

# 6장 응용 프레임워크와 가상함수의 만남

변영철 교수

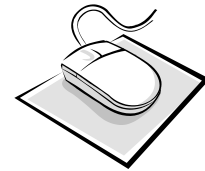
([ycb@jejunu.ac.kr](mailto:ycb@jejunu.ac.kr))

# 1. 응용 프로그램들이 비슷한 이유

- 윈도우 특징
  - GUI
  - 멀티태스킹
- 응용 프로그램들이 서로 모양이 비슷
  - 외부
  - 내부
- 중복되는 코드가 많이 존재
  - 기본적으로 혹은 라이브러리로... (뼈대)

# 1. 응용 프로그램들이 비슷한 이유

- MFC 응용 프로그램 작성의 예
  - 기본적으로 많은 코드가 작성되어 있다.



- 여러분이 할 일은?
  - 기존에 작성되어 있는 코드를 바꾸거나
  - 새로운 코드를 추가하거나
  - > 뼈대에 살을 붙이는 것

## 2.Console4 프로젝트 생성

- CBase 클래스 작성하기

```
#include <stdio.h>
```

```
class CBase
{
public:
    void WhoAreYou()
    {
        printf("CBase!\n");
    }
};
```

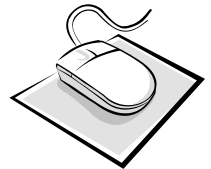


**CBase gildong;**

```
void main()
{
    gildong.WhoAreYou();
}
```

# 3.CBase를 항상 작성해야 한다면

- 반복해서 CBase 클래스를 작성하는 것이 귀찮다면 라이브러리로 만들라.
  - CBase 클래스를 라이브러리로 생각하자.
  - 라이브러리는 여러 사람이 필요할 때마다 사용하는 것이므로 수정할 수 없다.
  - 라이브러리는 (일반적으로) \*.dll, \*.lib 형태로 저장되어 소스 코드가 보이지 않는다.
  - “수정할 수도 없고 보이지도 않는다.”



```
#include <stdio.h>
```

```
class CBase  
{  
public:  
    void WhoAreYou()  
    {  
        printf("CBase!\n");  
    }  
};
```

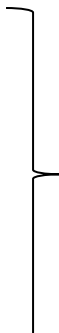
라이브러리 코드  
(기존에 있었던 코드)



**CBase gildong;**

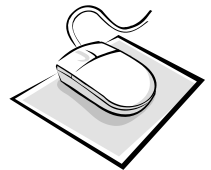
```
void main()  
{  
    gildong.WhoAreYou();  
}
```

라이브러리가 아닌 코드  
(여러분이 작성하는 코드)



## 4.마음에 들지 않으면?

- 이 프로그램을 실행하면??
- 화면에 표시되는 문자열이 마음에 들지 않으면?
- CBase는 라이브러리라고 가정하였으므로 코드가 마음에 들지 않아도 수정할 수 없다.
- OOP에서 마음에 들지 않은 코드를 직접 고치지 않으면서 고치는 세련된 방법 : 함수 재정의 (5장에서 공부한 내용)



```
#include <stdio.h>
```

```
class CBase {  
public:  
    void WhoAreYou()  
    {  
        printf("CBase!\\n");  
    }  
};
```

```
class CDerived : public CBase {  
public:  
    void WhoAreYou()  
    {  
        printf("-----\\n");  
        printf("Hello, I am a CBase!\\n");  
        printf("-----\\n");  
    }  
};
```

```
CDerived gildong;
```

```
void main()  
{  
    gildong.WhoAreYou();  
}
```

마음에 들지 않으니 제정의

마음에 들었다면? 재정의하  
지 않았을 것!



