

#### Aspen Townhouse Condo

Architect's 3 floor 3 bed townhouse  
Aspen Club membership included  
[aspenskirental.com](http://aspenskirental.com)

#### Mens Invisible High Heels

Make you 5 inches Taller Invisibly  
200+ Styles Selection, From \$29.99  
[www.TallMenShoes.com](http://www.TallMenShoes.com)

#### Ushuaia x \$ 116

¿Buscas Hotel en Ushuaia? Super Descuentos Reservando Online!  
[Despegar.com.ar/Ushuaia](http://Despegar.com.ar/Ushuaia)

### The Liskov Substitution Principle - Agile Software Development Principles Patterns and Practices

For the past few weeks I have been doing a [chapter-by-chapter review of Agile Software Development Principles, Patterns, and Practices by Robert Martin](#). So far, it has covered a lot of good introductory information on various Agile Methodologies (Test-Driven Development, Refactoring, Pair Programming, Project Planning, Extreme Programming, etc.) as well as talked about various object-oriented programming principles, which is what I am still covering now.

Here is a chapter-by-chapter breakdown on what I have covered so far:

Agile Software Development, Principles, Patterns, and Practices

**Book:** Agile Software Development, Principles, Patterns, and Practices ([Amazon](#))

**Author:** Robert C. Martin ([Amazon](#))

**Publisher:** Prentice Hall; 1st edition (October 15, 2002)

**Hardcover:** 552 pages

Previous Chapters

- [Chapters 1 - 4 - Agile Practices, XP, TDD](#)
- [Chapters 5 and 6 - Refactoring and Pair Programming](#)
- [Chapter 7 - What is Agile Design?](#)
- [Chapter 8 - Single-Responsibility Principle](#)
- [Chapter 9 - Open-Closed Principle](#)

Chapter 10 is on the **Liskov Substitution Principle**:

#### Liskov Substitution Principle

"What is wanted is something like the following substitution property:  
If for each object o1 of type S there is an object o2 of type T such that for all programs P defined in terms of T, the behavior of P is unchanged when o1 is substituted for o2, then S is a subtype of T.  
[Liskov88]"


Is it me, or is that painful to read? Robert Martin puts it a bit simpler:

**Subtypes must be substitutable for their base types**

The classic example of this principle in code is inheriting the **Square** Class from the **Rectangle** Class.

#### Classic Example of LSP

#### Main

- [Blog Home](#)
- [Syndication](#) 

FOLLOW ME ON G+

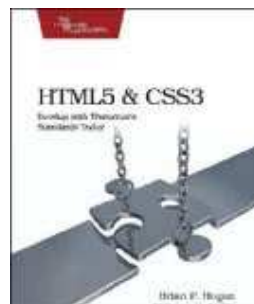
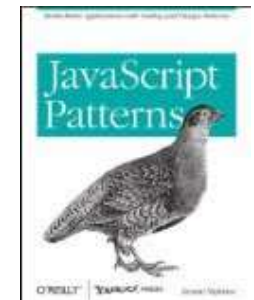
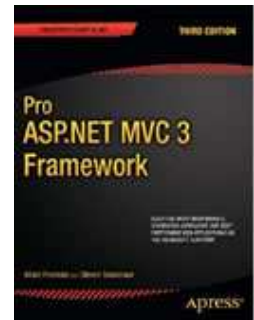


FOLLOW ME ON

twitter

#### Health & Fitness

- [Ab Rocket Review](#)
- [Ab Wheel Review](#)
- [Bowflex Dumbbells Review](#)
- [Iron Gym Extreme Review](#)
- [Perfect Pushup Review](#)
- [Proform 890E Elliptical](#)
- [Weight Loss - Healthy Diets](#)



```

public class Rectangle
{
    protected int _width;
    protected int _height;

    public int Width
    {
        get { return _width; }
    }

    public int Height
    {
        get { return _height; }
    }

    public virtual void SetWidth(int width)
    {
        _width = width;
    }

    public virtual void SetHeight(int height)
    {
        _height = height;
    }
}

public class Square: Rectangle
{
    public override void SetWidth(int width)
    {
        _width = width;
        _height = width;
    }

    public override void SetHeight(int height)
    {
        _height = height;
        _width = _height;
    }
}

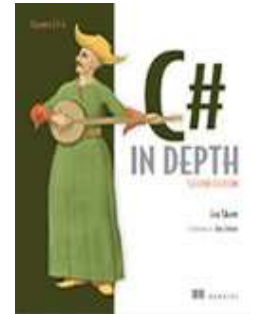
[TestFixture]
public class RectangleTests
{
    [Test]
    public void AreaOfRectangle()
    {
        Rectangle r = new Square();

        r.Width = 5;
        r.Height = 2;

        // Will Fail - r is a square and sets
        // width and height equal to each other.
        Assert.AreEqual(r.Width * r.Height, 10);
    }
}

```

If you look at the test above, it will fail because a square is being substituted for a rectangle and the area won't be 10 as expected. It will actually be 4 because "unexpectedly" in this case, both height and width are being set to each other when the width or height is set on a square. Therefore, if this behavior by Square is unacceptable and unexpected, Square should not be derived from Rectangle (at least not coded like this with these expectations anyway).



