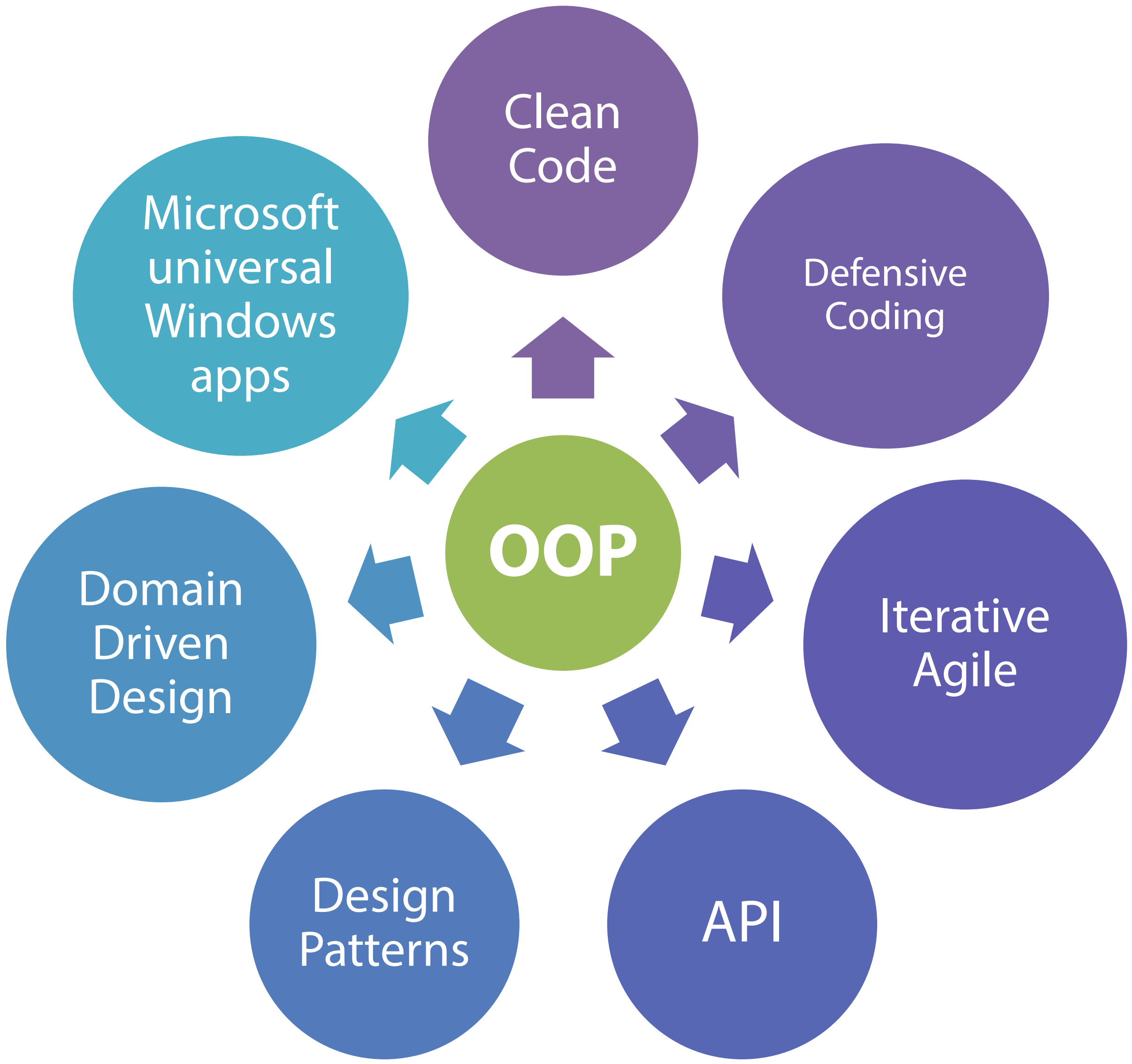
Object-Oriented Programming Fundamentals in C#

