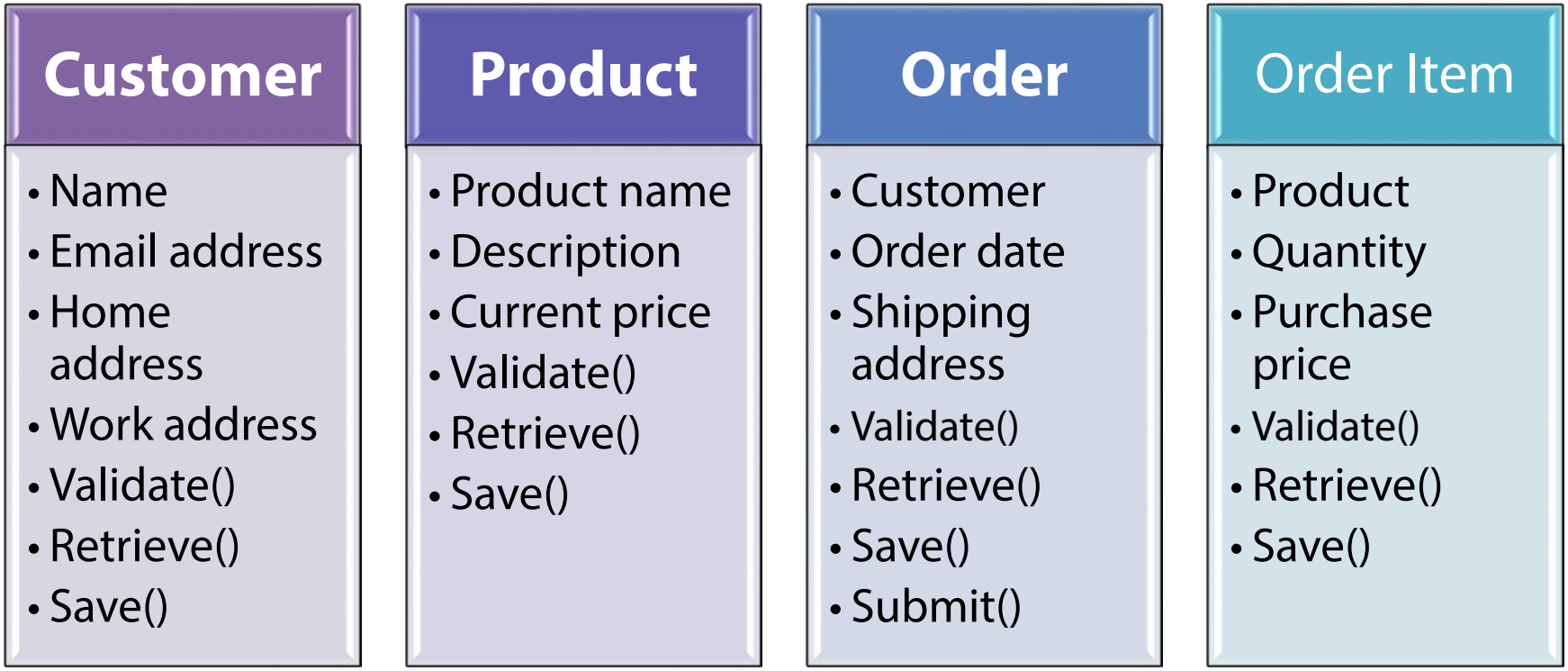
Building Entity Classes

Object-Oriented Programming Fundamentals in C#

Deborah Kurata
http://msmvps.com/blogs/deborahk/
@DeborahKurata
deborahk@insteptech.com

