

Chapter 02. 미리 만들어보는 C# 프로그램

목차

1. 이 장에서 만들 프로그램
2. 계산기 프로그램 기본 기능 구현
3. 계산기 프로그램 기능 확장
4. 그림판 프로그램 작성

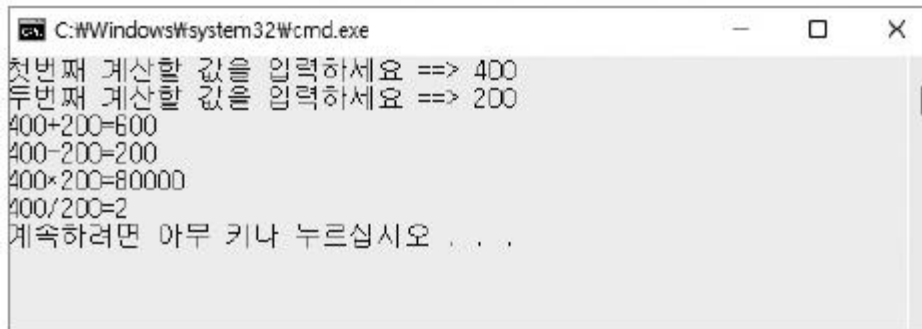
01

이 장에서 만들 프로그램

01. 이 장에서 만들 프로그램

I. [프로그램 1] 간단 계산기

- 첫 번째 프로그램은 두 숫자를 입력하면 덧셈, 뺄셈, 곱셈, 나눗셈, 나눗셈의 몫과 나머지의 결과가 나오는 프로그램이다.
- 계산기에 2개의 숫자를 직접 입력 하면 일반 계산기와 비슷하게 계산 결과가 나온다.



```
C:\Windows\system32\cmd.exe
첫번째 계산할 값을 입력하세요 ==> 400
두번째 계산할 값을 입력하세요 ==> 200
400+200=600
400-200=200
400*200=80000
400/200=2
계속하려면 아무 키나 누르십시오 . . .
```

그림 2-1 간단 계산기 출력 결과

01. 이 장에서 만들 프로그램

II. [프로그램 2] 간단 그림판

- 두 번째로 만들 프로그램은 원폼 프로그램으로, 원폼(WinForm)은 GUI(Graphical User Interface)를 제공하는 C#의 기능이다.
- 윈도우 창에서 마우스를 이용해 선을 그리는 프로그램이며, 창 왼쪽 상단의 버튼을 클릭하면 기존의 선들이 모두 없어진다.

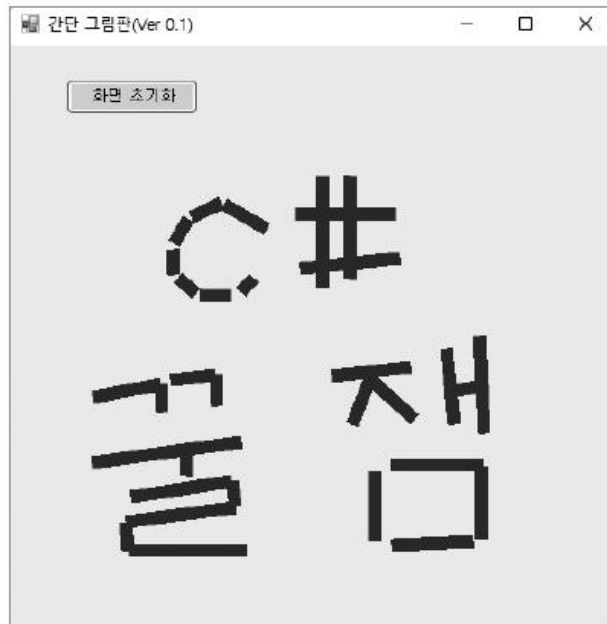


그림 2-2 간단 그림판 출력 결과

02

계산기 프로그램 기본 기능 구현

02. 계산기 프로그램 기본 기능 구현

I. 프로그램 작성 순서

- 프로그램의 작성 순서:



그림 2-3 프로그램 작성 순서

02. 계산기 프로그램 기본 기능 구현

II. 기본적인 콘솔 앱 생성

- 이번 프로젝트의 이름은 **Project02_1**로 한다.

[실습 2-1] 간단 계산기 프로젝트 생성하기

- ① Visual Studio를 실행한다.
- ② 프로젝트를 생성하기 위해 실행 창 오른쪽의 [새 프로젝트 만들기]를 클릭한다.

02. 계산기 프로그램 기본 기능 구현

II. 기본적인 콘솔 앱 생성

[실습 2-1] 간단 계산기 프로젝트 생성하기

- ③ [새 프로젝트 만들기] 창에서 언어를 [C#]으로 선택한다. 창 아래로 스크롤해 [콘솔 앱(.NET Framework)]을 선택한 후 을 클릭한다.



그림 2-4 Project02_1 새 프로젝트 생성 1

02. 계산기 프로그램 기본 기능 구현

II. 기본적인 콘솔 앱 생성

[실습 2-1] 간단 계산기 프로젝트 생성하기

- ④ [새 프로젝트 구성] 창에서 '프로젝트 이름'에는 **Project02_1**을 입력한 후, '위치'에서 <...>을 클릭해 **C:\WCookC#** 폴더를 선택하자. 그리고 <솔루션 및 프로젝트를 같은 디렉터리에 배치>를 체크한 후 <만들기>를 클릭한다.

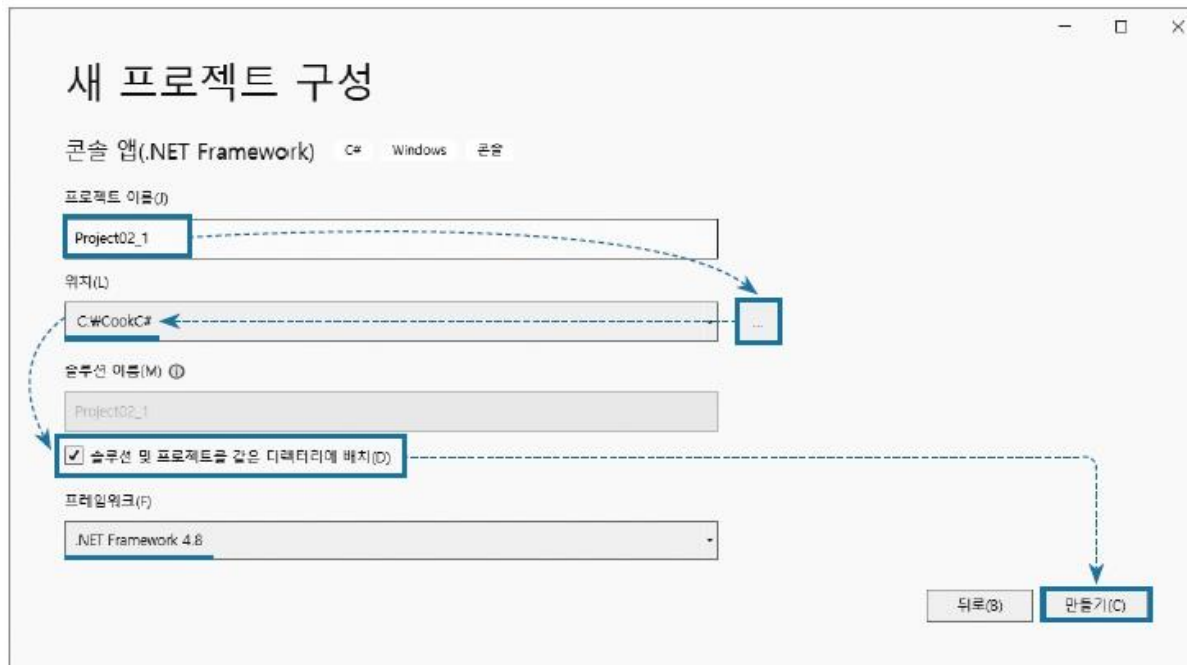


그림 2-5 Project02_1 새 프로젝트 생성 2

02. 계산기 프로그램 기본 기능 구현

II. 기본적인 콘솔 앱 생성

[실습 2-1] 간단 계산기 프로젝트 생성하기

- ⑤ 최종적으로 다음과 같은 프로젝트(또는 솔루션)가 완성되었다. 이 프로젝트의 이름은 Project02_1 이다.

