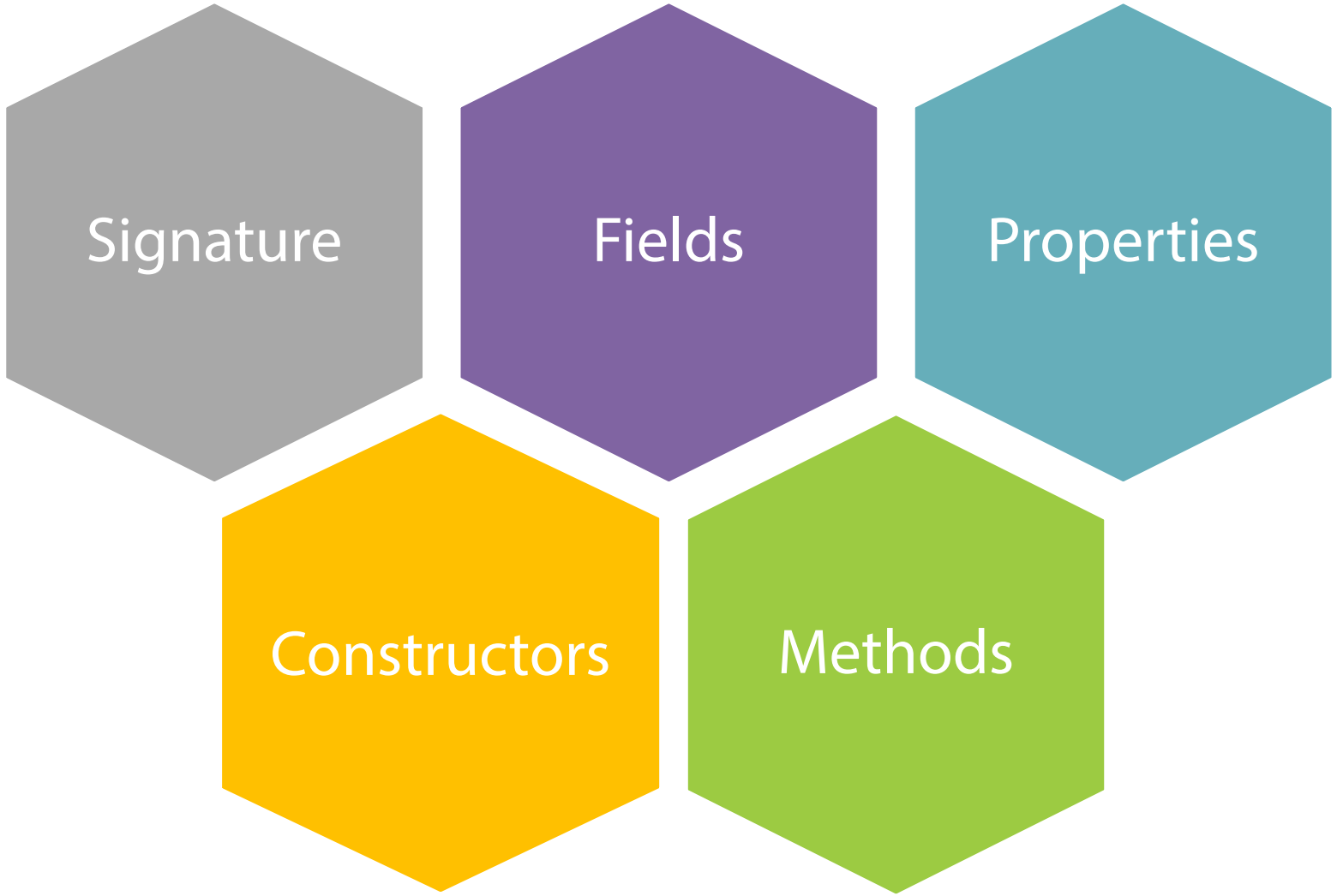
Building Good Classes

Deborah Kurata

@deborahkurata | blogs.msmvps.com/deborahk/

pluralsight

Building a Class



pluralsight

Module Overview

- Building a Class
- Defining Constructors
- Namespaces
- Building a Static Class
- Defining a Singleton
- FAQ

