항목 33: 상속된 이름을 숨기는 일은 피하자

변수와 함수 모두 유효범위(scope)가 있습니다. 다음 코드를 보세요.

*int x; // 전역 변수*

*void someFunc( )*

*{*

*double x; // 지역 변수*

*std::cin >> x; // 입력을 받아, 지역 변수 x에 새 값을 읽어 넣습니다.*

*}*

위 코드를 실행 했을 때 과연 어떤 x에 값이 채워 질까요? 정답은 double x 지역 변수 입니다. 이유는 **컴파일러가 someFunc함수에서 지역 변수 x라는 이름을 만나서 사용하려고 하면, 현재 처리하고 있는 유효범위(local scope)내에서 같은 이름을 가진 것이 있는지 찾습니다. 찾다가 같은 이름으로 선언 된 것을 발견하면 더 이상 탐색하지 않습니다.** 어떤 타입으로 선언 되었는지는 관계 없습니다.

**C++의 이름 가리기 규칙은 이름을 가려 버립니다**. 위와 같은 경우에는 지역변수 double x가 전역변수 int x를 가려 버린 것입니다.

이제 상속으로 넘어가 보도록 하겠습니다. 알다시피 상속을 받으면 파생 클래스는 기본 클래스에 선언된 것을 모두 물려받기 때문에 부모 클래스의 변수 혹은 함수를 찾아낼 수 있습니다.

*class Base {*

*private:*

*int x;*

*public:*

*virtual void mf1( ) = 0;*

*virtual void mf2( );*

*void mf3( );*

*. . .*

*};*

*class Derived: public Base {*

*public:*

*virtual void mf1( );*

*void mf4( );*

*. . .*

*};*

*void Derived::mf4( ) // 파생 클래스 함수*

*{*

*mf2( );*

*}*

파생 클래스 함수 mf4가 호출 되면 컴파일러는 mf2라는 이름을 만나면 우선 지역범위에서 mf2를 찾고 없으면 범위를 넓혀서 Derived 클래스 내에서 찾아봅니다. 하지만 여전히 mf2라는 이름을 찾지 못하여 범위를 더 넓혀서 기본 클래스의 유효 범위에서 찾습니다. 비로소 탐색이 끝나는 것 입니다.

*class Base {*

*private:*

*int x;*

*public:*

*virtual void mf1( ) = 0;*

*virtual void mf1(int);*

*virtual void mf2( );*

*void mf3( );*

*void mf3(double);*

*. . .*

*};*

*class Derived: public Base {*

*public:*

*virtual void mf1( );*

*void mf3( );*

*void mf4( );*

*. . .*

*};*

위 코드에서 Derived클래스는 Base의 파생 클래스 이지만 Base의 어떤 함수는 호출 할 수 없습니다. 기본 클래스의 함수 mf1과 mf3은 모두 Derived클래스의 함수 mf1과 mf3에 의해 가려지고 맙니다.

예를 들어 다음과 같이 사용할 수 없습니다.

*Derived d;*

*int x;*

*d.mf1( ); // 사용 가능 Derived::mf1을 호출 합니다.*

*d.mf1( x ); // 에러! Derived::mf1이 Base::mf1을 가립니다.*

*d.mf2( ); // 좋습니다. Base::mf2를 호출 합니다.*

*d.mf3( x ); // 에러! Derived::mf3가 Base::mf3을 가립니다.*

***위와 같은 문제를 해결하려면 using 선언을 사용할 수 있습니다.***

*class Derived: public Base {*

*public:*

***using Base::mf1;*** *// Base에 있는 것들 중 mf1과 mf3을 이름으로 가진 것들을*

***using Base::mf3;*** *// Derived의 유효범위에서 볼 수 있도록(또 public 멤버로) 만듭니다.*

*. . .*

*}*

public 상속은 is – a 관계 입니다. 어떤 파생 클래스가 기본 클래스로부터 public 상속으로 만들어진 경우, **기본 클래스의 public 영역에 있는 이름들은 파생 클래스에서도 public 영역에 들어 가야 합니다.**

이름 가리기로 인하여 어떤 함수를 상속 받지 못하면 is – a 관계 위반이 되고 진정한 public 상속이 아니게 됩니다. 따라서 상속된 이름을 숨기는 일은 피하도록 합니다.

**만약 private 상속을 하여 특정 함수만 상속했을 경우에는** **전달 함수를 만들면 됩니다.** 예제를 보세요.

*class Base {*

*public:*

*virtual void mf1( ) = 0;*

*virtual void mf1( int );*

*};*

*class Derived: private Base {*

*public:*

*virtual void mf1( ) // 전달 함수*

*{*

*Base::mf1( );*

*}*

*};*

위 Derived 파생 클래스는 private 상속이고 기본 클래스의 mf1만 상속 받고 싶어 합니다. 이럴 때는 전달 함수를 사용해야 합니다. 만약 using Base::mf1;을 사용하게 되면 mf1에 해당 하는 모든 함수(오버로딩 된 것들)들도 상속 받게 되어 using문은 쓸 수 없습니다.