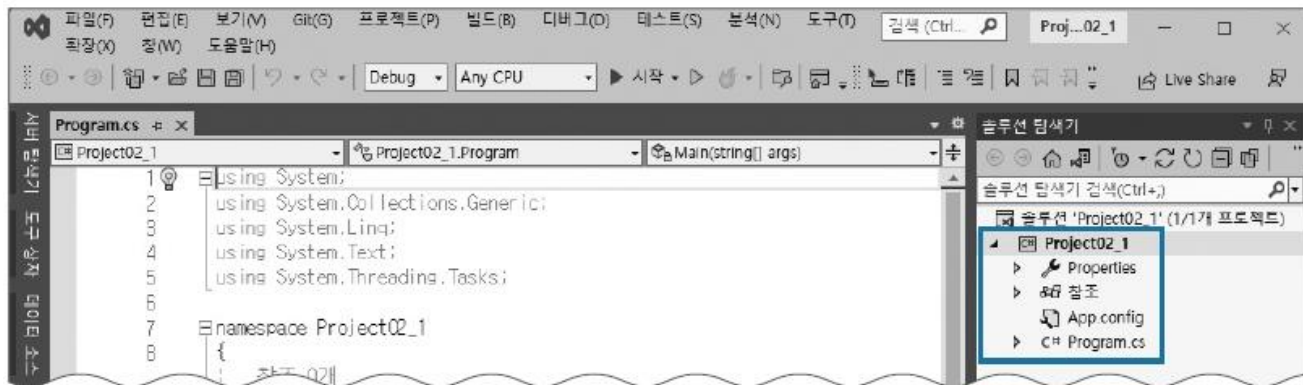


그림 2-6 생성된 프로젝트 확인

- ⑥ 실행 창 메뉴의 [파일] - [끝내기]를 선택해 Visual Studio를 종료한다.

02. 계산기 프로그램 기본 기능 구현

II. 기본적인 콘솔 앱 생성

SELF STUDY 2-1

새로운 프로젝트 2개를 빠른 속도로 만들어보자. 프로젝트 이름은 Test2, Test3으로 한다.

02. 계산기 프로그램 기본 기능 구현

II. 기본적인 콘솔 앱 생성

[실습 2-2] 덧셈, 뺄셈, 곱셈, 나눗셈 기능 구현하기

- ① Visual Studio를 실행한다
- ② 실행 창 오른쪽의 [프로젝트 또는 솔루션 열기]를 선택한 후, 이전에 작업했던 **C:\CookC#\W Project02_1** 폴더의 Project02_1.sln을 선택해 <열기>를 클릭한다.
- ③ 창 오른쪽에 위치한 [솔루션 탐색기]의 프로젝트 이름(Project02_1) 아래 [Program.cs]를 더블클릭한다.
- ④ 창 왼쪽에 위치한 코드 편집창에 우선 100과 50의 덧셈, 뺄셈, 곱셈, 나눗셈을 수행하는 프로그램을 코딩한다.
- ⑤ 틀린 글자가 없는지 확인한 후 실행 창 메뉴의 [파일] - [모두 저장]을 선택하거나 [모두 저장] 아이콘을 클릭하거나 Ctrl + S 키를 눌러 입력한 내용을 저장한다.

02. 계산기 프로그램 기본 기능 구현

II. 기본적인 콘솔 앱 생성

[소스 2-1] 프로그램 작성 (프로젝트명 : Project02_1)

```
1  static void Main(string[] args)
2  {
3      int a, b;
4      int result;
5
6      a = 100;
7      b = 50;
8
9      result = a + b;
10     Console.WriteLine(a + "+" + b + "=" + result);
11
12     result = a - b;
13     Console.WriteLine(a + "-" + b + "=" + result);
14
15     result = a * b;
16     Console.WriteLine(a + "*" + b + "=" + result);
17
18     result = a / b;
19     Console.WriteLine(a + "/" + b + "=" + result);
20 }
```

02. 계산기 프로그램 기본 기능 구현

III. 변수의 개념 이해

- 변수: 값을 저장하는 그릇(또는 방) 정도로 생각하면 된다.
 - 우선, 3행과 4행에서 변수를 3개 준비한다.



그릇 이름 : a



그릇 이름 : b



그릇 이름 : result

그림 2-7 그릇 준비

- 6, 7행에서 그릇 a에는 100을, 그릇 b에는 50을 넣는다.



그릇 이름 : a



그릇 이름 : b



그릇 이름 : result

그림 2-8 두 그릇에 값을 넣음

02. 계산기 프로그램 기본 기능 구현

III. 변수의 개념 이해

- 9행에서 그릇 a의 값과 그릇 b의 값을 더한 결과값을 그릇 result에 넣는다.



그림 2-9 덧셈 작업

02. 계산기 프로그램 기본 기능 구현

IV. Console.WriteLine() 메서드의 사용

- `Console.WriteLine()`은 괄호 안의 내용을 모니터에 출력하라는 의미의 메서드
- 코드 중간의 **+ 기호**는 내용을 이어서 출력하라는 뜻이다. **“+”**나 **“=”**는 문자열이다.

