Introduction to .Net, C#, and Visual Studio

January 16, 2008

Coordinates

- Time: Wednesdays, 10-11am
- Place: 140 Fisher–Bennett Hall

- Instructor: Jeff Vaughan
- Office Hours: Tuesdays 2:00-3:00, 514 Levine Hall

- Website:
  `http://www.seas.upenn.edu/~vaughan2/cis399-005/`

Goals

- Learn C#
  - Core Language
  - Generics
  - .Net library ...
- Practice good software engineering
  - Good design
  - Clean code
  - Source control
  - Documentation ...

Problem Sets and Grading

- Course grade will be based entirely on problem set grades
- 5 Problem Sets
  - Emphasis on writing code
  - First four will ask you to do specific tasks
  - Last will be an open ended project
- Policies
  - Code *must* compile with Visual Studio Express 2008
  - Late Policy: -25% per day.
  - Cheating: Unpleasant for all involved.

Java and C# share many features

- Similar syntax, object model, runtime model, etc.
- Today: Learn basic C# features by analogy to Java.
- Throughout the semester: Learn advanced features of C#.

Java: Hello World

```
package greeting;

class Hello
{
    public static void main(String[] args)
    {
        System.out.println(GetText());
    }

    private static String GetText()
    {
        return "Hello World";
    }
}
```

Translating Hello World (I)

```
// Java package declaration
package greeting;

// C# namespace
namespace Greeting
{ ...
}
```

Namespaces contain classes and other other namespaces. Two different classes (or namespaces) with the same name may coexist in different namespaces.

Translating Hello World (II)

```
// Java and C#
class Hello{
  ...
}
```

Basic class declaration look the same in Java and C#.

Translating Hello World (III)

```
// Java
private static String GetText()
{
    return "Hello World";
}

// C#
private static string GetText()
{
    return "Hello World";
}
```

Identical, except that C# treats `string` as a keyword.

Translating Hello World (IV)

```
// Java
public static void main(String[] args)
{
    System.out.println(GetText());
}

// C#
static void Main(string[] args)
{
    System.Console.Out.WriteLine(GetText());
}
```

Unlike Java, a C# program's main method does not need to be public. (By default methods are private.)
In both cases, args is the list of strings provided as input on the command line.

C#: Hello World

```
namespace Greeting
{
    class Hello
    {
        static void Main(string[] args)
        {
            System.Console.Out.WriteLine(GetText());
        }

        private static string GetText()
        {
            return "Hello␣World";
        }
    }
}
```

More Syntax: Inheritance and Interface Syntax

```
class Animal { ... }

// .Net convention: name interfaces
// using IName format
interface IWarmBlood { ... }

interface IBackBone { ... }

// Uniform syntax for extending a base
// class and implementing an interface
class Mammal : Animal, IWarmBlood, IBackBone {
    ...
}
```

More details of class and object system later this semester...

More Syntax: Arrays

```
// Declare a new array
int[] myArray = new int[6];

// Declare and initialize an array
int[] myOtherArray = {2, 4, 6, 8, 10, 12};

// Iterate through an array
for( int i = 0; i < myOtherArray.Length; i++) {
    myArray[i] = myOtherArray[i];
}
```