pluralsight
hardcore dev and IT training

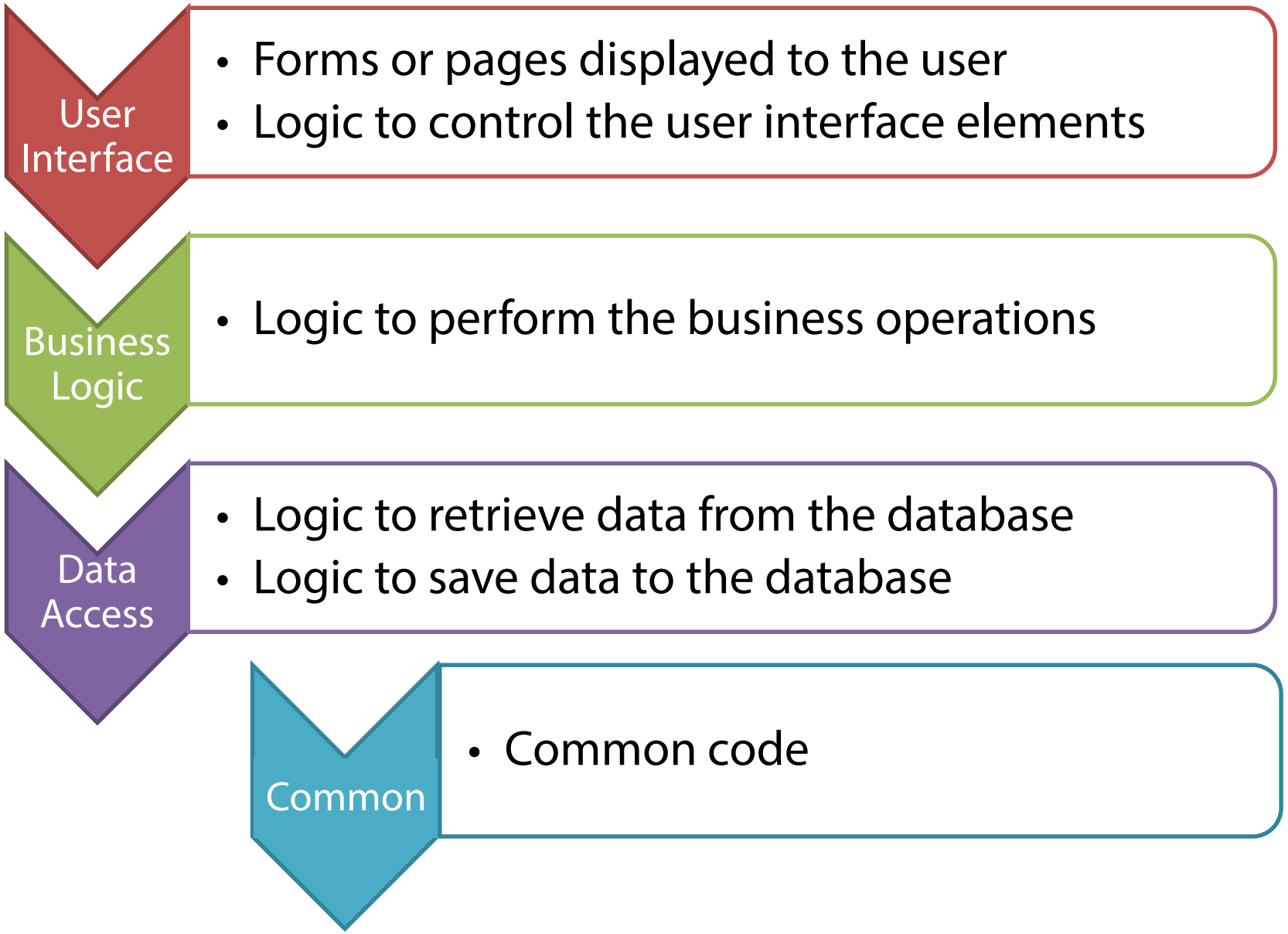
Identified Classes



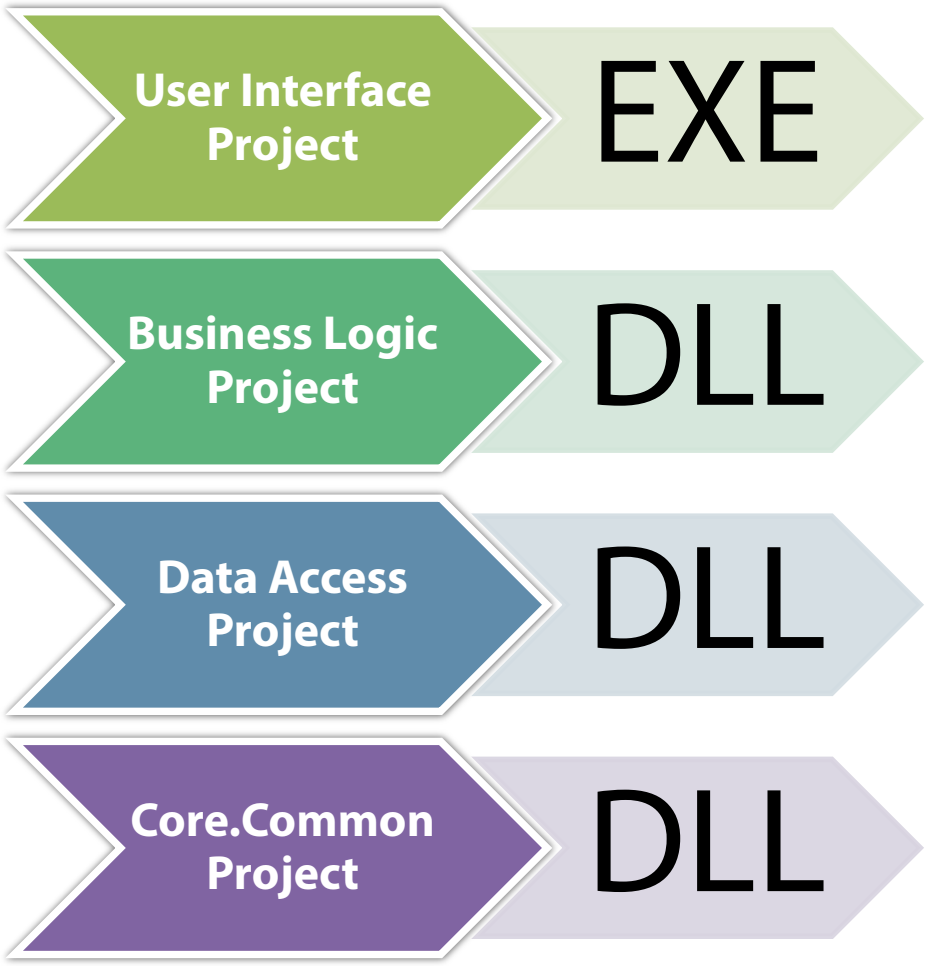
Application Structure

- **Every application has a basic structure**
- **Common structure today: layered**
- **Structure is implemented in Visual Studio as:**
  - Application -> Solution
  - Layer -> Project