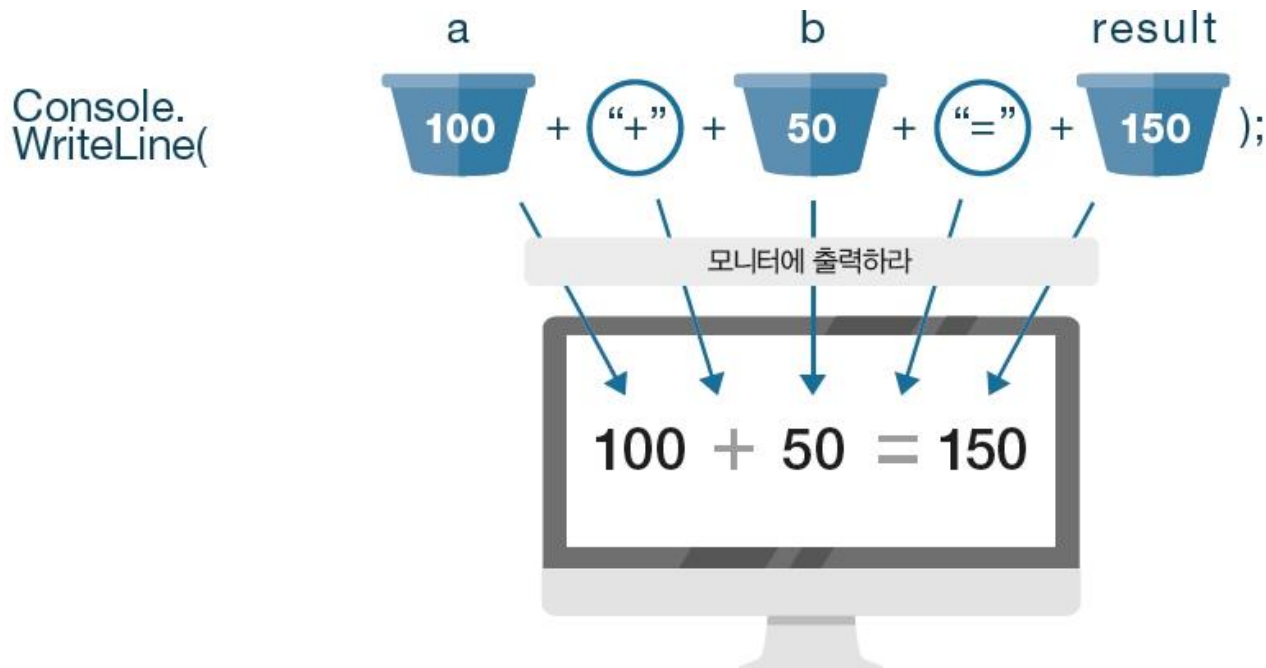


그림 2-10 Console.WriteLine() 메서드의 작동

02. 계산기 프로그램 기본 기능 구현

IV. Console.WriteLine() 메서드의 사용

여기서 잠깐

변수와 그릇의 차이

- 진짜 그릇이라면 [그림 2-9]에서 a와 b는 빈 그릇이 되겠지만, 변수는 result에 값이 들어가더라도 그대로 남아 있다

02. 계산기 프로그램 기본 기능 구현

IV. Console.WriteLine() 메서드의 사용

[실습 2-3] C# 프로젝트 빌드하기 - 컴파일과 링크

- ① 실행 창 메뉴의 [빌드] - [솔루션 빌드]를 선택해 프로젝트를 빌드한다.
- ② 특별히 문제가 없다면 Visual Studio 실행 창 왼쪽 하단에 **빌드: 성공 1, 실패 0, ...**
및 빌드에 성공했습니다 메시지가 출력될 것이다. 만약 실패 메시지가 뜨면 소스 코드에서 틀린 부분을 수정 한 후 다시 빌드하도록 한다.



그림 2-11 프로젝트의 빌드 성공 메시지

02. 계산기 프로그램 기본 기능 구현

IV. Console.WriteLine() 메서드의 사용

[실습 2-4] C# 프로젝트 실행하기

- ① Ctrl + F5 키를 누르거나, 실행 창 메뉴의 [디버그] - [디버그하지 않고 시작]을 선택해 실행한다.



```
C:\Windows\system32\cmd.exe
100+50=150
100-50=50
100*50=5000
100/50=2
계속하려면 아무 키나 누르십시오 . . .
```

그림 2-12 실행 결과

- ② 결과를 확인했으면 아무 키나 눌러 결과 창을 닫는다.

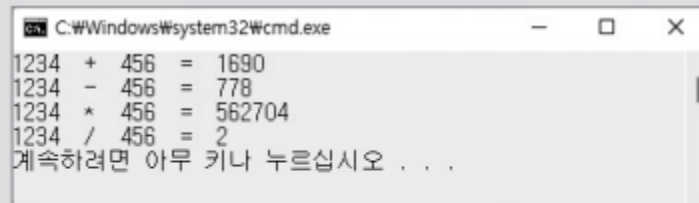
02. 계산기 프로그램 기본 기능 구현

IV. Console.WriteLine() 메서드의 사용

SELF STUDY 2-2

[소스 2-1]을 수정하여 1234와 456의 덧셈, 뺄셈, 곱셈, 나눗셈 결과를 확인해보자. 또, 글자가 다음과 같이 띄어쓰기되어 나오도록 해보자.

힌트 변수 a와 b에 대입하는 값을 변경하고, "+" 등의 앞뒤 공백(Space)을 출력에 추가한다.



```
C:\Windows\system32\cmd.exe
1234 + 456 = 1690
1234 - 456 = 778
1234 * 456 = 562704
1234 / 456 = 2
계속하려면 아무 키나 누르십시오 . . .
```

그림 2-13 실행 결과

03

계산기 프로그램 기능 확장

03. 계산기 프로그램 기능 확장

I. Console.ReadLine() 메서드의 서식

- 프로그램을 실행할 때 키보드로 숫자를 직접 입력하면 덧셈, 뺄셈, 곱셈, 나눗셈을 자유자재로 수행할 수 있다.

변수 a에 100을 넣는다 → 변수 a에 입력할 값을 키보드로 입력받는다
변수 b에 50을 넣는다 → 변수 b에 입력할 값을 키보드로 입력받는다



그림 2-14 변수에 키보드로 값을 직접 입력

03. 계산기 프로그램 기능 확장

I. Console.ReadLine() 메서드의 서식

- 키보드로 값을 입력받으려면 `Console.ReadLine()` 메서드를 사용하면 된다.

```
string 문자열_변수_이름;  
문자열_변수_이름 = Console.ReadLine( );
```

- 문제는 `Console.ReadLine()` 메서드는 무조건 문자열로만 입력을 받는다는 점이다. 그래서 `string` 형식의 변수를 별도로 준비해 입력받아야 한다. 계산을 위해서는 정수가 필요하므로, 문자열을 숫자로 변환하는 `Convert.ToInt32()` 메서드를 사용한다.

```
정수형_변수_이름 = Convert.ToInt32(문자열_변수);
```


03. 계산기 프로그램 기능 확장

II. Console.ReadLine() 메서드를 사용한 데이터 입력

[실습 2-5] 5 Console.ReadLine() 메서드 사용하기

- ① 새로운 프로젝트를 생성하기 위해 실행 창 메뉴의 [파일] - [새로 만들기] - [프로젝트]를 클릭 한다.
- ② [새 프로젝트 만들기] 창에서 언어를 [C#]으로 선택한다. 창 아래로 스크롤해 [콘솔 앱(.NET Framework)]을 선택한 후, 을 클릭한다.
- ③ [새 프로젝트 구성] 창에서 '프로젝트 이름'에 **Project02_2**라고 입력한 후, '위치'에서 을 <...>클릭해 **C:\WCookC#** 폴더를 선택한다. 그리고, <솔루션 및 프로젝트를 같은 디렉터리에 배치>에 체크한 후 <만들기>를 클릭한다.