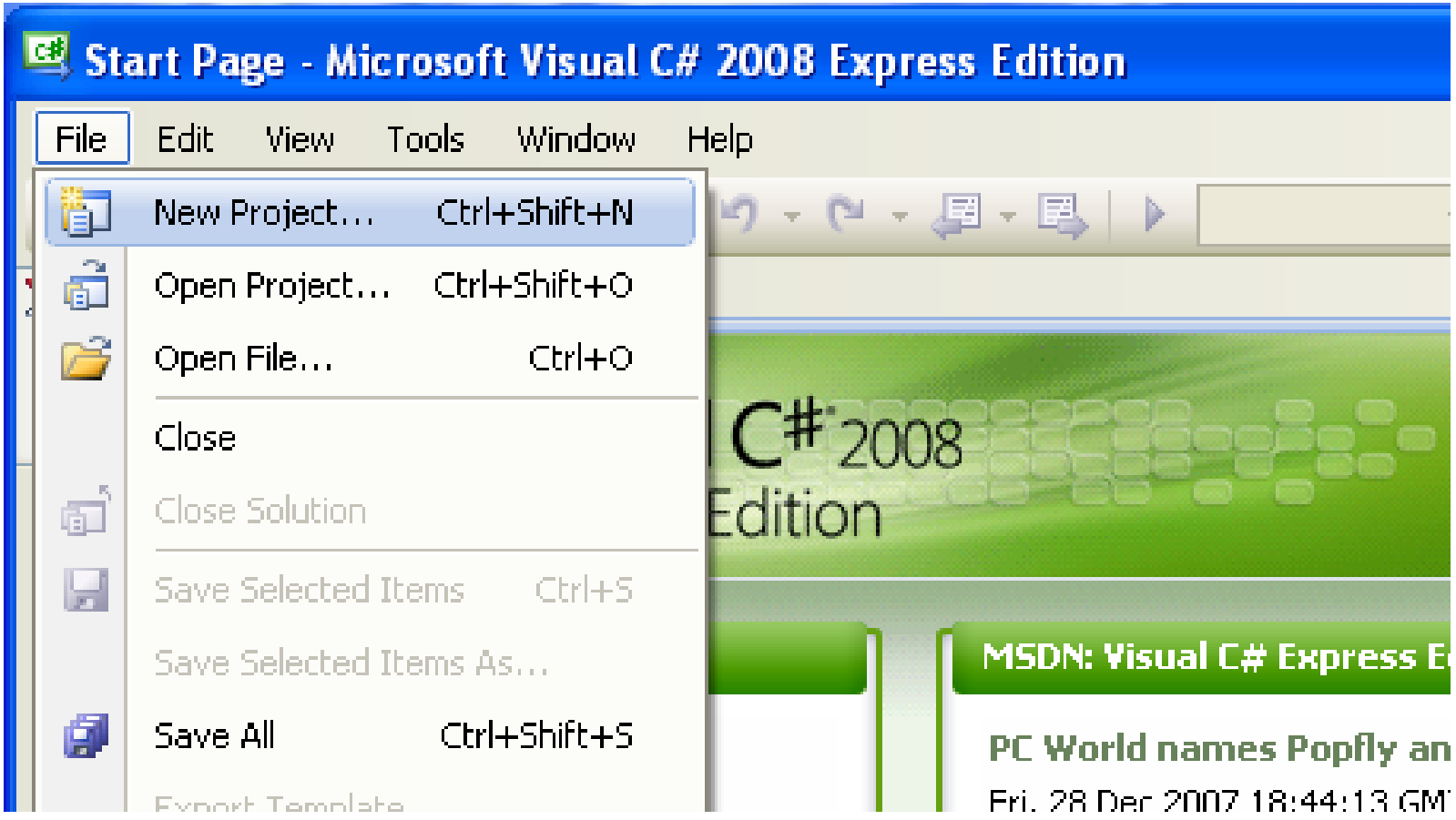
But we will learn about better ways to store data soon...