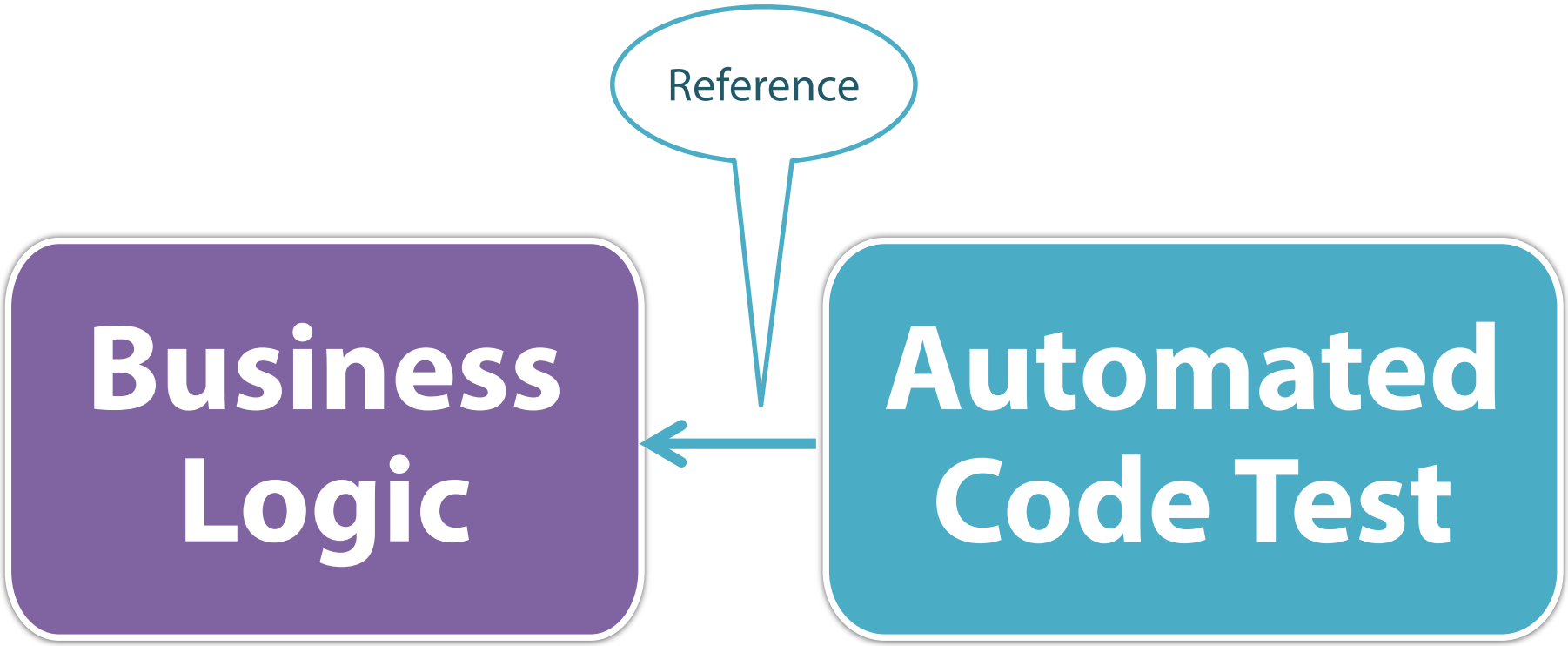
Layering the Application



Visual Studio Solution



Automated Code Testing



Creating a New Object

```
Customer customer = new Customer();
```

```
var customer = new Customer();
```

Accessing Properties

```
var customer = new Customer();
```

```
customer.LastName = "Baggins";
customer.FirstName = "Bilbo";
```

```
var actual = customer.FullName;
```

```
public class Customer
{
    private string _lastName;
    3 references | 2/2 passing
    public string LastName
    {
        get
        {
            // Any code here
            return _lastName;
        }
        set
        {
            // Any code here
            _lastName = value;
        }
    }

    4 references | 2/2 passing
    public string FirstName { get; set; }

    0 references
    public string EmailAddress { get; set; }

    0 references
    public int CustomerId { get; private set; }

    3 references | 3/3 passing
    public string FullName
    {
        get
        {
            string fullName = LastName;
            if (!string.IsNullOrWhiteSpace(FirstName))
            {
                if (!string.IsNullOrWhiteSpace(fullName))
                {
                    fullName += ", ";
                }
                fullName += FirstName;
            }
            return fullName;
        }
    }
}
```

Objects are Reference Types

```
int i1;
i1 = 42;

int i2 = i1;
i2 = 2;
```

What is i1?

```
var c1 = new Customer();
c1.FirstName = “Bilbo”;

var c2 = c1;
c2.FirstName = “Frodo”;
```

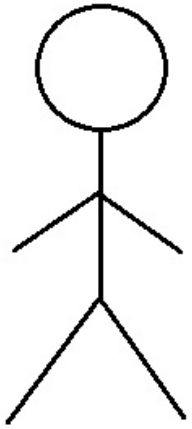
What is c1.FirstName?

i1

i2

c1

c2



Static Modifier

- **Declares a member that belongs to the class itself**

```
public static int InstanceCount { get; set; }
```

- **Accessed using the class name**
  - Not an object variable

```
Customer.InstanceCount += 1;
```

Summary

Layering the Application

- User Interface Business Logic Data Access
- Common

Building the Business Logic Layer

- Application => Visual Studio Solution
- Layer Component => Visual Studio Project

Building a Class

- Class defines a type
- Access Modifier

Defining Properties

- Backing Field C# Property
- Auto-Implemented Properties
- Visual Studio Snippets

Testing the Class

- Separate Test Project
- Set a Reference
- Arrange Act Assert

Summary

Creating an object from a class

```
Customer customer = new Customer();
```

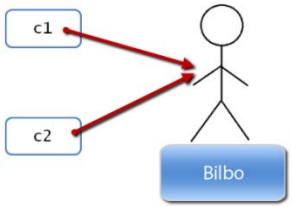
The var keyword

```
var customer = new Customer();
```

Setting and getting data using the C# properties

```
customer.LastName = "Baggins";
```

Objects are Reference Types



The static modifier

```
public static int InstanceCount { get; set; }
Customer.InstanceCount += 1;
```