03. 계산기 프로그램 기능 확장

II. Console.ReadLine() 메서드를 사용한 데이터 입력

[실습 2-5] 5 Console.ReadLine() 메서드 사용하기

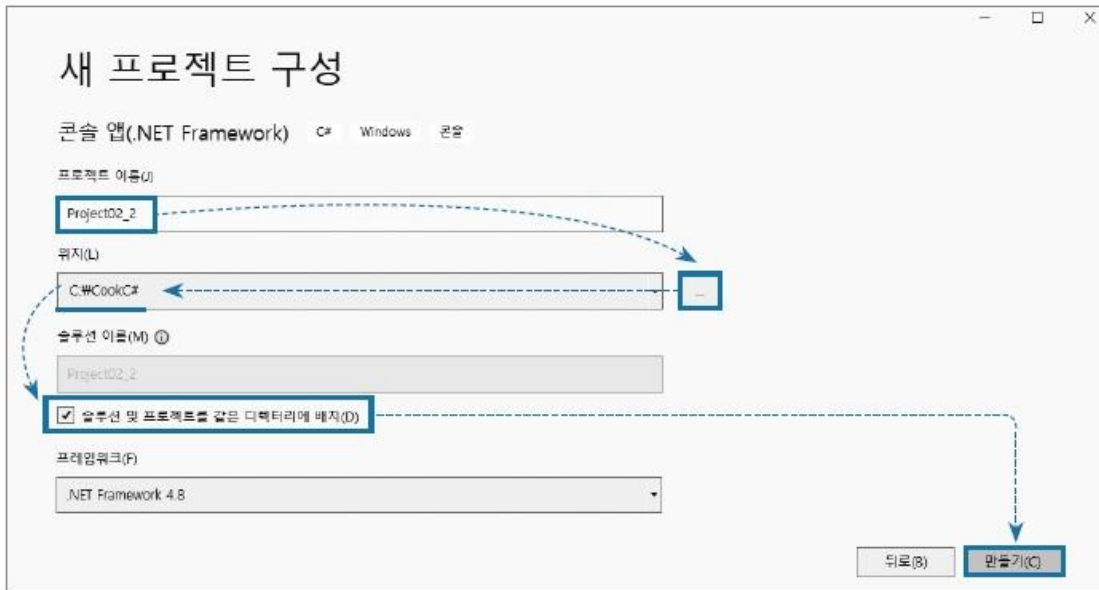


그림 2-15 Project02_2 새 프로젝트 생성

- ④ 자동 완성된 Program.cs 파일에 다음과 같이 소스 코드를 추가한다.

03. 계산기 프로그램 기능 확장

II. Console.ReadLine() 메서드를 사용한 데이터 입력

[소스 2-2] 키보드로 값을 직접 입력받기 (프로젝트명 : Project02_2)

```
1  static void Main(string[] args)
2  {
3      int a, b;
4      int result;
5
6      string str;
7
8      str = Console.ReadLine();
9      a = Convert.ToInt32(str);
10
11     str = Console.ReadLine();
12     b = Convert.ToInt32(str);
13
14     result = a + b;
15     Console.WriteLine(a + "+" + b + "=" + result);
16
17     // 이다음은 [소스 2-1]의 12~19행과 동일함
18     ...
25 }
```

03. 계산기 프로그램 기능 확장

II. Console.ReadLine() 메서드를 사용한 데이터 입력

[실습 2-5] 5 Console.ReadLine() 메서드 사용하기

- 6행은 문자열을 입력 받기 위해 문자열 변수 str을 선언한다.
 - 8행은 키보드로 문자열을 입력 받아 str에 대입한다.
 - 9행은 **Convert.ToInt32()** 메서드를 이용해서 문자열인 str을 정수로 변환해 a에 대입한다.
 - 11, 12행은 키보드로 문자열을 입력받아 str에 대입한 후, 정수로 변환해 b에 대입
- ⑤ Ctrl + F5 키를 눌러서 빌드 및 실행을 동시에 수행한다. 명령 프롬프트 창을 보면, 커서만 깜박거리는 것을 확인할 수 있다.



그림 2-16 실행 결과 1

03. 계산기 프로그램 기능 확장

II. Console.ReadLine() 메서드를 사용한 데이터 입력

[실습 2-5] 5 Console.ReadLine() 메서드 사용하기

- ⑥ 일단 숫자를 하나 입력하고 Enter 를 누른다. 다시 숫자 하나를 입력하고 Enter 를 누른다.



그림 2-17 실행 결과 2

- ⑦ 좀 더 사용하기 편하도록 다음과 같이 소스 코드를 수정해보자.

03. 계산기 프로그램 기능 확장

II. Console.ReadLine() 메서드를 사용한 데이터 입력

[소스 2-3] 도움말 출력하기 (프로젝트명 : Project02_2)

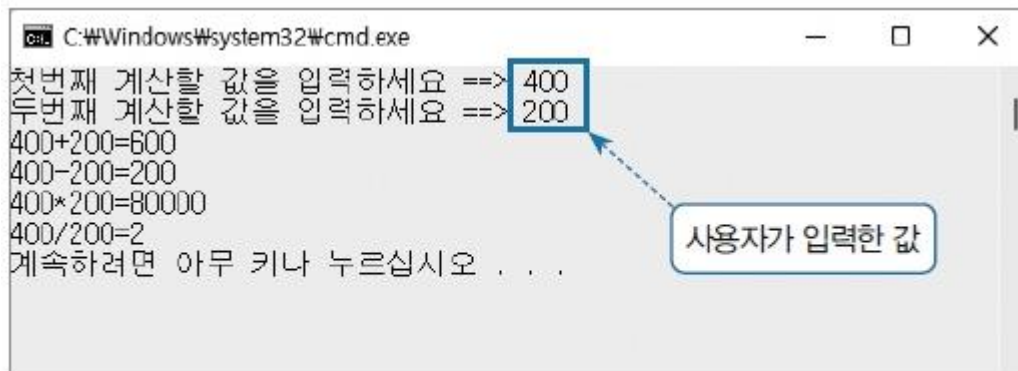
```
1  static void Main(string[] args)
2  {
3      int a, b;
4      int result;
5
6      string str;
7
8      Console.Write("첫번째 계산할 값을 입력하세요 => ");
9      str = Console.ReadLine();
10     a = Convert.ToInt32(str);
11
12     Console.Write("두번째 계산할 값을 입력하세요 => ");
13     str = Console.ReadLine();
14     b = Convert.ToInt32(str);
15
16     result = a + b;
17     // 이다음은 [소스 2-2]의 15행 이하와 동일함
...
```

03. 계산기 프로그램 기능 확장

II. Console.ReadLine() 메서드를 사용한 데이터 입력

[실습 2-5] Console.ReadLine() 메서드 사용하기

- ⑧ 다시 Ctrl + F5 키를 눌러서 빌드 및 실행을 동시에 수행한다. 이제는 어떠한 값이든 입력하기만 하면 즉각 결과 값이 나올 것이다.



```
C:\Windows\system32\cmd.exe
첫번째 계산할 값을 입력하세요 ==> 400
두번째 계산할 값을 입력하세요 ==> 200
400+200=600
400-200=200
400*200=80000
400/200=2
계속하려면 아무 키나 누르십시오 . . .
```

그림 2-18 실행 결과 3

03. 계산기 프로그램 기능 확장

II. Console.ReadLine() 메서드를 사용한 데이터 입력

SELF STUDY 2-3

[소스 2-3]의 a b 값으로 200, 400을 입력하면 나눗셈은 결과가 0이 나온다. 결과가 0.5가 나오도록 소스 코드를 수정해보자.

힌트 소수점이 있는 실수를 처리하기 위해서는 변수를 double형으로 선언해야 한다.

04

그림판 프로그램 작성

04. 그림판 프로그램 작성

I. 기본적인 원폼 앱 생성

- 원폼(WinForm, Windows Form의 약자): 윈도우 창이 나오는 프로그램
- 대부분의 프로그램(웹브라우저, 포토샵, 그림판, 엑셀, 한글 등)은 윈도우 창이 나오는 GUI(Graphical User Interface) 환경으로 작동한다.



그림 2-19 GUI 프로그램의 예 - 그림판 프로그램

04. 그림판 프로그램 작성

I. 기본적인 원폼 앱 생성

[실습 2-6] 간단 그림판 프로젝트 생성하기

- ① 실행 창 메뉴의 [파일] - [새로 만들기] - [프로젝트]를 클릭한다. 만약 Visual Studio를 새로 실행했다면 창 오른쪽의 [새 프로젝트 만들기]를 클릭한다.
- ② [새 프로젝트 만들기] 창에서 언어를 [C#]으로 선택한다. 창 아래로 스크롤해 이번에는 [Windows Forms 앱(.NET Framework)]을 선택한 후 <다음>을 클릭한다.