Course Software

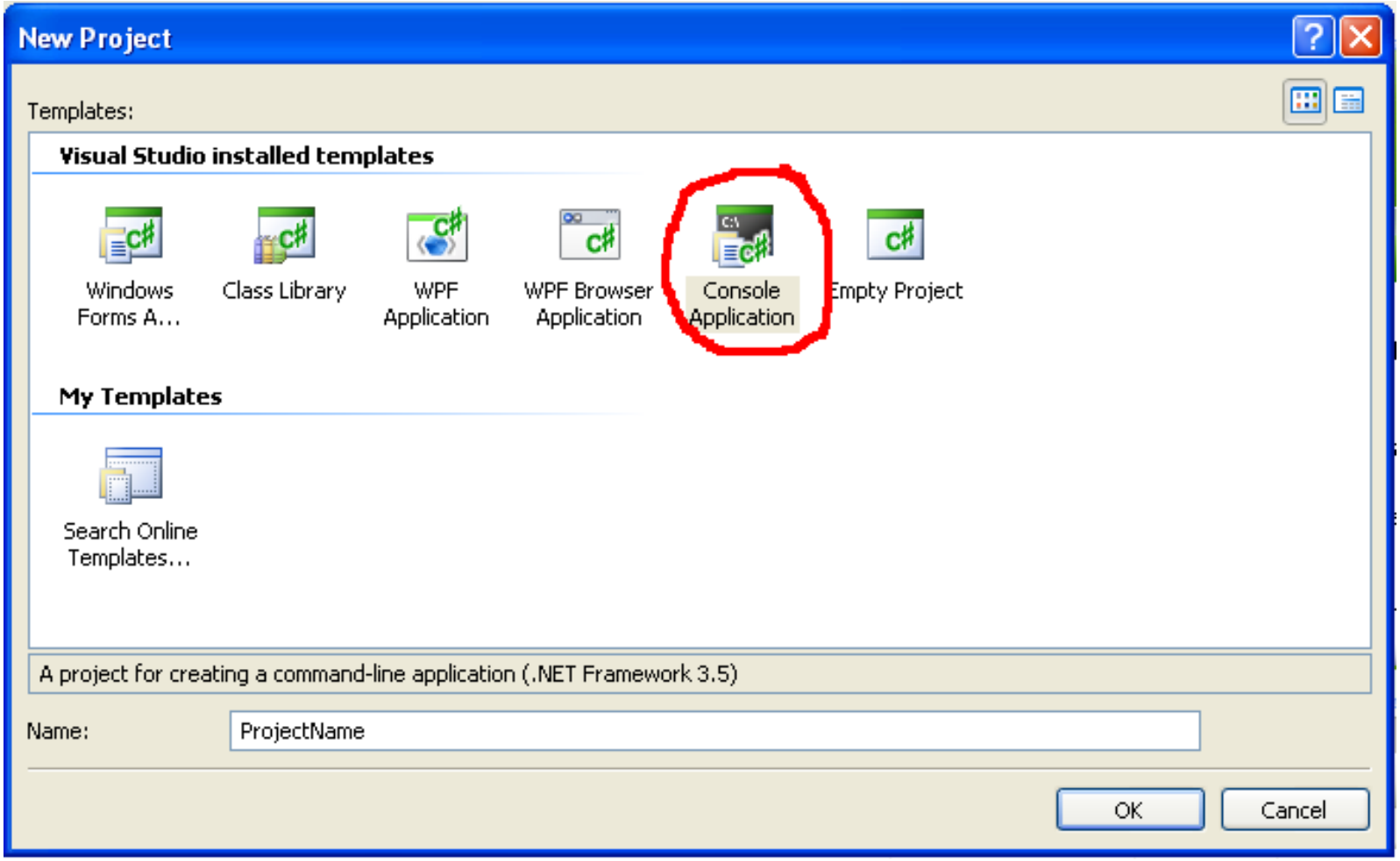
- Visual Studio Express 2008, C# edition, Microsoft's free compiler.
- NUnit 2.4.6, A framework for unit testing.

Links to both on the course website.

