Inheritance

Agenda

Inheritance

virtual and override keywords

Abstract classes and methods

Sealed classes and methods

Inheritance

Inheritance

- A class can inherit from another
- Subclasses extend the behavior of the parent or base class with new members
- A class can inherit from one class only
- Structures don’t support inheritance (But support interface implementation)
- Inheritance is transitive
- No limitation on the number of inheritance levels

Parent class

```
public class MyParentClass
{
    // members
}
```

Subclass

```
public class MySubClass : MyParentClass
{
    // inherited members from MyParentClass
}
```

Code reuse

Inheritance promotes code reuse

Minimize redundant code in subclasses

Don’t overuse inheritance

Parent class constructor

```
public class MyParentClass
{
    private int _myField;

    public MyParentClass(int myField)
    {
        _myField = myField;
    }
}
```

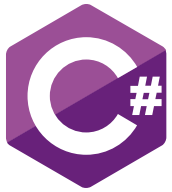
Subclass constructor

```
public class MySubClass : MyParentClass
{
    public MySubClass(int myField) : base(myField)
    {
    }
}
```

Demo

Create shapes

Virtual and override



Virtual members

A virtual member is a member marked with the virtual keyword

Can’t be applied on fields

Allow derived class to override a member with its own implementation

The derived member must use the override keyword to indicate that the member is overridden

An abstract method is implicitly a virtual method

Method override

A derived class can override a base class member only if the base class member is declared as virtual or abstract

Parent class

```
public class MyParentClass
{
    // members
    public virtual void DoSomething()
    {
        // Parent behavior
    }
}
```

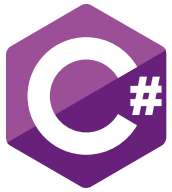
Subclass

```
public class MySubClass : MyParentClass
{
    // inherited members

    public override void DoSomething()
    {
        // subclass behavior
    }

    public void DoAnotherThing()
    {
        // subclass specialization
    }
}
```

Demo

Create and override virtual members

Abstract classes and methods



Abstract

Abstract = missing implementation

Can be used on classes, methods, properties, indexers and events

Abstract classes

- Marked with the abstract keyword
- Can’t be instantiated
- Used as base class, provides default state and behavior to subclasses
- Abstract members must be implemented by subclasses

Abstract class declaration

```
namespace MyNamespace
{
    public abstract class MyClass
    {
        // Class members
    }
}
```

Abstract methods

- Not implemented method signature marked with the abstract keyword
- Must be overridden in concrete classes that directly inherit from its containing class
- Abstract subclasses inherit abstract members without implementing them
- Can't be defined on non abstract class

Abstract method declaration

```
namespace MyNamespace
{
    public abstract class MyAbstractClass
    {
        public abstract void DoSomething();
    }
}
```

Method override

```
public class MyDerivedClass : MyAbstractClass
{
    public override void DoSomething()
    {
        // Method body
    }
}
```

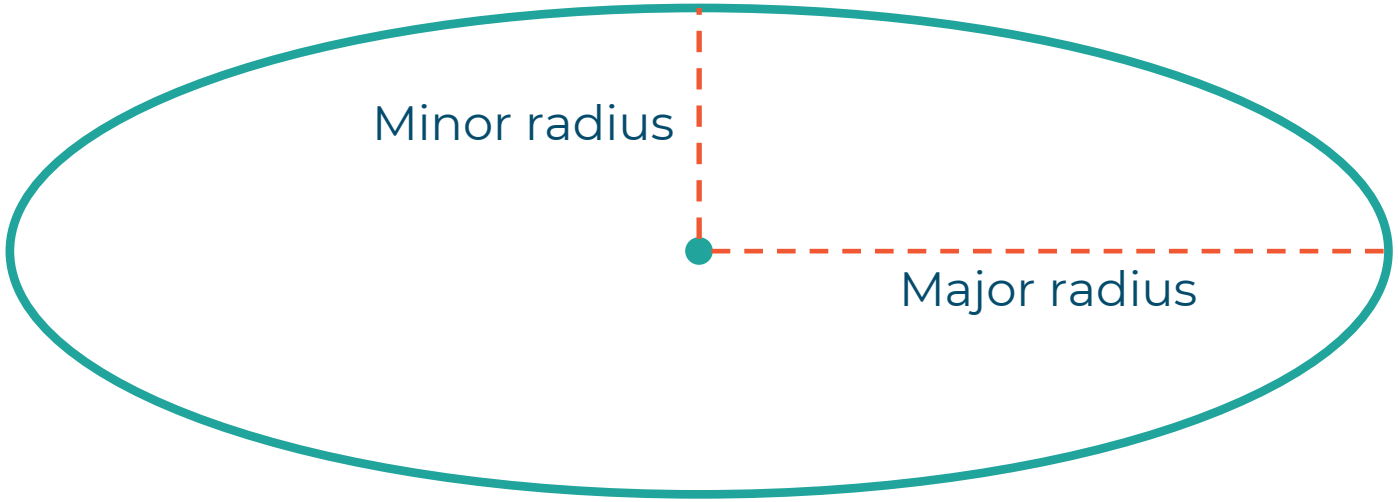
Abstract classes benefits

- Allow code reuse
- Establish an abstraction upon concrete classes
- Promote polymorphism
- Allow dynamic loading

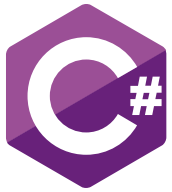
Demo

Create abstract classes and methods

Ellipsis



Sealed classes and methods



Sealed class

A class marked as sealed can’t be inherited

Sealed member

A derived class can stop member override by declaring an override as sealed by putting the sealed keyword before the override keyword in a class member declaration

Demo

Create sealed class and method

Summary

A class can inherit from another class and only one

A child class inherits the members of its parent class and extend it with new members

Virtual members can be overridden in the derived classes

Members can also be abstract, are not implemented and must be defined on abstract classes

A class can be abstract, can contain non abstract members

Abstract members can be overridden

A class marked as sealed can’t be inherited

A member marked as sealed can’t be overridden in the derived class