NonNullStack Static Checking Example

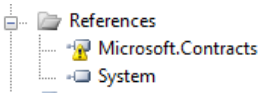
Abstract

This example shows object invariants checking involving ForAll over the elements of the array, and caching of the analysis results to avoid re-analysis. The example is a simple stack containing only non-null elements.

1 Adding the Contract Library Reference

1.1 Visual Studio 2008

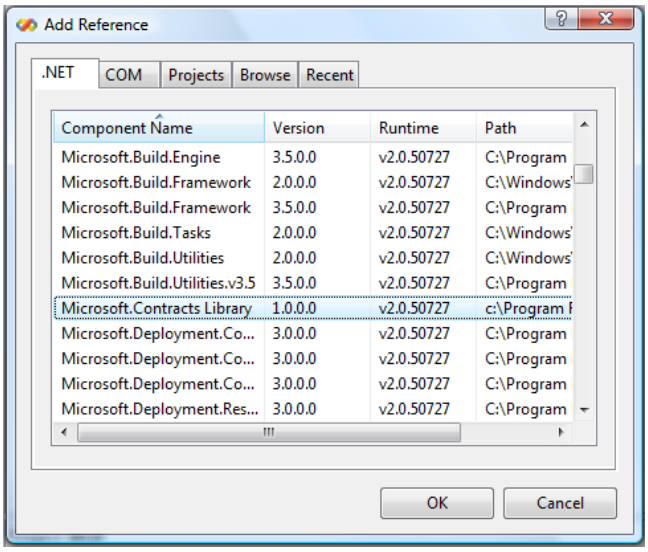
Before getting started with the sample, we need to make sure we have the proper reference for Microsoft.Contracts.dll. Look at the references of project NonNullStack:



If Visual Studio cannot find the Microsoft.Contracts.dll (indicated by the yellow warning), delete the reference and follow the steps below. Otherwise, you are ready to go to Section 2.

For new projects to use contracts, we need to add a reference to the Microsoft.Contracts library to the project.

Assuming you have installed the code contract tools, open the NonNullStack solution and right-click on References in the NonNullStack project and select Add Reference. Find the Microsoft.Contracts library in the .NET tab as shown below and click OK.



1.2 Visual Studio 2010

If you want to try the sample with Visual Studio 2010 you have two choices:

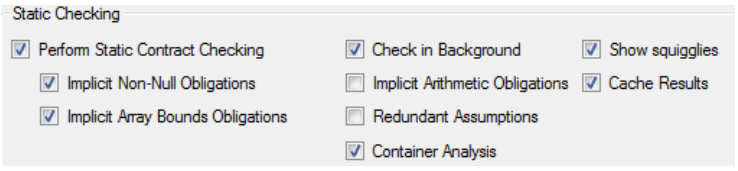
- Target the v3.5 framework (the sample default). In this case, follow the instructions under Visual Studio 2008 as this mode will require the Microsoft.Contracts library to be referenced.
- Target the v4.0 framework. In this case, the contract class is defined in mscorlib and *no* reference to Microsoft.Contracts.dll is necessary.

Thus, to target v4.0 you need to perform the following steps:

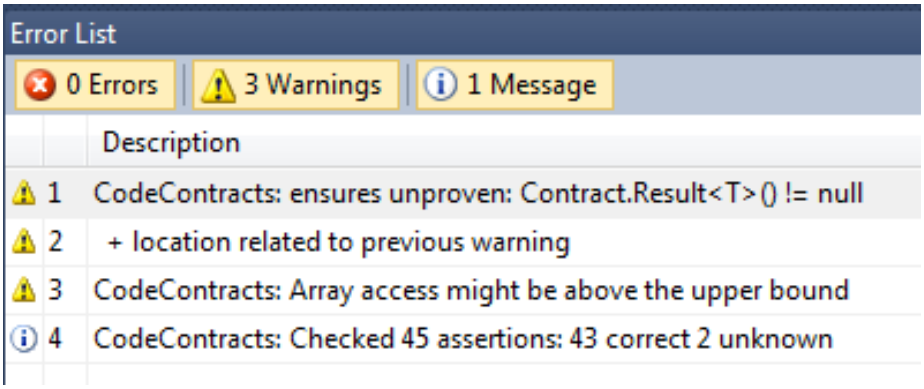
- Delete the Microsoft.Contracts.dll reference from all projects
- Change the target framework of all projects to v4.0.

2 Sample Walkthrough

After adding the proper reference, go to the Properties of project NonNullStack, select the Code Contracts pane (at the bottom), and enable static checking by clicking on the static checking box. Also enable implicit non-null, array checks, container analysis and caching as shown in this screenshot:



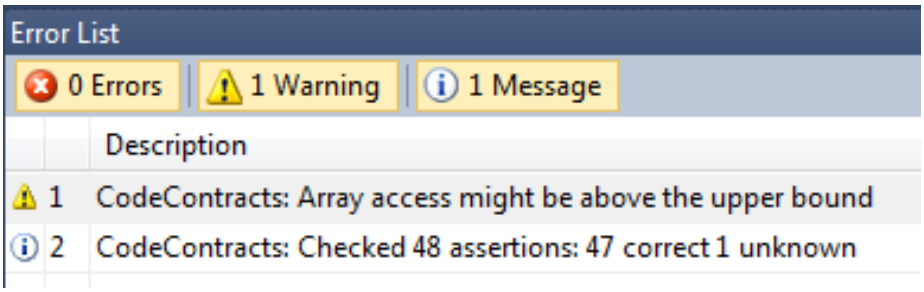
Then build the example. The build should succeed. After a moment[1], the static checker should warn about the following problems:



The first warning is orginated by the fact that the static checker is missing the information that all the array elements of indexes `0 ... index` are not null. We can make this information explicit by adding the following object invariant (in the example, just uncomment it):

```
Contract.Invariant(Contract.ForAll(0, nextFree, i => arr[i] != null));
```

If we build again, we notice that the number of checked assertions as increased. This is due to the fact that now also the condition on the elements of `arr` is checked at method exits.

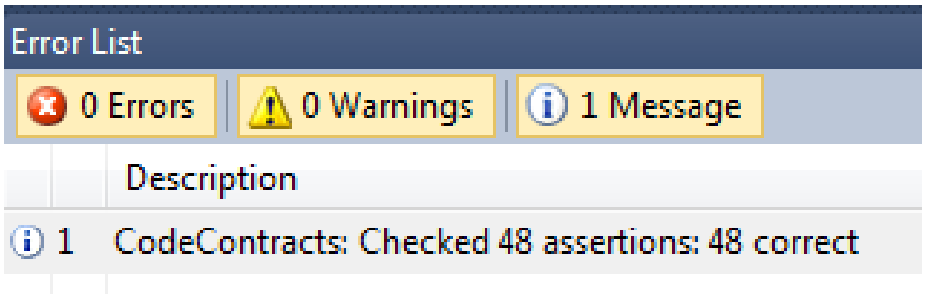


We are left with the last message from the static checker, warning us that the array access may be not be in bounds. This happens when `arr.Length` is zero. Changing the array creation expression as

```
var newArr = new T[arr.Length * 2 + 1];
```

solves the problem:

[1] The static checker runs in the background after the regular build.



`Push` was the only modified method, so the checker used the cache to avoid re-analyzing the unmodified methods (Properties are not cached):

```
CodeContracts: NonNullStack: Total methods analyzed 6
CodeContracts: NonNullStack: Total method analysis read from the cache 3
```