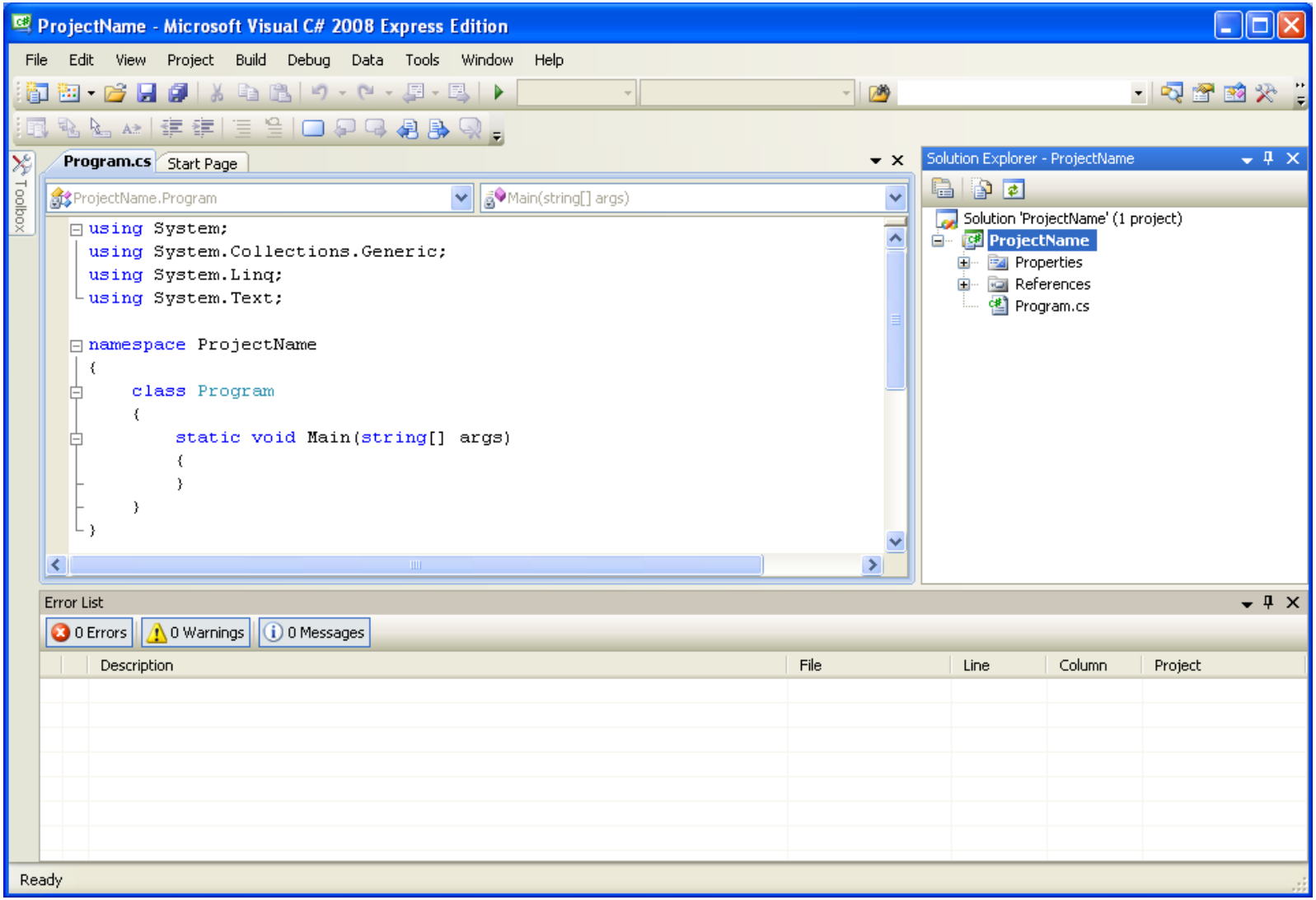
Creating a console app in Visual Studio



Creating a console app in Visual Studio



Creating a console app in Visual Studio



Visual Studio organizes files into Projects and Solutions

- Solutions (usually) correspond to file system directories
- Each solution contains one or more projects
- Projects contains items:
  - .cs files (which can represent normal objects, winforms, etc...)
  - references (e.g. nunit.framework),
  - *linq*s to databases
  - xml files...

NUnit is a tool for automated testing

- Programmer specify test methods using *attributes*.
- NUnit runs these methods and reports if *assertions* hold.
- Agile Programming Philosophy: Test every corner case, and run test cases after every time you compile!

NUnit example (I)