pluralsight

Class Signature

```
namespace Acme.Biz
{
    /// <summary>
    /// Manages products carried in inventory.
    /// </summary>
    public class Product
    {

    }
}
```

class keyword

Class name

Product

Order

OrderItem

Accessibility modifier

Default is: `internal`

XML document comments

pluralsight

Fields

```
public class Product
{
    private int productId;
    private string productName;

    public int ProductId
    {
        get { return productId; }
        set { productId = value; }
    }
    public string ProductName
    {
        get { return productName; }
        set { productName = value; }
    }

    public decimal CalculateQuantityOnHand()
    {
        var quantity = 0;
        Calculate the number in inventory
        return quantity;
    }
}
```

A variable in the class

Holds the data

productId: 1
productName: Hammer

productId: 2
productName: Saw

pluralsight

Properties

```
public class Product
{
    private int productId;
    private string productName;

    public int ProductId
    {
        get { return productId; }
        set { productId = value; }
    }
    public string ProductName
    {
        get { return productName; }
        set { productName = value; }
    }

    public decimal CalculateQuantityOnHand()
    {
        var quantity = 0;
        Calculate the number in inventory
        return quantity;
    }
}
```

- Getter and setter functions
- Guard access to the fields

pluralsight

Methods

```
public class Product
{
    private int productId;
    private string productName;

    public int ProductId
    {
        get { return productId; }
        set { productId = value; }
    }
    public string ProductName
    {
        get { return productName; }
        set { productName = value; }
    }

    public decimal CalculateQuantityOnHand()
    {
        var quantity = 0;
        Calculate the number in inventory
        return quantity;
    }
}
```

- Functions
- Behaviors and operations

pluralsight

Class Best Practices

Do:

Class naming

- Define a meaningful name
- Use a noun
- Use PascalCasing

Add XML document comments

Use properties to encapsulate fields

Use methods for logic

Avoid:

Class naming

- Abbreviations
- Prefixes
- Underscores

pluralsight

Class Best Practices (cont)

Do:

Ensure the class has a well-defined purpose

Create one class per code file

Add properties above the methods

Avoid:

Large classes

pluralsight

Defining Constructors

```
public class Product
{
    public Product()
    {

    }
```

- Special method in the class
- Executed when instance is created
- Named with the class name
- Default constructor has no parameters
- Not required

pluralsight

Parameterized Constructors

```
public class Product
{
    public Product()
    {

    }

    public Product(string productName) : this()
    {
        ProductName = productName;
    }
```

- Defines parameters to initialize the instance
- Constructor overloading
- Use "this" to invoke another constructor
- Constructor chaining
- Minimizes repeated code

pluralsight

Constructor Best Practices

Do:

Consider providing a default constructor

Consider providing a parameterized constructor

Initialize the minimum properties for a valid object

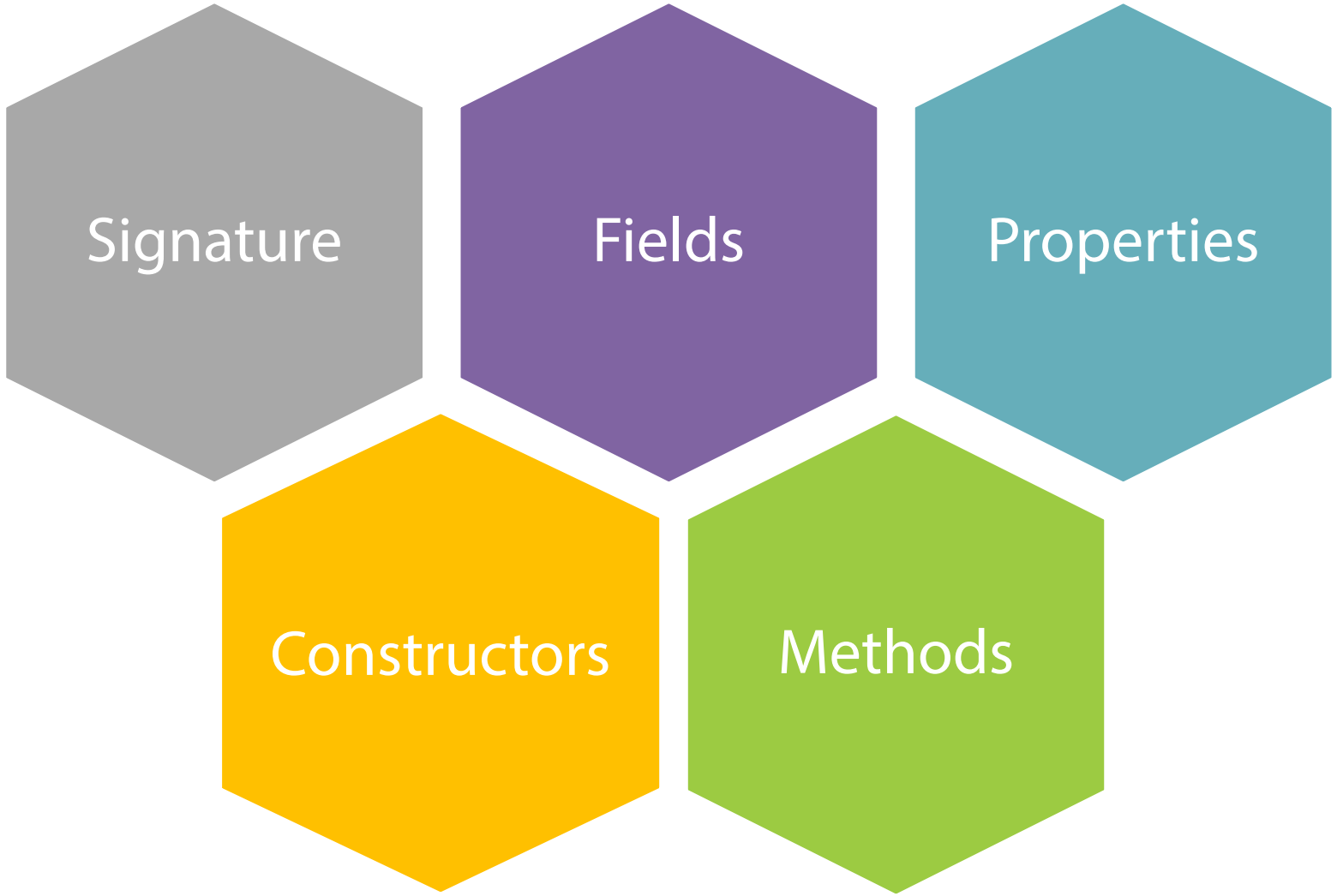
Name the parameters the same name as the related properties

Avoid:

Performing too much work

pluralsight

Building a Class



pluralsight

Namespaces

```
namespace Acme.Biz
{
    /// <summary>
    /// Manages products carried in inventory.
    /// </summary>
    public class Product
    {

    }
}
```

