages at the bottom about the syntactic or semantic model being out of date, then try quitting Visual Studio and restarting it.

## I see contracts for things from other assemblies, but not from my own projects.

We cannot extract any contracts until you build your project. We do not have the hooks to get the information we need from the background compiler so we do everything based on the post-build binaries.

## I still don’t see contracts from my own projects.

You may not have turned on the building of a contract reference assembly for your project. Please go to the Code Contracts pane in the project properties and make sure that “Build” is selected:



## Which .NET languages are supported?

Currently the extensions work only for C#. We hope to provide support for VB and F# soon.

## I still can’t see anything!

Please use the MSDN [Code Contracts Forum](http://social.msdn.microsoft.com/Forums/en-US/codecontracts/threads/) to report problems. A skilled technician will be with you shortly…

## Can I use the extensions without any of the other Code Contracts tools?

Yes, you actually can install it and use it without installing the [Code Contracts tools](http://msdn.microsoft.com/en-us/devlabs/dd491992.aspx), but then you will definitely not get any contracts for your own code. You will see contracts only for the framework assemblies for which we have contracts.

# Acknowledgements

This extension wouldn’t exist without the help of Manuel Fähndrich, Peli de Halleux, and Herman Venter.