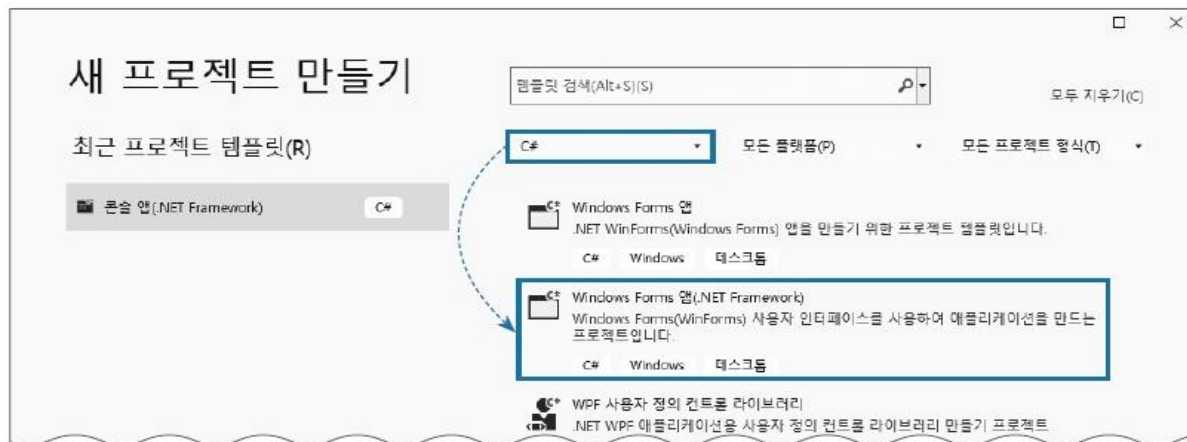


그림 2-20 Windows Forms 앱(.NET Framework) 선택

04. 그림판 프로그램 작성

I. 기본적인 원폼 앱 생성

[실습 2-6] 간단 그림판 프로젝트 생성하기

- ③ [새 프로젝트 구성] 창에서 '프로젝트 이름'에 **Project02_3**이라고 입력한 후, '위치'에서 <...>을 클릭해 **C:\W\CookC#** 폴더를 선택한다. 그리고, <솔루션 및 프로젝트를 같은 디렉터리에 배치>에 체크한 후 <만들기>를 클릭한다.
- ④ 창이 비어 있는 원폼이 준비된 상태로 프로젝트가 완성된다

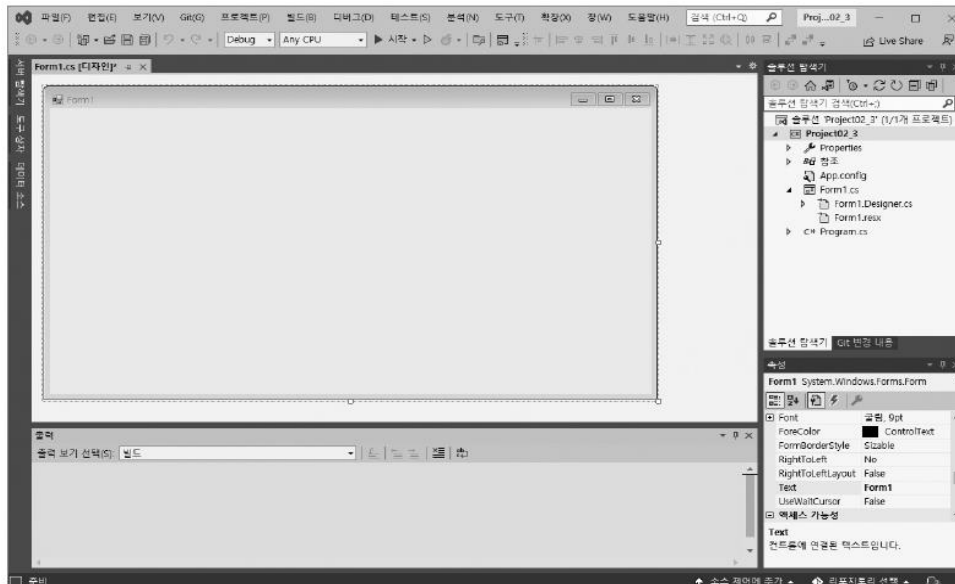


그림 2-21 완성된 원폼 프로젝트

04. 그림판 프로그램 작성

I. 기본적인 원폼 앱 생성

[실습 2-6] 간단 그림판 프로젝트 생성하기

- ⑤ 일단 Ctrl + F5 키를 눌러서 빌드 및 실행을 동시에 진행한다. 빈 윈도우창이 실행되는데, 아무것도 없지만 프로그램을 작성한 것이다. 확인했으면 오른쪽 상단의 <X>를 눌러서 실행 창을 닫는다

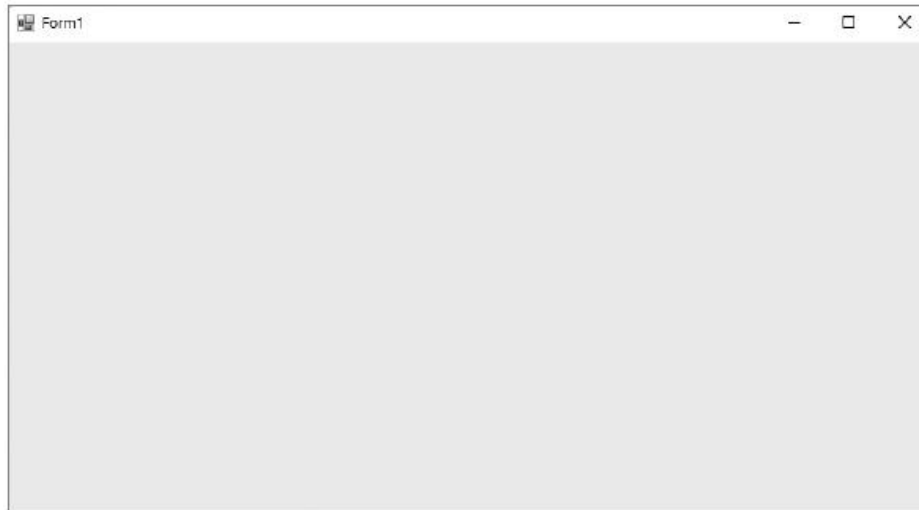


그림 2-22 실행된 원폼 프로그램

04. 그림판 프로그램 작성

I. 기본적인 원폼 앱 생성

[실습 2-6] 간단 그림판 프로젝트 생성하기

- ⑥ 화면 왼쪽의 [도구 상자]를 클릭하고 [공용 컨트롤]을 확장하면, 컨트롤이 여러 개 보인다. 이 중 PictureBox를 드래그해 원폼에 끌어다 놓자. [도구 상자]를 계속 화면에 나타나게 하고 싶다면 [도구 상자] 오른쪽 상단의 압정 모양 버튼을 눌러 놓자.

