프로퍼티

이것이 C#이다

Contents

- public 필드의 유혹
- 메소드보다 프로퍼티
- 자동 구현 프로퍼티
- 프로퍼티와 생성자
- 무명 형식
- 인터페이스의 프로퍼티
- 추상 클래스와 프로퍼티

9.1 public 필드의 유혹

- 할당 연산자 =를 이용한 필드 액세스의 유혹
  - 데이터의 오염 가능성이 높아짐
- Get/Set 메소드를 사용한 필드 은닉
  - 번거롭고 귀찮음

```
MyClass obj = new MyClass();
obj.SetMyField( 3 );
Console.WriteLine( obj.GetMyField() );
```

- 은닉성과 편의성을 모두 잡는 방법 → 프로퍼티

9.2 메소드보다 프로퍼티

- Get/Set 접근자 , value 키워드
- 선언 형식 및 사용 예

```
class 클래스이름
{
    데이터형식 필드이름;
    접근한정자 데이터형식 프로퍼티이름
    {
        get
        {
            return 필드이름;
        }
        set
        {
            필드이름 = value;
        }
    }
}
```

```
class MyClass
{
    private int myField;
    public int MyField
    {
        get
        {
            return myField;
        }
        set
        {
            myField = value;
        }
    }
}
```

```
MyClass obj = new MyClass();
obj.MyField = 3 ;
Console.WriteLine( obj.MyField );
```

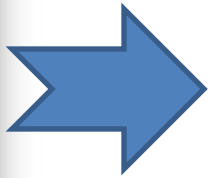
- 읽기전용 프로퍼티 – Get 접근자만 구현
- 데모 예제 – Property

9.3 자동 구현 프로퍼티

❖ 단순히 필드를 읽고 쓰기만 하는 경우

```
public class NameCard
{
    private string name;
    private string phoneNumber;

    public string Name
    {
        get { return name; }
        set { name = value; }
    }
```