Automatically added around every class

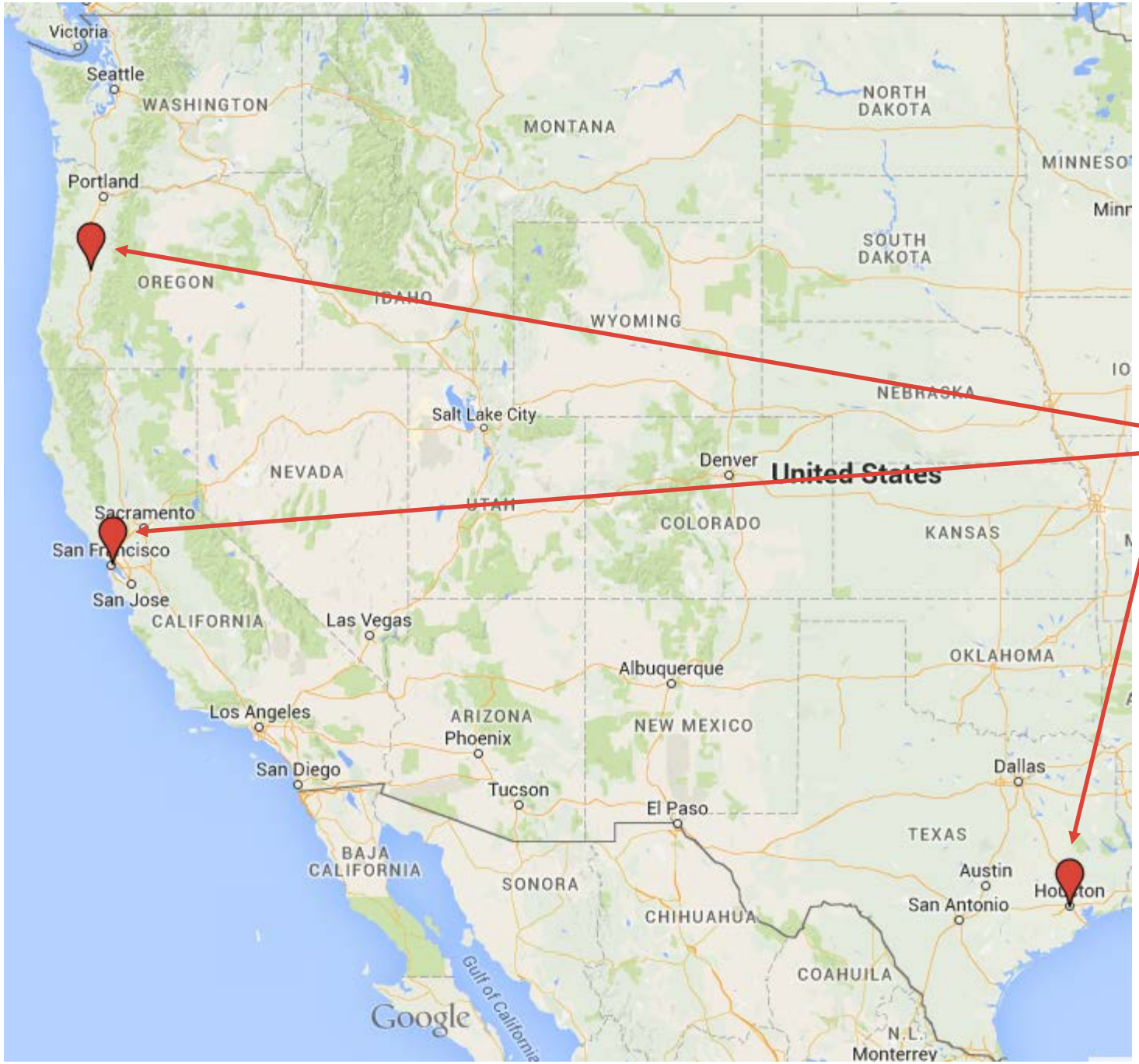
Same name as the project

Used to