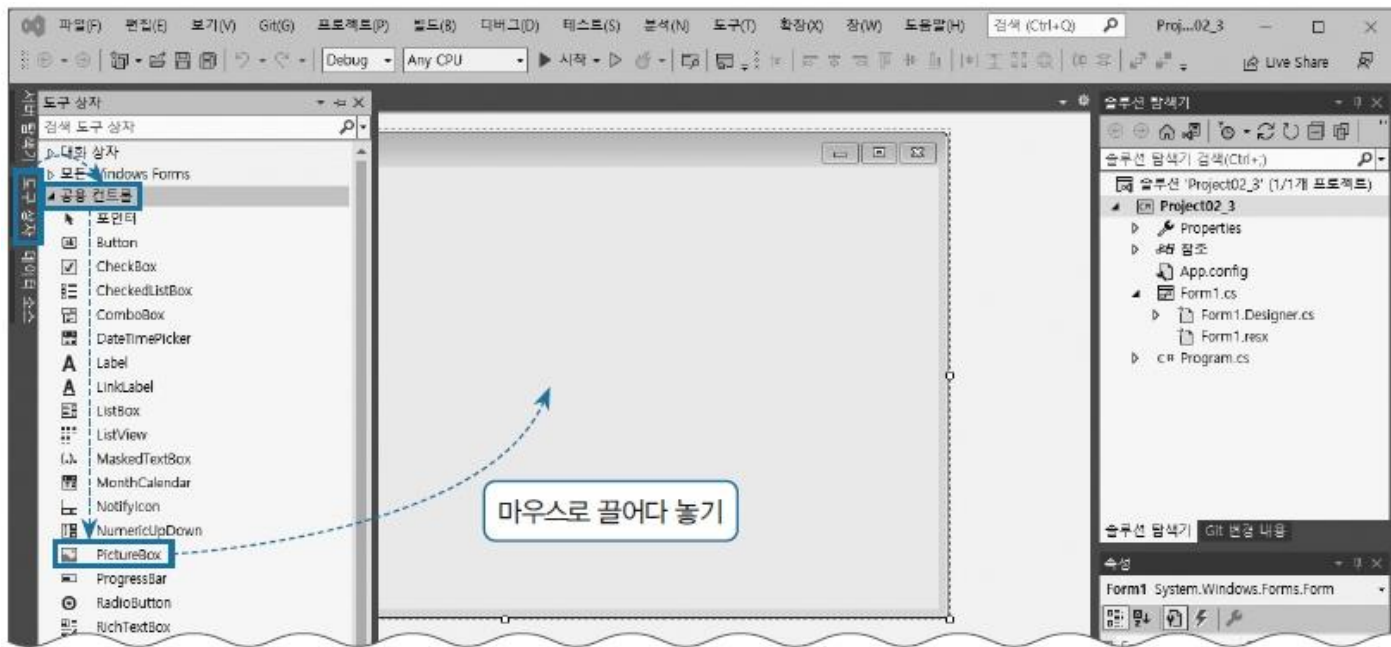


그림 2-23 PictureBox 추가

04. 그림판 프로그램 작성

I. 기본적인 원폼 앱 생성

[실습 2-6] 간단 그림판 프로그램 생성하기

- ⑦ PictureBox를 선택하고 박스의 오른쪽 위 화살표(▶)를 클릭한다. <이미지 선택>을 클릭한 뒤 [리소스 선택] 창이 뜨면 <로컬 리소스>를 선택하고 <가져오기>를 클릭해 적당한 그림 파일을 고른다. 미리보기 후 <확인>을 클릭해서 창을 닫는다

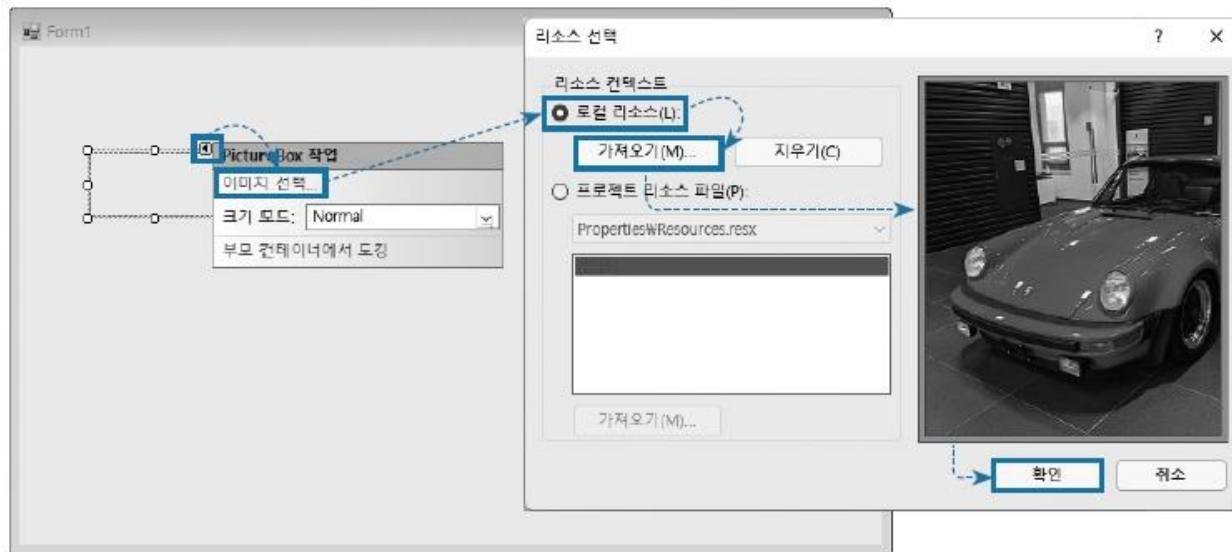


그림 2-24 그림 선택

04. 그림판 프로그램 작성

I. 기본적인 원폼 앱 생성

[실습 2-6] 간단 그림판 프로젝트 생성하기

- ⑧ 그림이 나타나면 그림 및 폼의 크기를 마우스로 적절히 조절하면 된다

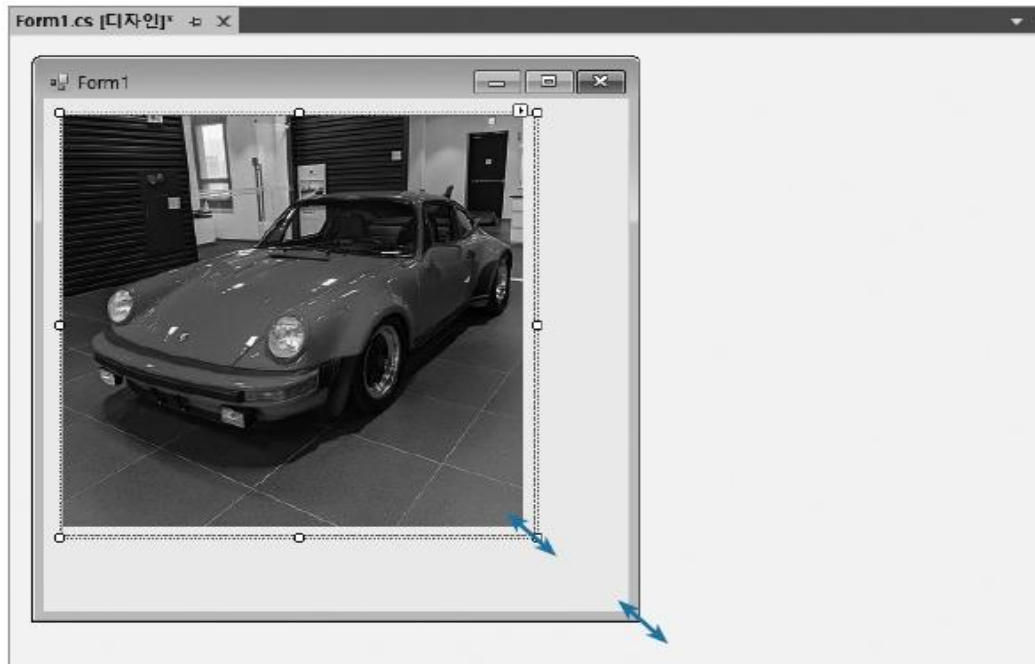


그림 2-25 그림 및 폼 크기 조절

04. 그림판 프로그램 작성

I. 기본적인 원폼 앱 생성

[실습 2-6] 간단 그림판 프로그램 생성하기

⑨ 이번에는 적당한 위치에 Button을 2개 끌어서 놓아보자.