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

    public string PhoneNumber
    {
        get; set;
    }
}
```

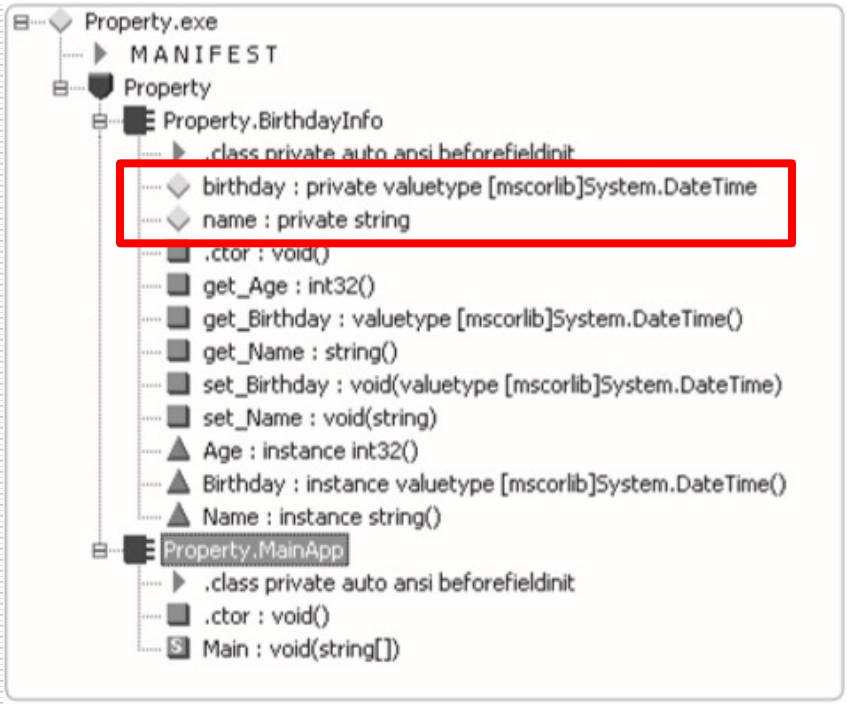
❖ 선언과 동시 초기화(7.0)

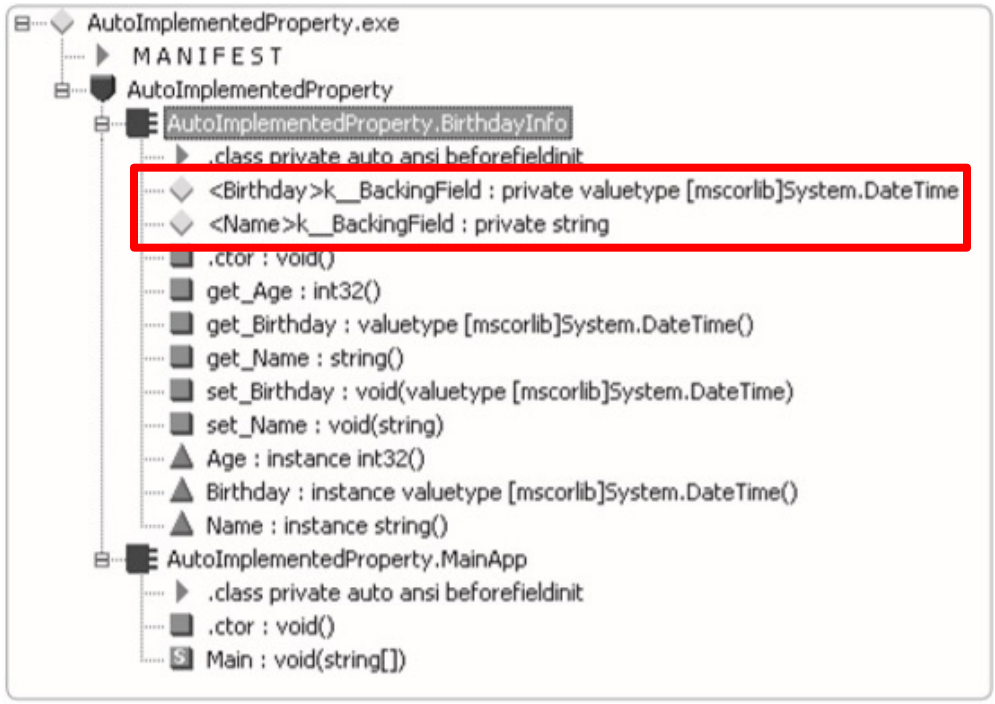
```
public class NameCard
{
    public string Name{ get; set; } = "Unknown";
    public string PhoneNumber{ get; set; } = "000-0000";
}
```

❖ 데모 예제 - AutoImplementedProperty

9.3.1 자동 구현 프로퍼티의 뒤에서 일어나는 일

❖ 직접 선언한 프로퍼티 vs 자동 구현 프로퍼티





❖ 컴파일러의 자동 선언 형식: <필드명>k_BackingField

9.4 프로퍼티와 생성자

- 객체 생성시 프로퍼티를 이용한 필드 초기화
- 필요한 프로퍼티만 초기화 가능
- 사용 형식