## ASP.NET MVC

- ASP.NET MVC 3 Tutorials
- ASP.NET MVC 3 Videos
- ASP.NET MVC 3 Training

## Orchard CMS

- Creating a Google +1 Part and Custom Module in Orchard CMS - Part 1
- Adding Google +1 and Twitter Social Networking Links to Orchard CMS
- Upgraded Website to Orchard 1.2
- Orchard 1.2 Performance Settings - Warmup Module Goes Native
- Orchard 1.2 Released - Bug Fixes and Improved Performance
- Delete Content Types in Orchard CMS with Custom Module
- Add Widget Layer to Orchard CMS - Usability Workflow Improvement
- Custom Orchard Module that Plays HTML5 Video with Flash Fallback
- Create New Orchard Website Using WebMatrix - Orchard Web Development
- Delete Blog in Orchard CMS - Orchard Tips and Tricks
- Customizing Orchard Themes using Shape Tracing and Url Alternates in Designer Tools
- Orchard Installation
- Orchard Configuration
- Running Orchard CMS
- Orchard CMS Administration
- Creating New Page in Orchard
- Orchard Widgets
- Orchard Modules
- Orchard Themes
- Developing Modules in Orchard CMS
- Custom Orchard Modules
- Orchard SEO
- Orchard Shared Hosting
- Visual Studio 2010 SP1 and

This is the whole point of the **Liskov Substitution Principle**. It basically wants you to think clearly about the expected behavior and expectations of a class before you derive new classes from it. It could turn out that when subtypes are substituted for a parent, you may get unexpected results. This is where unit tests can really be handy. The unit tests essentially describe and test for the expected behavior of objects (design by contract, if you will).

If you want to read some discussions as to the usefulness of this principle, whether it should be a principle, and thoughts on the classic example above, check out this [wiki](#). You can also read what Robert Martin has to say about it from this [PDF](#).

posted on Friday, June 10, 2005 10:53 AM

#### **Buenos Aires: Ofertas 70%**

Descuentos Increíbles. Ofertas en Buenos Aires hasta un 70% menos!

[www.GROUPON.com.ar/Ofertas](http://www.GROUPON.com.ar/Ofertas)

- [Visual Studio 2010 SP1 and SQL CE 4 Tooling For Orchard CMS](#)
- [Free Website Database - SQL Server Compact Edition](#)
- [Develop Orchard Modules - SlideShow Orchard CMS Module](#)
- [Orchard Web Design and Development - Why We Chose Orchard](#)
- [Orchard Web Development Using WebMatrix IDE from Microsoft](#)
- [Orchard Website Development and Code Generation](#)
- [Orchard CMS Performance and Caching in Orchard Modules](#)
- [Orchard CMS Search Engine Optimization - Sitemap.xml Module](#)
- [Migrate Orchard Database from SQL Server CE to SQL Server using WebMatrix](#)
- [Shape Tracing in Orchard Designer Tools for Creating Orchard Themes](#)
- [Orchard Command-Line to Enable Orchard Website Features](#)
- [Favicon and Apple Touch Icon for Orchard CMS Websites](#)
- [Orchard and IIS Application Pool Idle Timeout](#)
- [Orchard Startup Times - Orchard Keep Alive Module for Pinging Orchard Websites](#)
- [Orchard Warmup Module - Orchard Startup Times on Shared Hosting](#)
- [Orchard Blogs - Using Windows Live Writer with Orchard CMS](#)
- [Orchard Recipe - Orchard Configuration Profiles in Orchard CMS 1.1](#)
- [Orchard Training at MIX - Deconstructing Orchard: Build, Customize, Extend, Ship](#)
- [Orchard 1.1 Coming April 12 2011 During MIX](#)
- [Orchard 1.1 Warmup Module for Startup Performance](#)

#### **Categories**

- [ADO.NET 2.0](#)
- [ADO.NET 3.0](#)
- [ADO.NET Data Services](#)
- [ADO.NET Entity Framework](#)
- [Agile & Iterative Development](#)
- [Agile Software Development](#)
- [Applying UML and Design Patterns](#)
- [ASP.NET](#)
- [ASP.NET 2.0](#)
- [ASP.NET 3.5](#)
- [ASP.NET Dynamic Data](#)
- [ASP.NET Security](#)
- [C#](#)
- [C# 2.0](#)
- [C# 3.0](#)
- [Code Generation](#)
- [Design Patterns C#](#)
- [LINQ to SQL - DLINQ](#)
- [O/R Mappers](#)
- [SQL Server](#)
- [SQL Server Management Objects \( SMO \)](#)