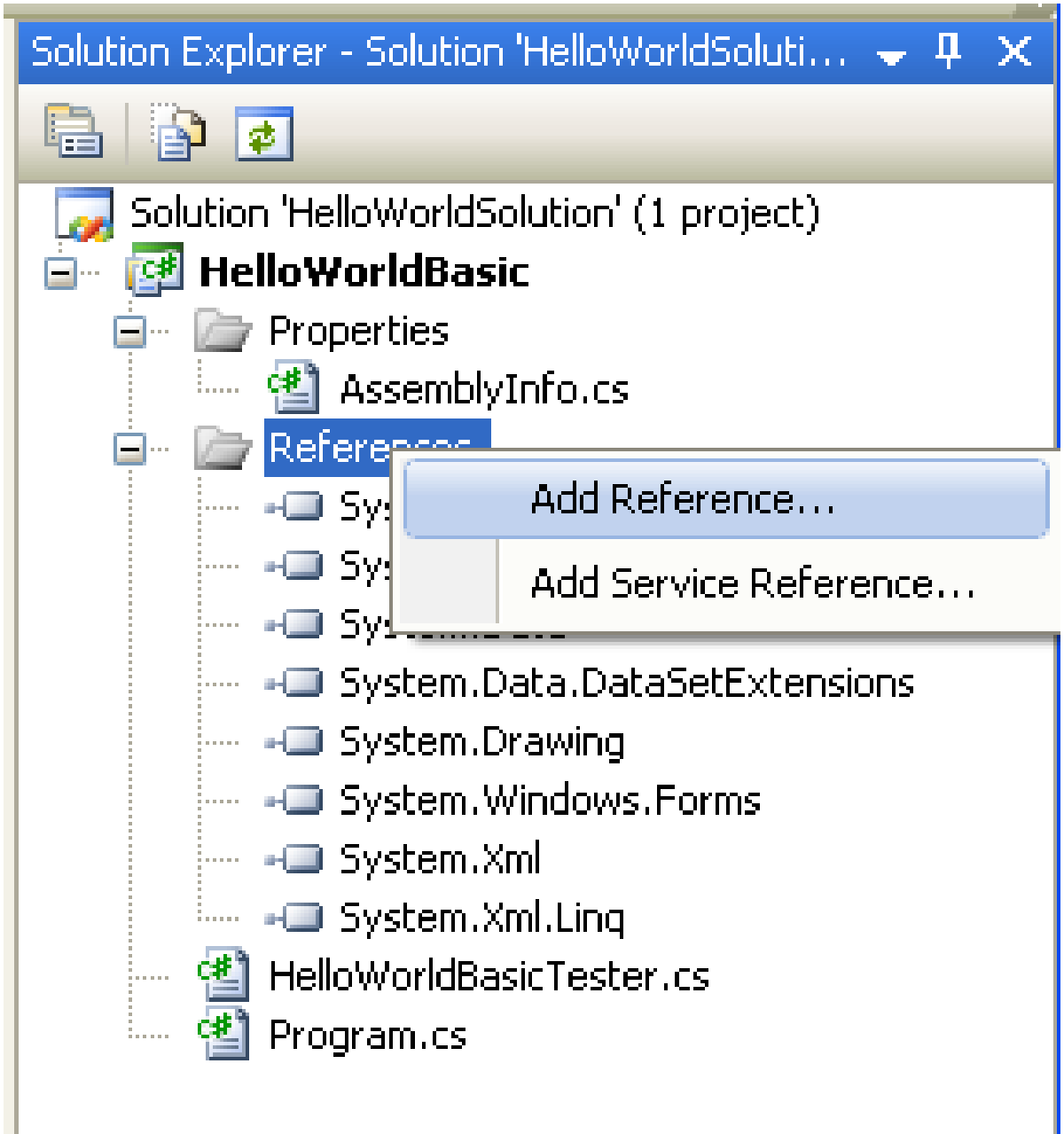
NUnit hooks into test code in your project.

```
using NUnit.Framework;

namespace Greeting
{
    [TestFixture]
    public class HelloTester
    {
        [Test]
        public void CorrectString()
        {
            Assert.AreEqual(Hello.GetText(),
                            "Hello␣World");
        }
    }
}
```