Introduction

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

pluralsight
hardcore dev and IT training

OOP is the Foundation

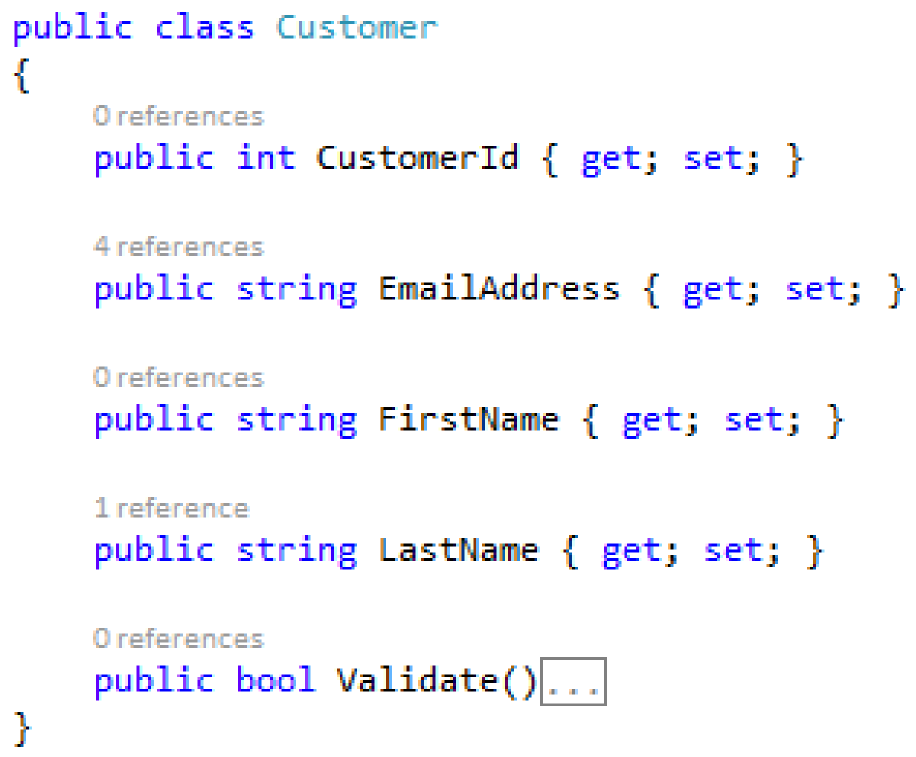


Object != Class

```
Customer customer = new Customer();

customer.FirstName = "Frodo";

customer.Validate();
```

```
public class Customer
{
    0 references
    public int CustomerId { get; set; }

    4 references
    public string EmailAddress { get; set; }

    0 references
    public string FirstName { get; set; }

    1 reference
    public string LastName { get; set; }

    0 references
    public bool Validate()...
}
```

We need to define the business objects



```
public class Customer
{
        0 references
        public int CustomerId { get; set; }

        4 references
        public string EmailAddress { get; set; }

        0 references
        public string FirstName { get; set; }

        1 reference
        public string LastName { get; set; }

        0 references
        public bool Validate()...
}
```

Business Object == Class

Class

Objects

Entity

Customer Management System

Entity	customer
Class	Customer • Last Name • First Name • Go On An Adventure
Objects	Bilbo Baggins Frodo Baggins

Object-Oriented Programming (OOP)

An approach to designing and building applications that are:

- Flexible
- Natural
- Well-crafted
- Testable

by focusing on objects that interact cleanly with one another

Identifying Classes

Separating Responsibilities

Establishing Relationships

Leveraging Reuse

Prerequisites

Basic knowledge of C#

- C# Fundamentals

Experience with Visual Studio

- Introduction to Visual Studio

Course Outline

- Identifying Classes from Requirements
- Building Entity Classes
- Separating Responsibilities
- Establishing Relationships
- Leveraging Reuse
- Building Reusable Components
- Understanding Interfaces
- Final Words and Next Steps

Identifying Classes

Separating Responsibilities

Establishing Relationships

Leveraging Reuse