## 5.상상력 발휘하기

- main 함수가 없는 프로그램은 없다.
  - 따라서 main는 반드시 존재해야 함
  - 반드시(항상) 존재해야(작성해야) 한다면 main 함수도 라이브러리로 되어 있으면 좋을 듯

```
void main()  
{  
    gildong.WhoAreYou();  
}
```

```
#include <stdio.h>
```

```
class CBase  
{  
public:  
    void WhoAreYou()  
    {  
        printf("CBase!\n");  
    }  
};
```

라이브러리 코드  
(기존에 있었던 코드)

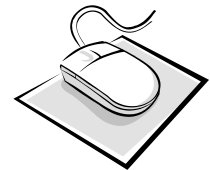
**CBase gildong;**

```
void main()  
{  
    gildong.WhoAreYou();  
}
```

라이브러리가 아닌 코드  
(여러분이 작성하는 코드)

## 5.상상력 발휘하기

- 라이브러리로 존재하는 코드 상상하기
  - CBase 클래스
  - main 함수



## 5.상상력 발휘하기

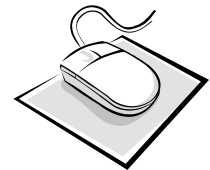
- gildong이라는 변수 이름은 누가 정하나?
- 만일 변수를 cheolsu라고 하면?
- 따라서 프로그래머가 정하는 변수 이름 때문에 main 함수를 라이브러리화 하는 것은 어색함.
- 변수 이름을 사용자가 맘대로 정하도록 하려면?

# 5.상상력 발휘하기

- 포인터 이용하기

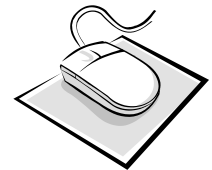
Afx = Application Framework

- AfxGetApp 함수 : CBase 클래스로 만든 전역변수 주소 값을 알아내는 함수



## 5.상상력 발휘하기

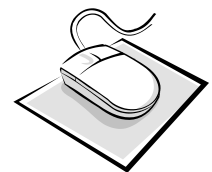
- 라이브러리로 존재하는 코드 상상하기
  - CBase 클래스
  - main 함수
  - AfxGetApp 함수



## 6.Is-a 관계

- 타원과 원 관계
  - 타원은 원이다. Ellipse is a Circle.
  - 남자는 사람이다. Man is a Human.
  - 왼쪽 것은 오른쪽 것이다.
  - 아래 것은 위의 것이다.
- 특화(specialization)와 일반화(generalization)

```
void main()
{
    CBase* p;
    p = AfxGetApp();
    p->WhoAreYou();
}
```



## 7.가상함수의 위력

- 라이브러리화 하였지만 실행되지 않는다.

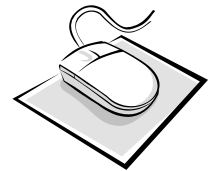


- 여기에 **virtual 키워드**를 붙이는 순간
- 아주 **놀라운 한 줄짜리 프로그램**
  - 눈에 보이지 않는 코드는?



# 7.가상함수의 위력

- 한 줄짜리 MFC 프로그램의 놀라움
  - 눈에 보이지 않는 코드는?
- 코드가 마음에 들지 않으면? 무엇인가를 표시하려면?