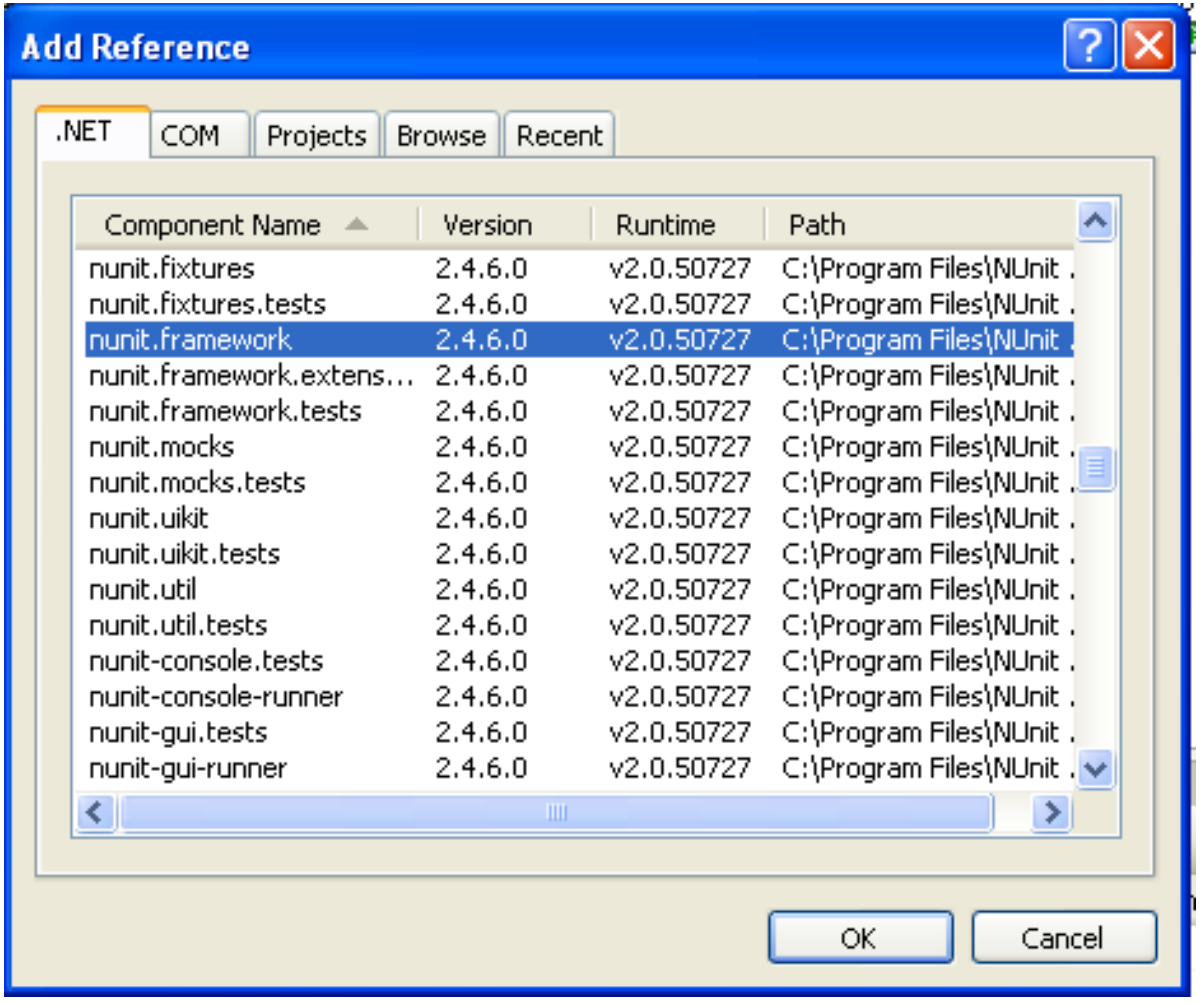
NUnit example (II)

Add Nunit.Framework as a reference for your project.



NUnit example (II)

Add Nunit.Framework as a reference for your project.

