모듈과 라이브러리

제주대학교 컴퓨터공학과
변영철 교수
(ycb@jejunu.ac.kr)

이 장을 공부하면

- 내가 작성한 프로그램에서 일부를 클래스 부품으로 만들 수 있다.
- 클래스 부품을 라이브러리 폴더로 모아 둘 수 있다.
- 내가 만든 라이브러리를 쉽게 이용(재사용)할 수 있다.

가장 간단한 프로그램 작성

- 새로운 프로젝트 My 생성
- 기존의 코드를 삭제한 후 다음 코드 작성

```
class My
{
    public static void Main()
    {
        System.Console.WriteLine("Hello,World!");
    }
}
```

코드 추상화

- 멤버 함수 Say로 추상화

```
class My
{
    public void Say()
    {
        System.Console.WriteLine("Hello,World!");
    }

    public static void Main()
    {
        My gildong = new My();
        gildong.Say();
    }
}
```

새로운 클래스 Hello 작성 후 함수 이동

```
class Hello
{
    public void Say()
    {
        System.Console.WriteLine("Hello,World!");
    }
}

class My
{
    public static void Main()
    {
        Hello gildong = new Hello();
        gildong.Say();
    }
}
```

새로운 파일 추가 후 클래스 이동

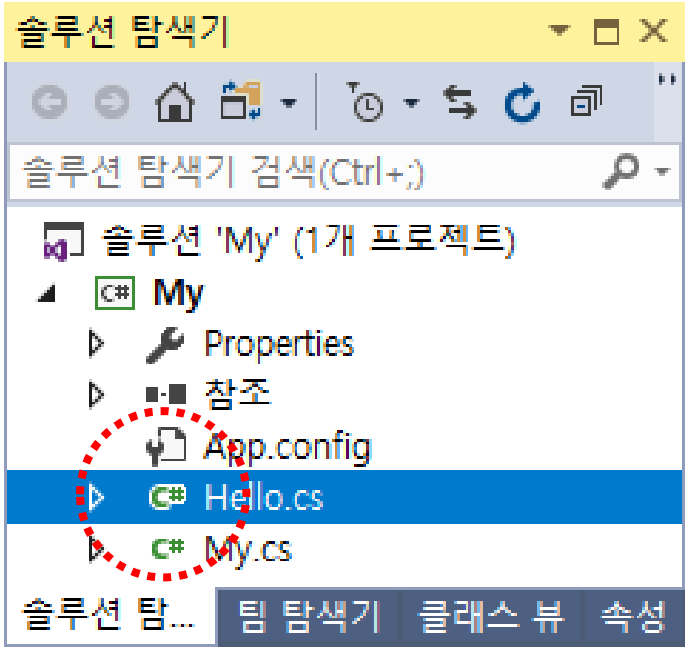
- 프로젝트 | 새 항목 추가
- '클래스' 선택한 후 파일 이름 Hello.cs 입력
- 자동으로 작성된 모든 코드 삭제 후 이곳으로 Hello 클래스 이동

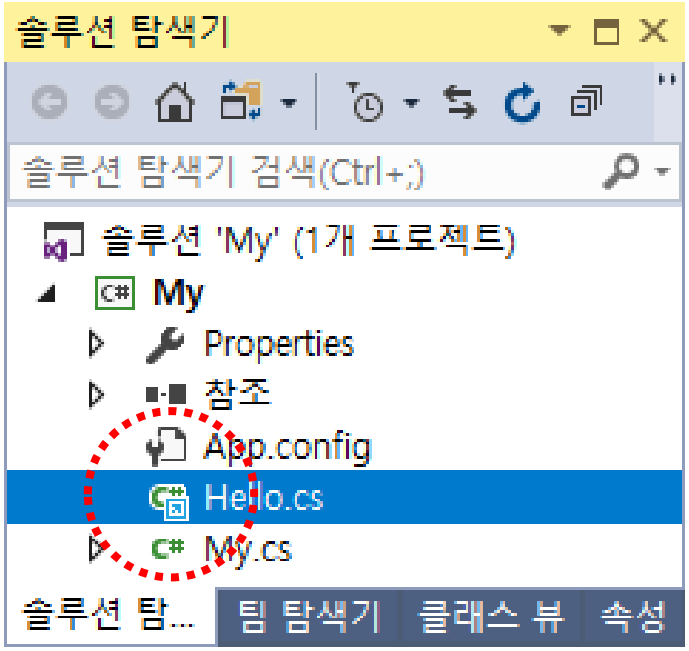
```
class Hello
{
    public void Say()
    {
        System.Console.WriteLine("Hello,World!");
    }
}
```