Provide a unique address

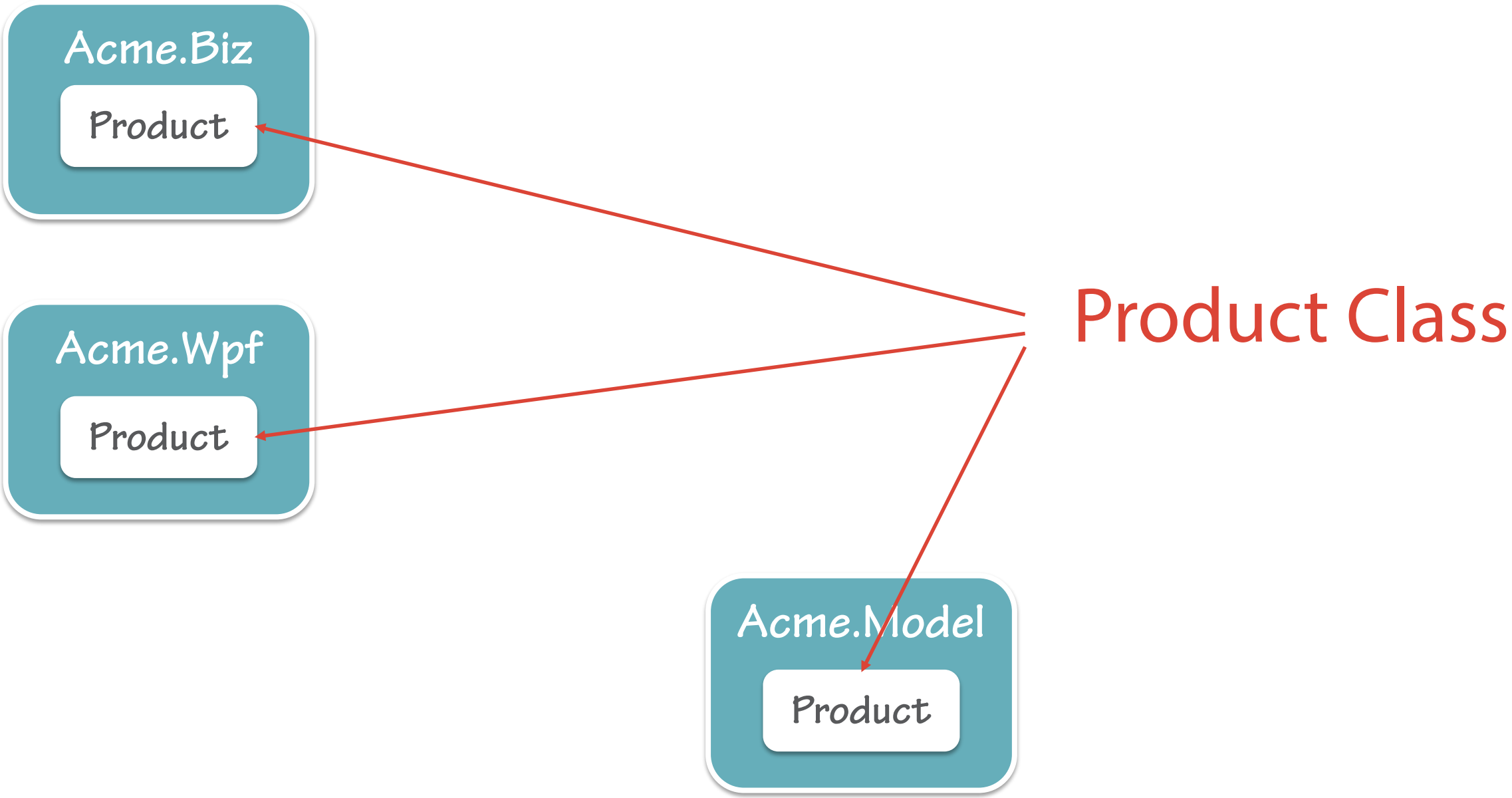
Organize classes into a logical hierarchy