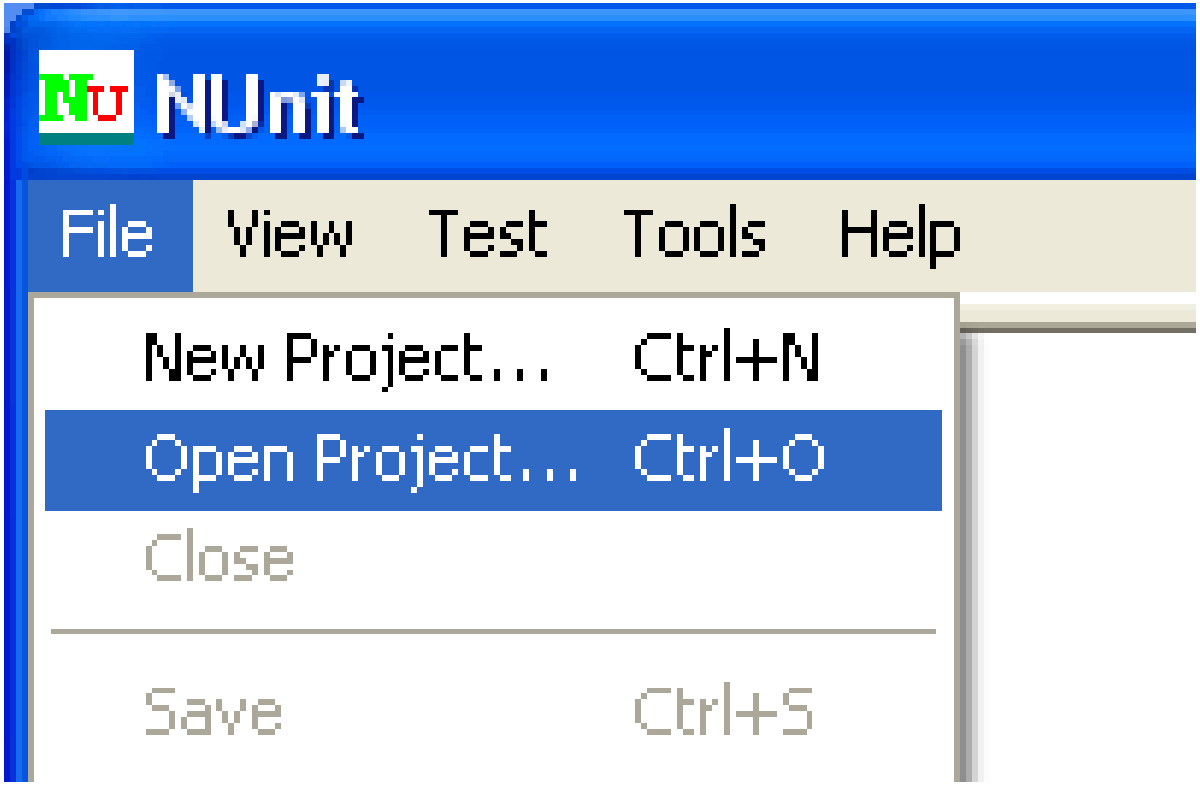


NUnit example (III)

Run tests from within the Nunit program.

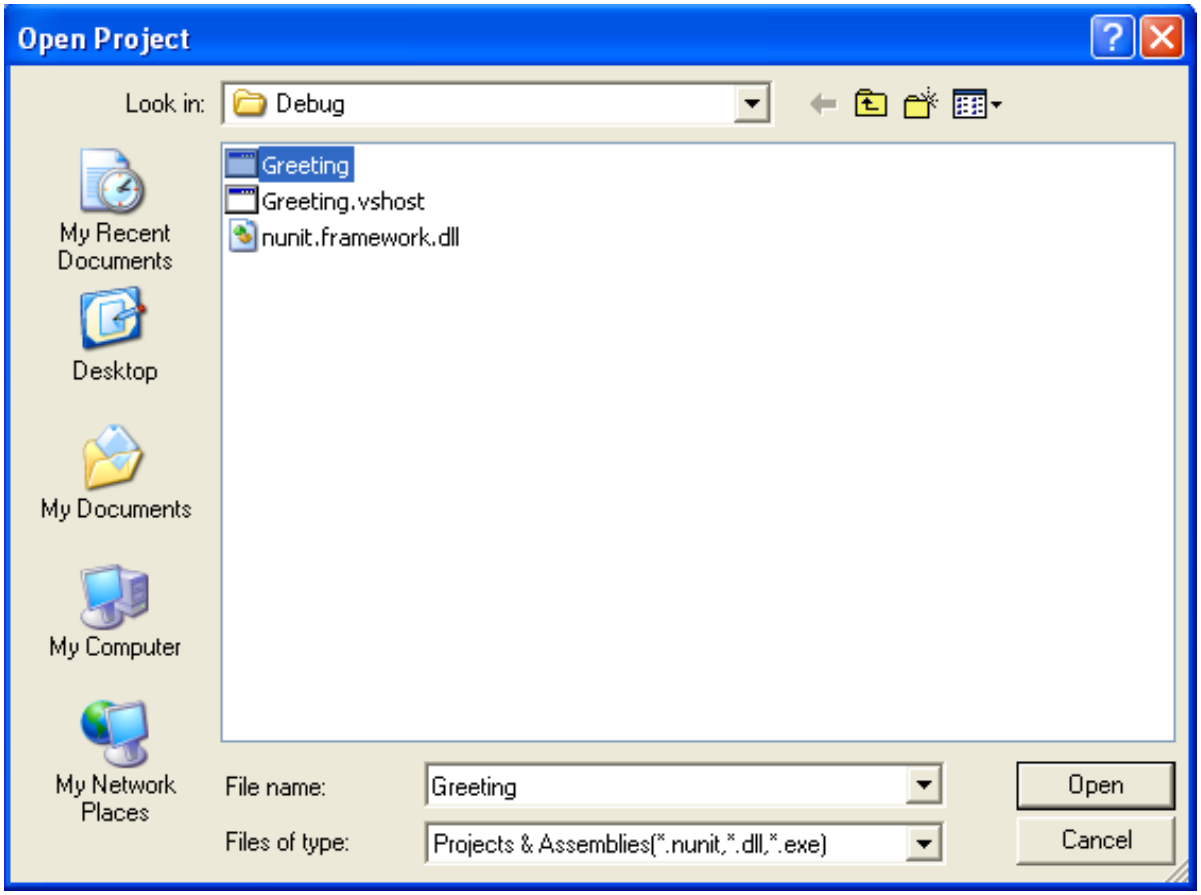
NUnit example (III)

Run tests from within the Nunit program.



