Object Oriented Programming

K. Scott Allen
odetocode.com
@OdeToCode

pluralsight
hardcore dev and IT training

Pillars of OOP

Encapsulation

Inheritance

Polymorphism

Encapsulation

```
public void WriteGrades(TextWriter textWriter)
{
    textWriter.WriteLine("Grades:");

    int i = 0;
    do
    {
        textWriter.WriteLine(_gra
        i++;
    } while (i < _grades.Count);


    textWriter.WriteLine("*******
}
```

```
public class GradeBook
{
    public GradeBook(string name = "There is no name")...

    public void AddGrade(float grade)...

    public GradeStatistics ComputeStatistics()...

    public void WriteGrades(TextWriter textWriter)...

    public string Name...

    public event NamedChangedDelegate NameChanged;

    private string _name;
    private List<float> _grades;
}
```

Inheritance

```
public class NameChangedEventArgs : EventArgs
{
    public string OldValue { get; set; }
    public string NewValue { get; set; }
}
```

```
public class C
{
    public string Name
    {
        get;
        set;
    }
}

public class B : C
{

}

public class A : B
{
    // "A" object will have a Name property
}
```

Polymorphism

- **Polymorphism == “many shapes”**
  - One variable can point to different types of objects
  - Objects can behave differently depending on their type

```
public class A : object
{
    public virtual void DoWork()
    {
        // ...
    }
}

public class B : A
{
    public override void DoWork()
    {
        // ...
    }
}
```

Abstract Classes

- **Abstract classes cannot be instantiated**
  - Can contain abstract members

```
public abstract class Window
{
    public string Title { get; set; }

    public virtual void Draw()
    {
        // ...
    }

    public abstract void Open();
}
```

Interfaces

- **Interfaces contain no implementation details**
  - Defines only the signatures of methods, events, and properties
- **A type can implement multiple interfaces**

```
public interface IWindow
{
    string Title { get; set; }
    void Draw();
    void Open();
}
```

Important Interfaces

Name	Description
IDisposable	Release resources (files, connections)
IEnumerable	Supports iteration (foreach)
INotifyPropertyChange	Raises events when properties change
IComparable	Compares for sorting

Where to Go from Here

- C# Generics
- LINQ Fundamentals
- C# Fundamentals Part II

Summary

```
public interface IGradeTracker : IEnumerable
{
    void AddGrade(float grade);
    GradeStatistics ComputeStatistics();
    void WriteGrades(TextWriter textWriter);
    string Name { get; set; }
    event NamedChangedDelegate NameChanged;
    void DoSomething();
}
```