pluralsight



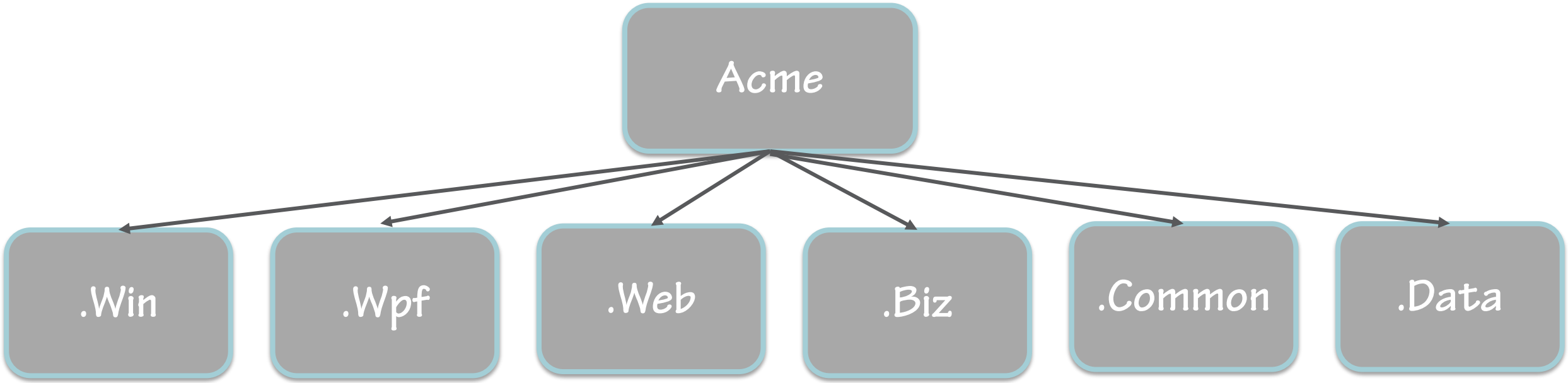
123 Main Street

pluralsight



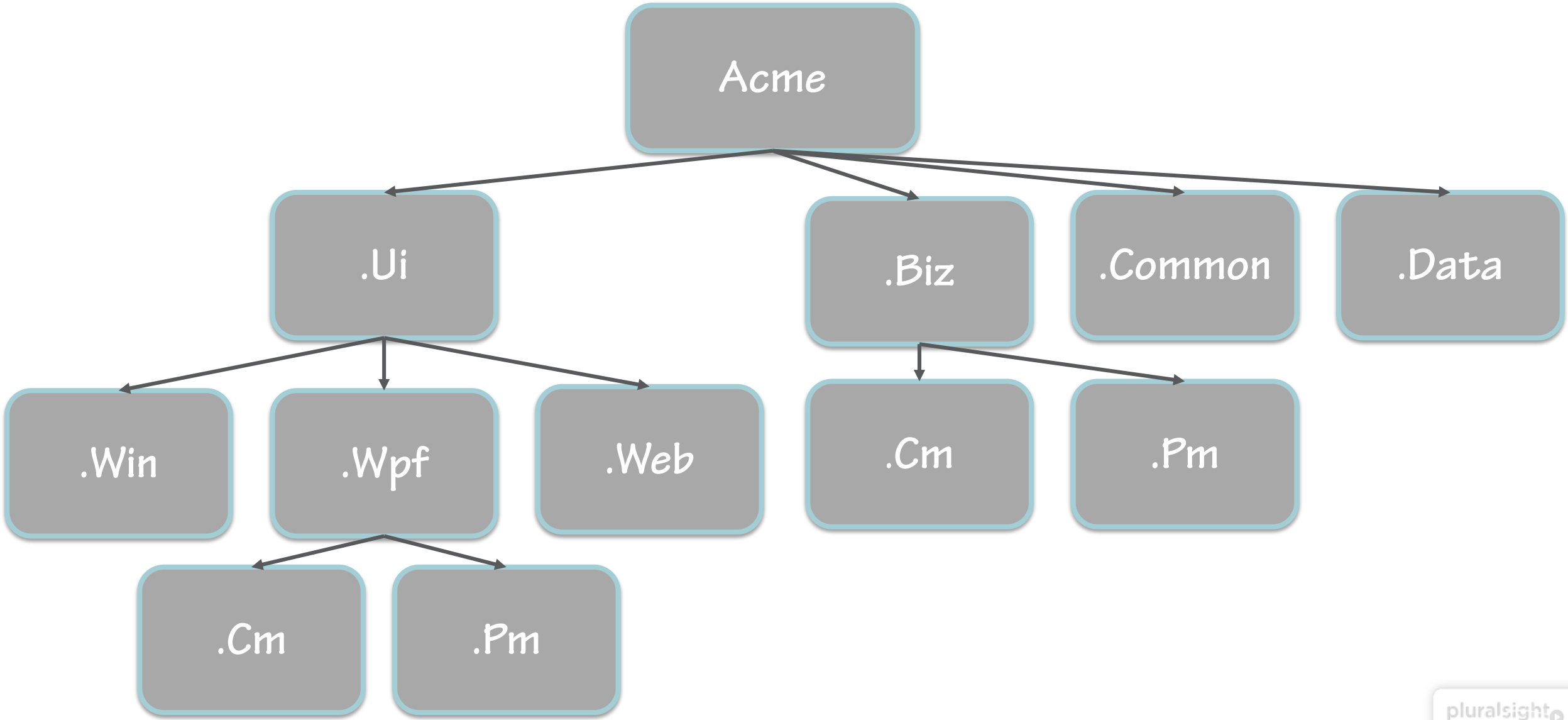
pluralsight

Namespace Hierarchy



pluralsight

Namespace Hierarchy



pluralsight

Namespace Best Practices

Do:

Follow:

<company>.<technology>.<feature>

Acme.Wpf.Pm

Microsoft.Office.Interop.PowerPoint

Use PascalCasing

Avoid:

Using System

System.Windows.Media.Imaging

Using a class name

Namespace: **Acme.Biz.Product**

Class: **Acme.Biz.Product.Product**

pluralsight

Building a Static Class

```
namespace Acme.Common
{
    /// <summary>
    /// Provides logging services.
    /// </summary>
    public static class LoggingService
    {
        /// <summary>
        /// Logs actions.
        /// </summary>
        /// <param name="action">Action to log.</param>
        public static string LogAction(string action)
        {
            var logText = "Action: " + action;
            Console.WriteLine(logText);

            return logText;
        }
    }
}
```

static keyword in the signature

Only static members

Can not instantiate a static class
Use the class name instead

Provides a container for utility features

pluralsight

Static Classes Best Practices

Do:

Use sparingly

Supporting classes

Use for common code library components when needed

Avoid

Using as a miscellaneous bucket

Every class should have a purpose

pluralsight

Defining a Singleton

```
public class User
{
    private static User instance;

    private User() { }

    public static User Instance
    {
        get
        {
            if (instance == null)
            {
                instance = new User();