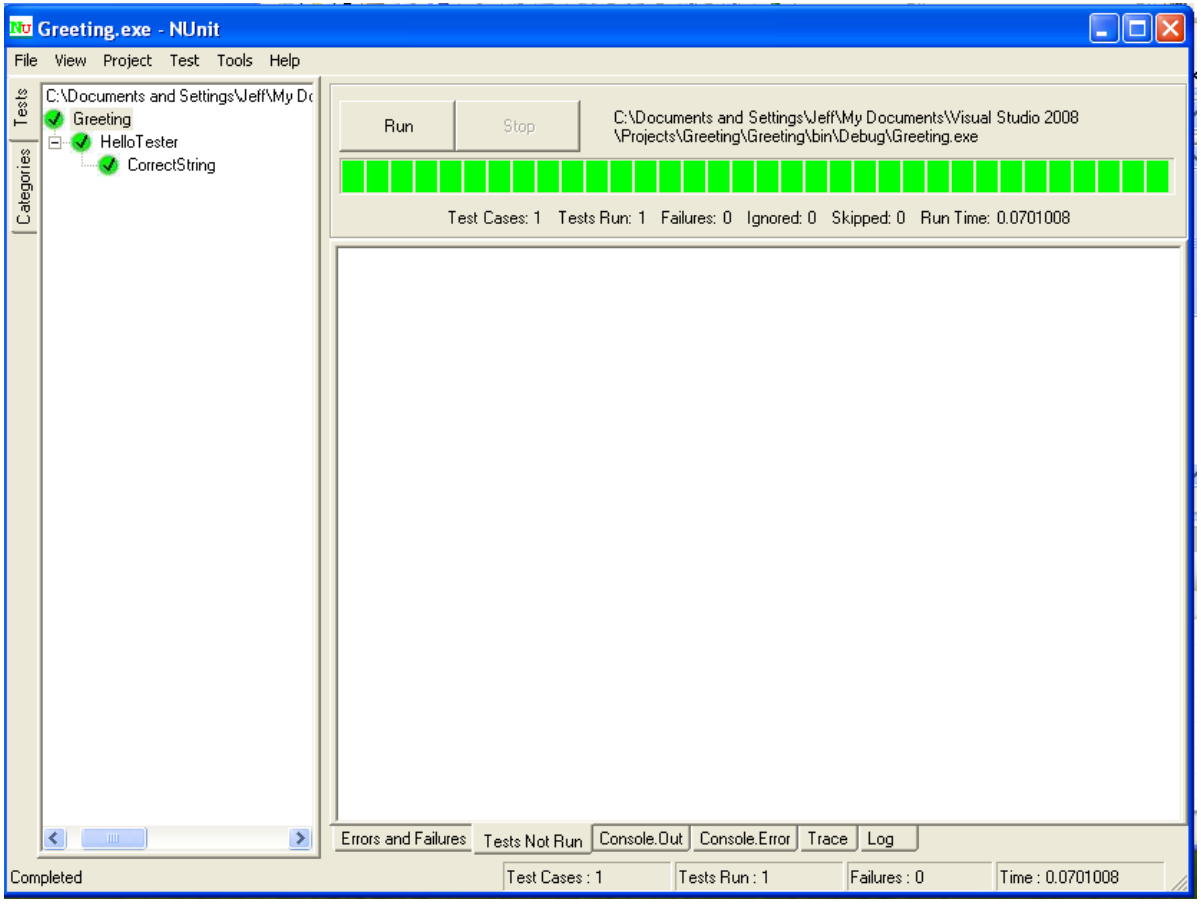
NUnit example (III)

Run tests from within the Nunit program.



NUnit example (III)

Run tests from within the Nunit program.



Problem Set 1

- First problem set available online today.
- Problem set is designed to get you setup for development.
- Due next Wednesday (1/23). Email your solution to me before class.