라이브러리화

- C 루트에 라이브러리 폴더(cslib) 생성
- 금방 만든 Hello.cs 파일을 라이브러리 폴더로 이동
  - 파일 | 다른 이름으로 저장 메뉴 선택
  - cslib에 Hello.cs 파일 저장

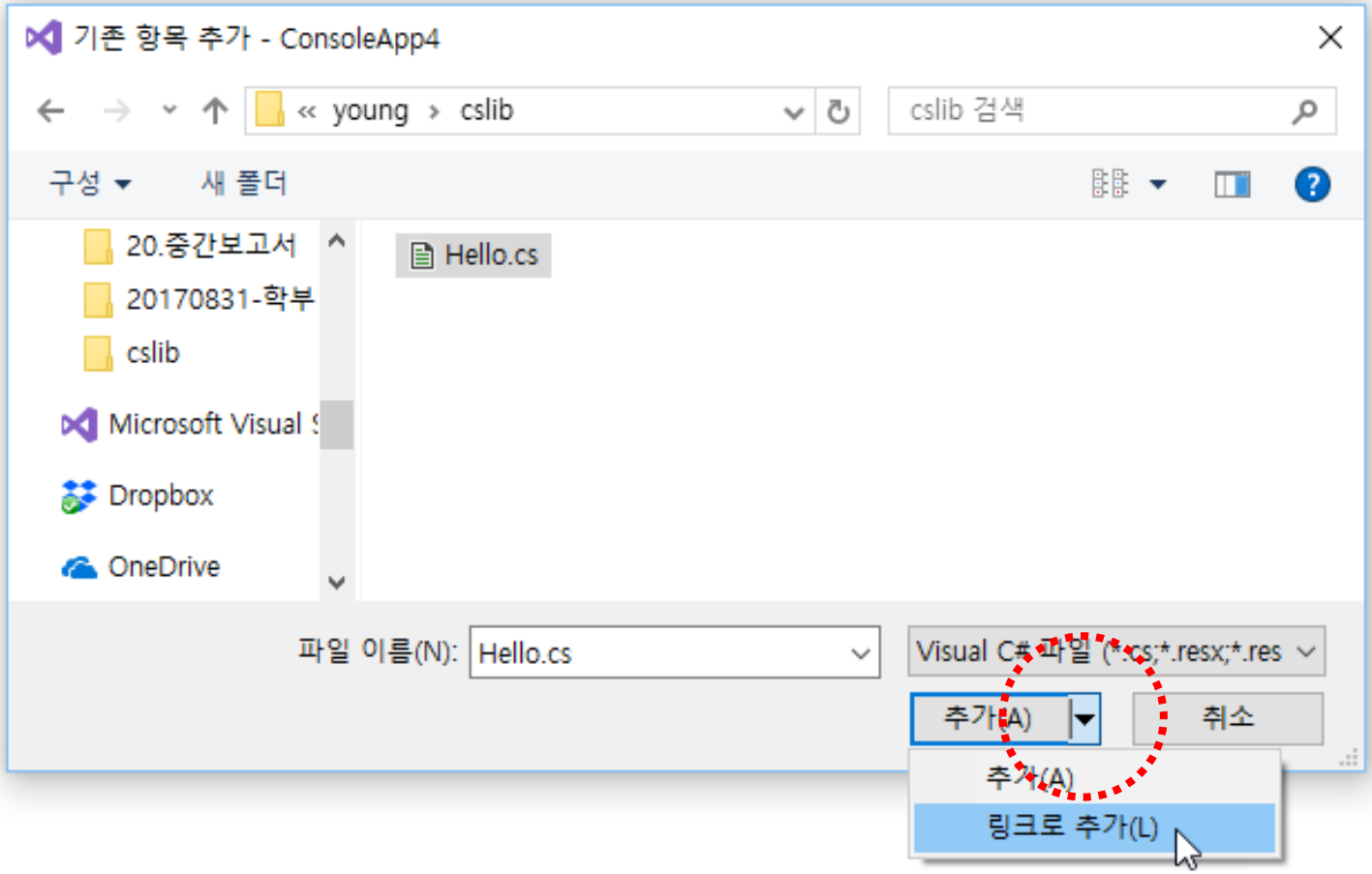
라이브러리화


솔루션 탐색기
솔루션 탐색기 검색(Ctrl+;)
솔루션 'My' (1개 프로젝트)
My
Properties
참조
App.config
Hello.cs
My.cs
솔루션 탐...
팀 탐색기
클래스 뷰
속성


솔루션 탐색기
솔루션 탐색기 검색(Ctrl+;)
솔루션 'My' (1개 프로젝트)
My
Properties
참조
App.config
Hello.cs
My.cs
솔루션 탐...
팀 탐색기
클래스 뷰
속성

라이브러리 이용하기

- 새로운 프로젝트 생성
- 프로젝트 | 기존 항목 추가...
  - clib 폴더로 이동하여 Hello.cpp 링크로 추가
  - 드롭다운 버튼 클릭 후 링크로 추가


기존 항목 추가 - ConsoleApp4
« young › cslib
cslib 검색
구성
새 폴더
20.중간보고서
20170831-학부
cslib
Microsoft Visual S
Dropbox
OneDrive
Hello.cs
파일 이름(N):
Hello.cs
Visual C# 파일 (*.cs;*.resx;*.res
추가(A)
취소
추가(A)
링크로 추가(L)

두번째 예

- 멤버는 멤버를 액세스 할 수 있다.
- 추상화하고 싶은 부분

```
class MainForm //gildong
{
    public int Font;
    public void CreateGraphics()
    {
    }
    public void MainForm_Click()
    {
        CreateGraphics();
        Font = 1;
        System.Console.WriteLine("Hello!");
    }
}
class My
{
    public static void Main()
    {
    }
}
```

두번째 예

- 멤버는 멤버를 액세스 할 수 있다.
- 추상화하고 싶은 부분

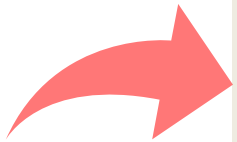
```
class MainForm //gildong
{
    public int Font;
    public void CreateGraphics()
    {
    }
    public void MainForm_Click()
    {
        this.CreateGraphics();
