항목 36: 상속받은 비가상 함수를 파생 클래스에서 재정의하는 것은 절대 금물!

왜 비가상 함수를 파생 클래스에서 재정의 하면 안 되는지 알아봅니다. 먼저 다음 코드를 보세요.

*class B {*

*public:*

*void mf( );*

*. . .*

*};*

*class D: public B {*

*public:*

***void mf( ); // 기본 클래스와 같은 이름의 함수 재정의***

*. . .*

*};*

*D objD; // D 객체 생성*

*B\* pB = &objD; // B포인터로 D객체를 가리킵니다.*

*pB -> mf( ); // B포인터로 mf함수를 호출 합니다.* ***B::mf 가 호출 됩니다.***

*D\* pD = &objD; // D포인터로 D객체를 가리킵니다.*

*pD -> mf( ); // D포인터로 mf함수를 호출 합니다.* ***D::mf 가 호출 됩니다.***

D 객체를 만들고 포인터를 달리하여 같은 이름의 함수 mf를 호출 하였습니다. **결과 모두 같은 D객체를 가리키지만 호출되는 mf함수는 달랐습니다.** **같은 객체 D에서 시작 했지만 다른 함수가 호출 된 것은 mf 함수가 비가상 함수로 만들어 졌기 때문입니다.**

위와 같은 코드에서 pB -> mf( ); 함수를 호출 할 때 D::mf 함수가 호출 되도록 만들려면 가상 함수로 만들어야 합니다. 가상 함수는 동적 바인딩으로 묶이기 때문에 기본클래스 포인터이든 파생클래스 포인터이든 상관없이 만들어진 객체를 따라갑니다.

**비가상 함수를 재정의 해서 사용하게 되면 포인터에 따라 호출되는 함수가 달라져서 어떤 함수가 호출될지 모르게 됩니다.**

1. 바인딩이란, 프로그램 소스에 쓰인 각종 내부 요소, 이름, 식별자들에 대해 값 혹은 속성을 확정하는 과정을 일컫는다. 이 과정이 빌드 중에 이루어지면 정적 바인딩이라 하고, 실행 중에 이루어지면 동적 바인딩이라고 한다.

예를 들어 “int foo = 2;”이 있을 때 데이터 타입이 int로 정해지는 것과 그 타입의 변수명이 foo로 정해지는 것은 정적 바인딩이며, foo 변수에 2가 대입되는 것은 동적 바인딩이다.