            }
            return instance;
        }
    }
}
```

Provides only one instance

Private constructor(s)

Static property provides the one instance

Instance accessed with `User.Instance`

pluralsight

Recommended Viewing

"Design Patterns Library"

"Design Patterns On-Ramp"

pluralsight

Advantages of a Singleton vs. Static Class

- A singleton has an instance
  - Can be passed to other code as needed
- A singleton can have child objects
  - Example: User instance has a set of roles associated with it.
- A singleton supports object-oriented programming features
  - It can implement an interface
  - It can be inherited from

pluralsight

Frequently Asked Questions

- What is the difference between a property and a method?
  - Properties are the gate-keepers, providing access to the data
  - Methods are the operations
- What is a constructor?
  - A method executed when an instance is created from a class
- What is the purpose of a namespace?
  - Organize classes into a logical hierarchy
  - Prevent class name collisions

pluralsight

Frequently Asked Questions (cont.)

- What is a static class?
  - A class that cannot be instantiated
  - It is best for use with common code libraries
- What is a singleton?
  - A class that provides a single instance of itself
- What is the difference between a static class and a singleton?
  - A static class cannot be instantiated
  - A singleton can instantiate itself and provide that instance

pluralsight

This Module Covered

- Building a Class
- Defining Constructors
- Namespaces
- Building a Static Class
- Defining a Singleton

pluralsight