        this.Font = 1;
        System.Console.WriteLine("Hello!");
    }
}
class My
{
    public static void Main()
    {
    }
}
```

코드추상화

```
class MainForm //gildong
{
    public int Font;
    public void CreateGraphics()
    {
    }
    public void Display() {
        this.CreateGraphics();
        this.Font = 1;
        System.Console.WriteLine("Hello!");
    }
    public void MainForm_Click()
    {
            Display();
    }
}
class My
{
    public static void Main()
    {
    }
}
```

다른 클래스로

다른 클래스로
옮겼을 때
에러가 나는 곳은?
해결 방법은?



```
class MyUtil
{
    public void Display() {
        this.CreateGraphics();
        this.Font = 1;
        System.Console.WriteLine("Hello!");
    }
}
class MainForm
{
    public int Font;
    public void CreateGraphics() {
    }
    public void MainForm_Click()
    {
        MyUtil cheolsu = new MyUtil();
        cheolsu.Display();
    }
}
```

다른 클래스로

```
class MyUtil
{
    public void Display(MainForm mf) {
        mf.CreateGraphics();
        mf.Font = 1;
        System.Console.WriteLine("Hello!");
    }
}
class MainForm
{
    public int Font;
    public void CreateGraphics() {
    }
    public void MainForm_Click()
    {
        MyUtil cheolsu = new MyUtil();
        cheolsu.Display(this);
    }
}
```

HAS-A 관계

사람은 핸드폰을
갖는다.