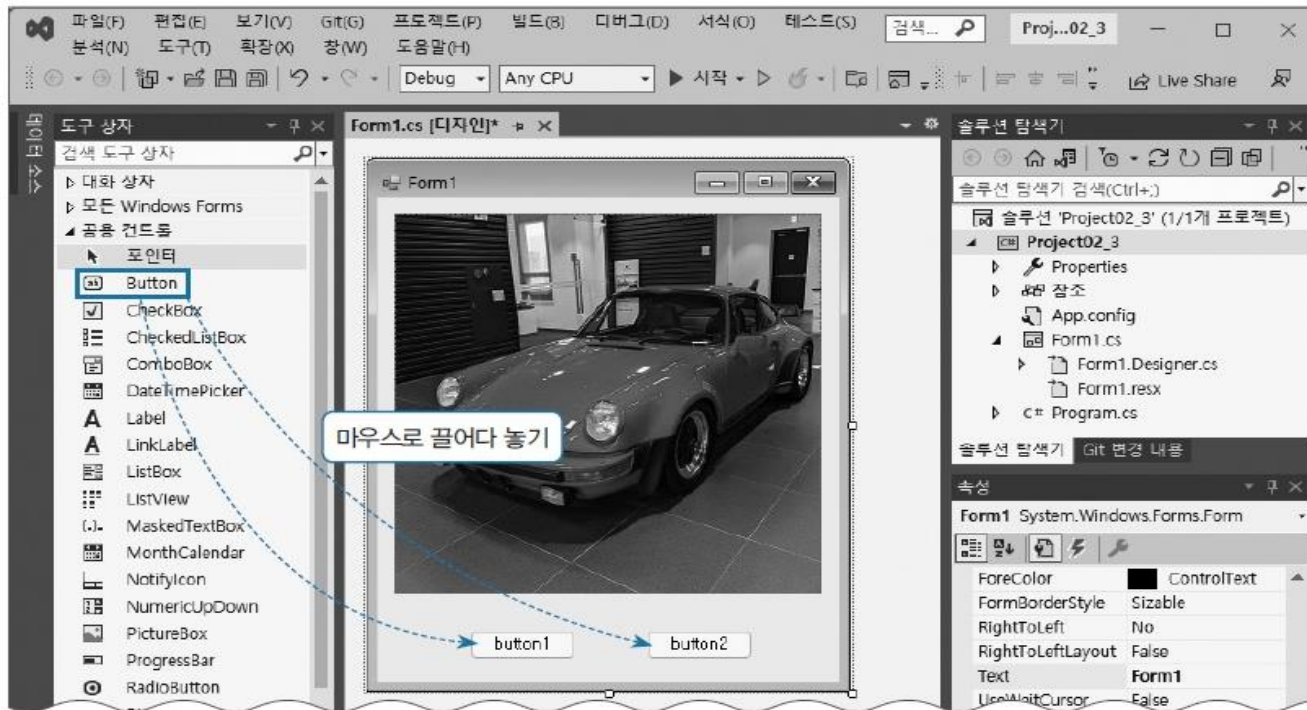


그림 2-26 Button 추가

04. 그림판 프로그램 작성

I. 기본적인 원폼 앱 생성

[실습 2-6] 간단 그림판 프로젝트 생성하기

- ⑩ 'button1'이라고 쓰여 있는 첫 번째 Button을 클릭해 선택하고, 화면 오른쪽 하단 [속성] 창 - [모양] - [Text] 부분을 **보이기**로 고치고 Enter 를 누른다

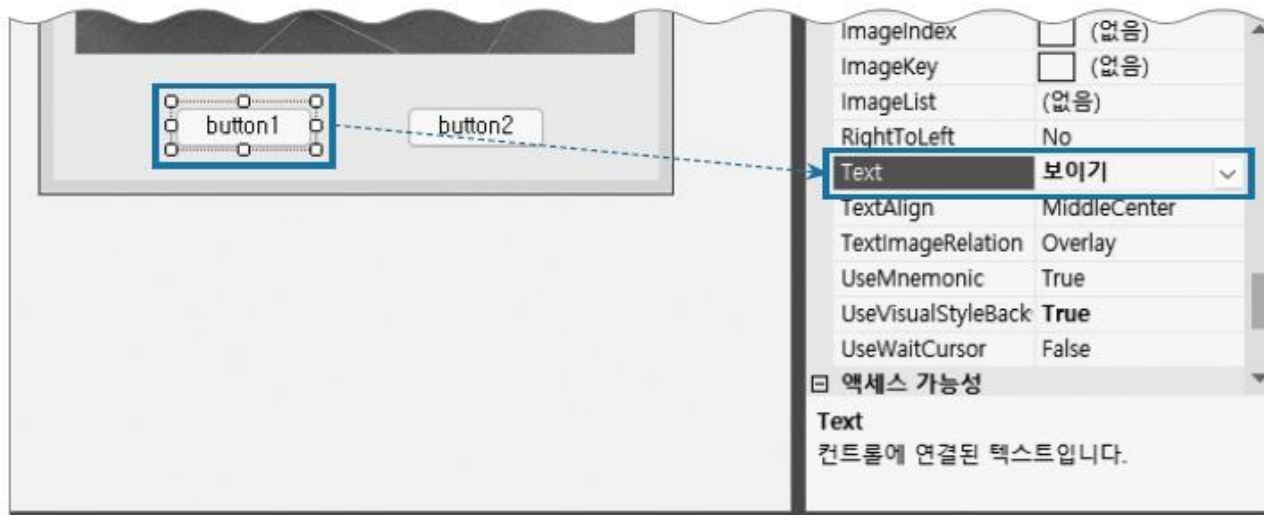


그림 2-27 Button의 텍스트 변경

04. 그림판 프로그램 작성

I. 기본적인 원폼 앱 생성

[실습 2-6] 간단 그림판 프로젝트 생성하기

- ⑪ 같은 방식으로 'button2'라고 쓰여 있는 두 번째 Button을 클릭해 선택하고, 화면 오른쪽 하단 [속성] 창 - [모양] - [Text] 부분을 **감추기**로 고치고 Enter 를 누른다. 최종 폼 화면은 [그림 2-28]과 같이 된다.



그림 2-28 최종 화면

04. 그림판 프로그램 작성

I. 기본적인 원폼 앱 생성

[실습 2-6] 간단 그림판 프로젝트 생성하기

- ⑫ 화면의 버튼을 더블클릭해 소스 코드가 뜨면, `button1_Click()` 메서드 안에 다음 한 줄을 채우도록 한다.

[소스 2-4] PictureBox 보이기 (프로젝트명 : Project02_3)

```
1 namespace Project02_3
2 {
3     public partial class Form1 : Form
4     {
5         public Form1()
6         {
7             InitializeComponent();
8         }
9     }
```

04. 그림판 프로그램 작성

I. 기본적인 원폼 앱 생성

[실습 2-6] 간단 그림판 프로젝트 생성하기

[소스 2-4] PictureBox 보이기 (프로젝트명 : Project02_3)

```
10     private void button1_Click(object sender, EventArgs e)
11     {
12         pictureBox1.Visible = true;
13     }
14 }
15 }
```

- 10~13행은 Button을 클릭하면 실행되는 메서드. 메서드의 이름은 컨트롤이름 _Click으로 구성되는데, 첫 번째 Button의 이름은 자동으로 button1으로 지정된다.
- 12행에서는 PictureBox의 속성 중 화면에 보일지의 여부를 결정하는 Visible 속성을 true 로 지정했다. 즉, PictureBox가 화면에 보이도록 설정한다. PictureBox 이름도 자동으로 pictureBox1으로 지정된다.