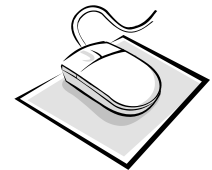
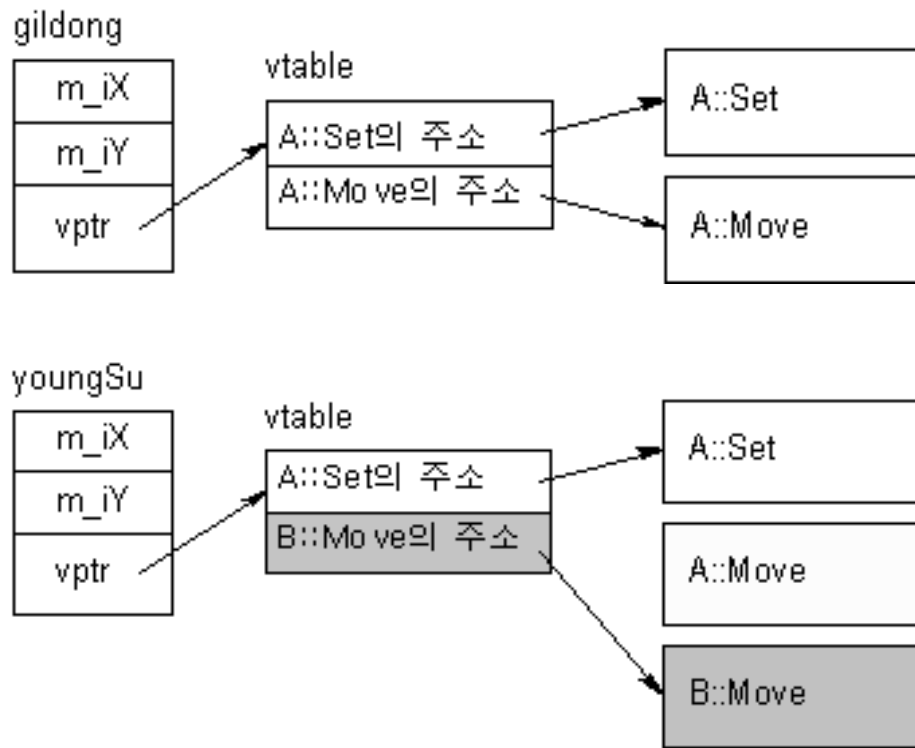
```
#include <afxwin.h>
```

```
class CDerived : public CWinApp  
{  
    public:  
        BOOL InitInstance() {  
            AfxMessageBox(_T("Hello, World!"));  
            return TRUE;  
        }  
};
```

```
CDerived gildong;
```

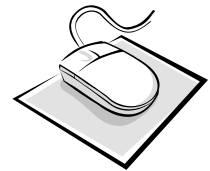
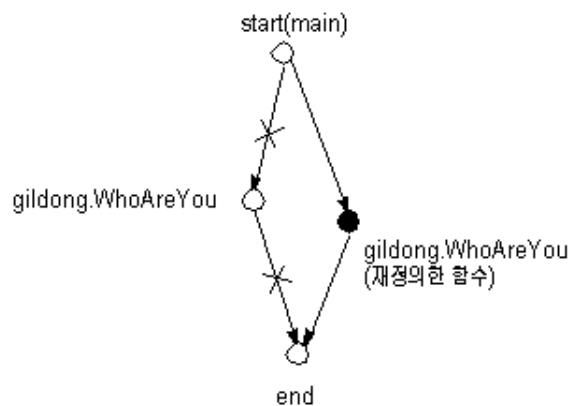
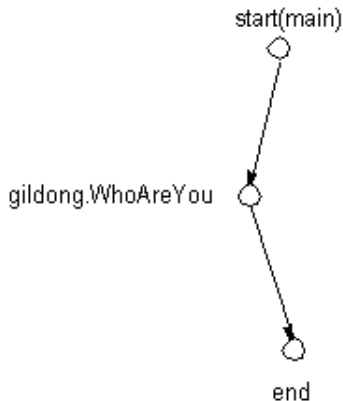
# 8.가상함수 구현방법

- 가상함수 테이블



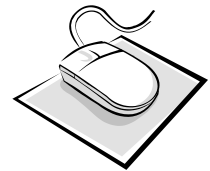
## 9.가상함수와 응용 프레임워크

- 응용 프레임워크는
  - 한 줄짜리 프로그램에 숨겨져 있는(!) 거대한 코드 덩어리
  - 가상함수 재정의로 응용 프레임워크 실행 흐름을 원하는 대로 바꿀 수 있다.



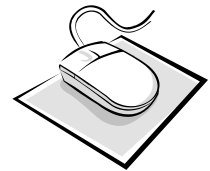
# 10.코드 분할 및 정리

- Lib.h
  - CBase 클래스
- Lib.cpp
  - main
  - AfxGetApp
- Console5.cpp
  - 한 줄짜리 코드 (객체 정의)



# 11.MFC 라이브러리 이용하기

- MFC 라이브러리에 이미 있는 코드들
  - CWinApp
  - main
  - AfxGetApp
- 프로젝트 속성 설정하기



## 12.객체를 보라

- 클래스를 보지 말고 객체를 보라.
  - 클래스 지향이 아닌 객체 지향 프로그래밍