MainForm객체는
MyUtil객체를 갖는다.

```
class MyUtil
{
    public void Display(MainForm mf) {
        mf.CreateGraphics();
        mf.Font = 1;
        System.Console.WriteLine("Hello!");
    }
}
class MainForm
{
    MyUtil cheolsu = new MyUtil();
    public int Font;
    public void CreateGraphics() {
    }
    public void MainForm_Click()
    {
        cheolsu.Display(this);
    }
}
```

상속되는 코드

```
class Form
{
    public int Font;
    public void CreateGraphics() {
    }
}
class MainForm : Form
{
    public void MainForm_Click()
    {
        this.CreateGraphics();
        this.Font = 1;
        System.Console.WriteLine("Hello!");
    }
}
class My
{
    public static void Main() {
    }
}
```

- 눈에 보이는 멤버만이 멤버가 아니다.
- 추상화하고 싶은 부분
- 뭐라고 추상화할까???

세번째 예

- 클래스 끼워 넣기!

```
class Form
{
    public int Font;
    public void CreateGraphics() {
    }
}
class MyForm : Form
{
}
class MainForm : MyForm
{
    public void MainForm_Click()
    {
        this.CreateGraphics();
        this.Font = 1;
        System.Console.WriteLine("Hello!");
    }
}
class My
{
    public static void Main() {
    }
}
```

코드 추상화

```
class Form
{
    public int Font;
    public void CreateGraphics() {
    }
}
class MyForm : Form
{
}
class MainForm : MyForm
{
    public void Display() {
        this.CreateGraphics();
        this.Font = 1;
        System.Console.WriteLine("Hello!");
    }
    public void MainForm_Click()
    {
        Display();
    }
}
```

위로 이동

```
class Form
{
    public int Font;
    public void CreateGraphics() {
    }
}
class MyForm : Form
{
    public void Display() {
        this.CreateGraphics();
        this.Font = 1;
        System.Console.WriteLine("Hello!");
    }
}
class MainForm : MyForm
{
    public void MainForm_Click()
    {
        Display();
    }
}
```

IS-A 관계

사람은 동물이다.

아랫것은 윗것이다.
MainForm은
MyForm이다.

```
class Form
{
    public int Font;
    public void CreateGraphics() {
    }
}
class MyForm : Form
{
    public void Display() {
        this.CreateGraphics();
        this.Font = 1;
        System.Console.WriteLine("Hello!");
    }
}
class MainForm : MyForm
{
    public void MainForm_Click()
    {
        Display();
    }
}
```

모듈만드는 방법 2가지

- 완전히 별도의 모듈로 만든 후 그곳으로 코드 이동
  - HAS-A 관계
- 상속 관계 사이에 끼이는 모듈을 만든 후 그곳으로 코드 이동
  - IS-A 관계

나만의 라이브러리, 왜 좋을까?

- 나중에 쉽게 재사용(함수 호출) 할 수 있다.
- 코드가 어떻게 돌아가는지 잊어버려도 문제없다(블랙박스).
- 쉽게 구현할 수 있으므로 프로그래밍이 부담스럽지 않다.

요 약

- 내가 작성한 프로그램에서 일부를 클래스 부품으로 만들 수 있다.
- 클래스 부품을 라이브러리 폴더로 모아 둘 수 있다.
- 내가 만든 라이브러리를 쉽게 이용(재사용)할 수 있다.