```
클래스이름 인스턴스 = new 클래스이름()
    {
        프로퍼티1 = 값,
        프로퍼티2 = 값,
        프로퍼티3 = 값
    };
```

세미콜론(;)이 아니라 콤마(,)입니다.

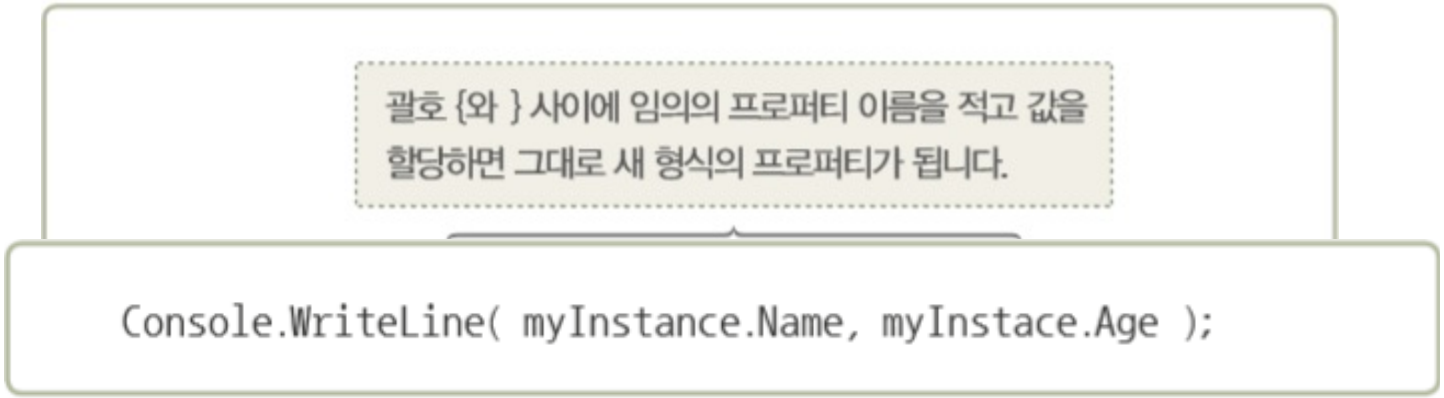
- 사용 예

```
BirthdayInfo birth = new BirthdayInfo()
        {
            Name = "서현",
            Birthday = new DateTime(1991, 6, 28)
        };
```

- 데모 예제 – ConstructorWithProperty

9.5 무명 형식

- 형식의 선언과 동시에 인스턴스 할당
- 인스턴스를 만들고 다시는 사 용하지 않을 때
- 선언 및 사용 예



- 프로퍼티에 할당된 값은 변경 불가
- 데모 예제 - AnonymousType

9.6 인터페이스의 프로퍼티

- 인터페이스의 프로퍼티 선언과 클래스의 자동 구현 프로퍼티 선언 동일
  - 상속하는 클래스에서 프로퍼티 구현
- 선언 형식 및 사용 예

```
interface 인터페이스이름
{
    public 형식 프로퍼티이름1
    {
        get; set;
    }

    // …
}
```

```
interface IProduct
{
    string ProductName
    {
        get;
        set;
    }
}
class Product : IProduct
{
    private string productName;

    public string ProductName
    {
        get{ return productName; }
        set{ productName = value; }
    }
}
```

파생 클래스는 부모 인터페이스에 선언된 모든 프로퍼티를 구현해야 합니다.

- 데모 예제 - PropertiesInInterface

9.7 추상 클래스와 프로퍼티

- ❖ 구현된 프로퍼티와 구현되지 않은 프로퍼티
- ❖ 구현되지 않은 프로퍼티
  - abstract 한정자 사용 → 구현 필수
- ❖ 선언 형식

```
abstract class 추상 클래스이름
{
    abstract 데이터형식 프로퍼티이름
    {
        get;
        set;
    }
}
```

```
abstract class Product
{
    private static int serial = 0;
    public string SerialID
    {
        get { return String.Format("{0:d5}", serial++); }
    }

    abstract public DateTime ProductDate
    {
        get;
        set;
    }
}
class MyProduct : Product
{
    public override DateTime ProductDate
    {
        get;
        set;
    }
}
```

추상 클래스는 구현을 가진 프로퍼티와

구현이 없는 추상 프로퍼티 모두를 가질 수 있습니다.

파생 클래스는 부모 추상 클래스의 모든 추상 메소드뿐 아니라 추상 프로퍼티를 재정의해야 합니다.

- ❖ 데모 예제 - PropertiesInAbstractClass

Thank You !

이것이 C#이다