인터페이스와 추상 클래스

이것이 C#이다

Contents

8.1 인터페이스의 선언

❖ interface 키워드를 이용한 선언 형식

```
interface ILogger
{
    void WriteLog( string log );
}
```

❖ 특징

```
class ConsoleLogger : ILogger
{
    public void WriteLog(string message )
    {
        ILogger logger = new ConsoleLogger();
        logger.WriteLog( "Hello, World!" );
    }
}
```

8.2 인터페이스는 약속이다

- ❖ 클래스가 따라야 하는 약속
- ❖ 파생될 클래스가 어떤 메소드를 구현해야 할지 정의

```
class ClimateMonitor
{
    private ILogger logger;
    public ClimateMonitor(ILogger logger)
    {
        this.logger = logger;
    }

    public void start()
    {
        while ( true )
        {
            Console.Write( "온도를 입력해주세요.: " );
            string temperature = Console.ReadLine();
            if (temperature == "")
                break;

            logger.WriteLog( "현재 온도 : " + temperature );
        }
    }
}
```

```
ClimateMonitor monitor = new ClimateMonitor(new ConsoleLogger());
monitor.start();
```

```
class FileLogger : ILogger
{
    private StreamWriter writer;

    public FileLogger(string path)
    {
        writer = File.CreateText(path);
        writer.AutoFlush = true;
    }

    public void WriteLog(string message)
    {
        writer.WriteLine("{0} {1}", DateTime.Now.ToShortTimeString(), message);
    }
}
```

```
ClimateMonitor monitor = new ClimateMonitor(new FileLogger("MyLog.txt"));

monitor.start();
```

- ❖ 데모 예제 - Interface

8.3 인터페이스를 상속하는 인터페이스

- 기존 인터페이스에 새로운 기능을 추가한 인터페이스를 만들고 싶을 때
- 필요한 인터페이스가 어셈블리로만 제공되는 경우.
- 필요한 인터페이스를 상속한 클래스가 있는 경우
- 사용 형식 및 사용 예

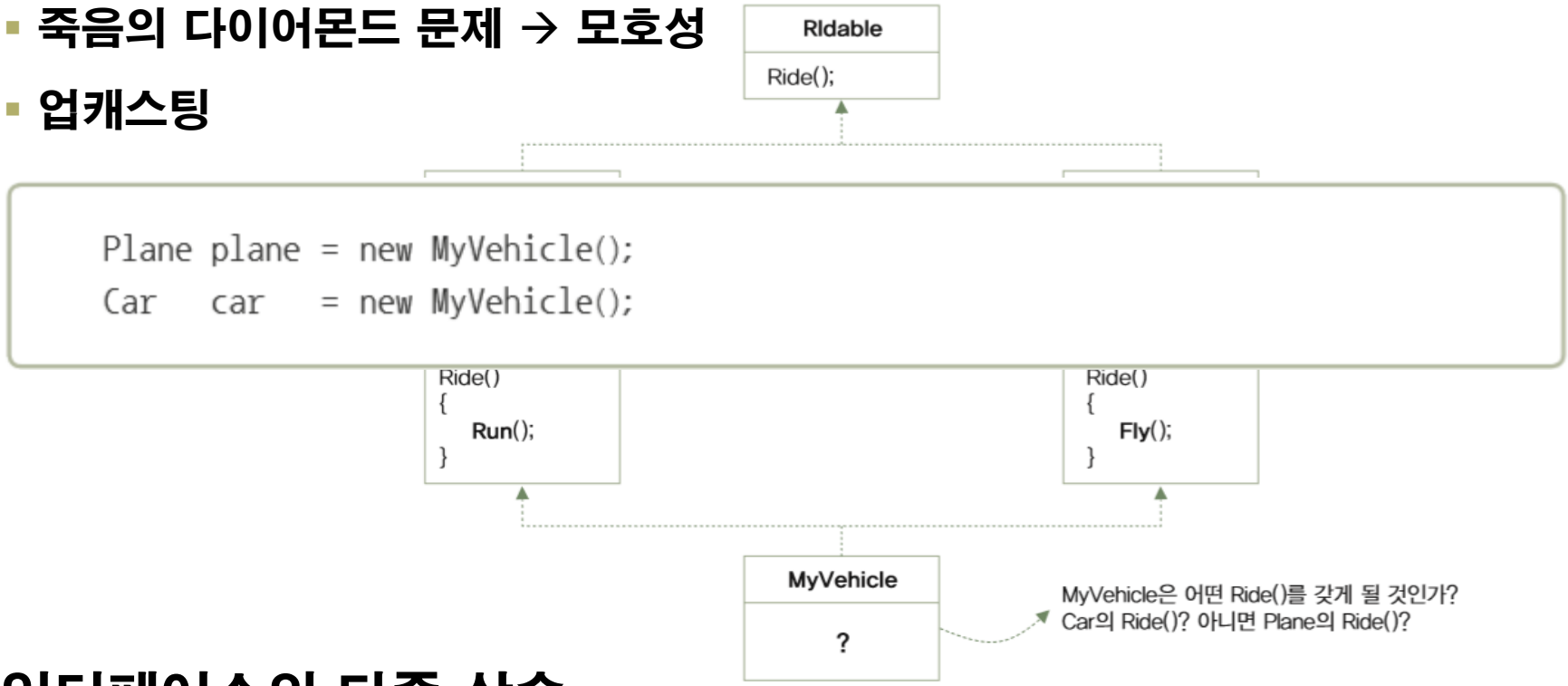
```
interface IFormattableLogger : ILogger
{
    void WriteLog(string format, params Object[] args);
}
```

- 데모 예제 - DerivedInterface

8.4 여러 개의 인터페이스, 한꺼번에 상속하기

❖ 클래스의 다중 상속이 갖는 문제

- 죽음의 다이어몬드 문제 → 모호성
- 업캐스팅



❖ 인터페이스의 다중 상속.

- 단순한 외형만 상속함으로 내부 구현은 상속자에게 맡김

❖ 데모 예제 – MultiInterfaceInheritance

8.5 추상 클래스: 인터페이스와 클래스 사이

- 구현은 갖되, 인스턴스는 갖지 못함.
- 선언 형식

```
abstract class 클래스이름
{
    // 클래스와 동일하게 구현
}
```

- 클래스의 접근성 사용
- 다른 추상 클래스 상속 가능
  - 자식 추상 클래스에서 부모의 추상 메소드 구현의무 없음

8.5 추상 클래스의 추상 메소드

- 추상 클래스이 인터페이스 역할을 위한 장치
- 파생 클래스에서 구현 필수
- 접근성
  - public, protected, internal, protected internal
- 추상 메소드 선언 예

```
abstract class AbstractBase
{
    public abstract void SomeMethod();
}

class Derived : AbstractBase
{
    public override void SomeMethod()
    {
        // Something
    }
}
```

- 데모 예제 - AbstractClass

Thank You !

이것이 C#이다