04. 그림판 프로그램 작성

I. 기본적인 원폼 앱 생성

[실습 2-6] 간단 그림판 프로젝트 생성하기

- ⑬ 화면 위쪽 탭 중 [Form1.cs [디자인]] 탭을 클릭하고, 버튼을 더블클릭한다.

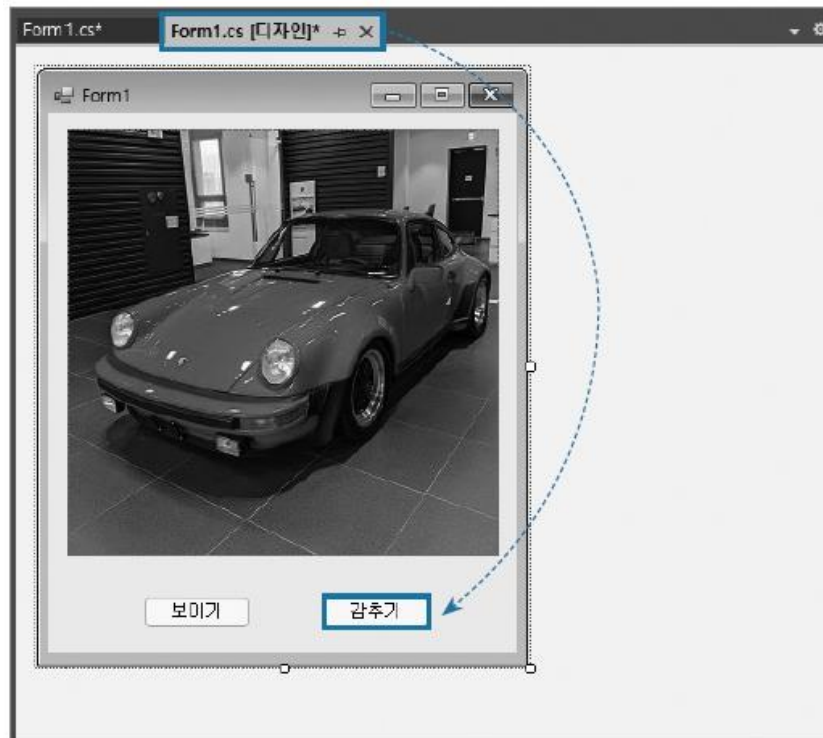


그림 2-29 디자인 화면으로 돌아와 감추기 버튼 클릭

04. 그림판 프로그램 작성

I. 기본적인 원폼 앱 생성

[실습 2-6] 간단 그림판 프로젝트 생성하기

⑭ 소스 코드가 뜨면, button2_Click() 메서드 안에 다음 소스 코드 한 줄을 채운다.

[소스 2-5] PictureBox 감추기 (프로젝트명 : Project02_3)

```
1 private void button2_Click(object sender, EventArgs e)
2 {
3     pictureBox1.Visible = false;
4 }
```

04. 그림판 프로그램 작성

I. 기본적인 원폼 앱 생성

[실습 2-6] 간단 그림판 프로젝트 생성하기

- ⑮ Ctrl + F5 키를 눌러서 빌드 및 실행을 동시에 실행한다. 이후 [Form1] 창이 뜨는데, <보이기> 및 <감추기> 버튼을 번갈아 눌러보면 그림이 화면에서 보였다, 사라졌다 할 것이다. 확인했으면 화면 오른쪽 상단의 <X>를 눌러서 실행 창을 닫는다.



04. 그림판 프로그램 작성

II. 구현할 기능 계획

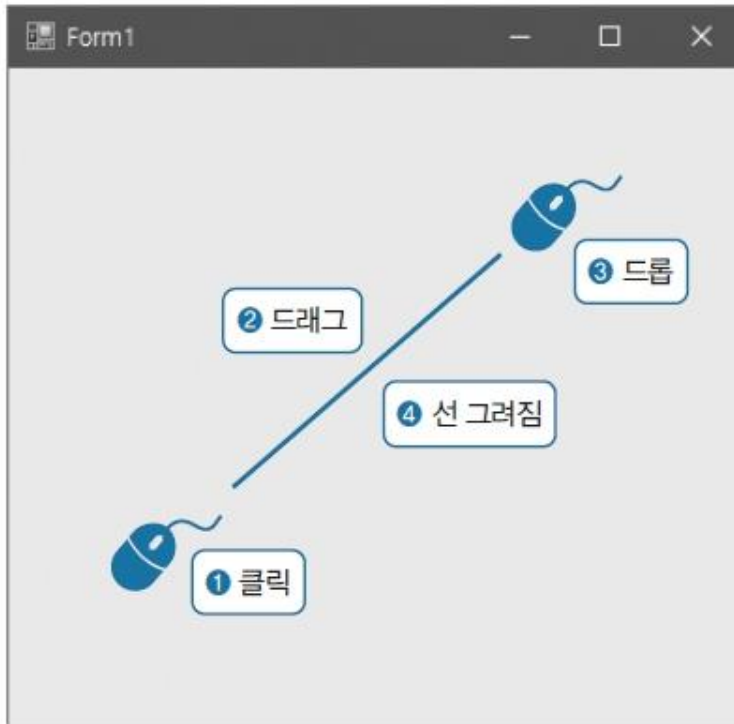


그림 2-31 마우스를 사용해 선 그리기

- 화면에서 마우스 왼쪽 버튼을 클릭 (click)하고, 누른 상태에서 마우스를 드래그(drag)한다. 마우스 버튼은 놓는 드롭(drop) 동작을 이어서 수행하면 두 점 사이에 선이 그려질 것이다.

04. 그림판 프로그램 작성

II. 구현할 기능 계획

- 우선 선의 시작 x , 시작 y , 끝 x , 끝 y 에 대한 변수를 전역 변수로 선언한다.

```
int 시작_x, 시작_y, 끝_x, 끝_y;
```

- 마우스를 클릭하는 순간에는 아무것도 그려지지 않고 시작점의 좌표만 저장되면 된

```
// 마우스를 클릭하는 순간  
시작_x = 현재_X_좌표;  
시작_y = 현재_Y_좌표;
```

- 마우스를 떼는 순간에는 끝점인 끝 x , 끝 y 좌표를 구한 후 선을 그린다.
- 선이 그려지는 C# 소스 코드는 다음 코드의 3~5행과 같이 거의 고정되어 사용된다.

04. 그림판 프로그램 작성

II. 구현할 기능 계획

```
1 // 마우스를 떼는 순간
2 끝_x = 현재_X_좌표;
3 끝_y = 현재_Y_좌표;
4 Graphics g = CreateGraphics();
5 Pen pen = new Pen(색상, 두께);
6 g.DrawLine(pen, 시작_x, 시작_y, 끝_x, 끝_y);
```

- 4행의 Graphics는 선을 그리기 위해 항상 준비해 두는 코드이다.
- 5행은 선을 그리기 위해 펜을 준비하는 코드이다. 색상과 두께는 필요에 따라 변경할 수 있다.
- 6행이 실제 선이 그려지는 코드인데 DrawLine() 메서드는 pen, 시작 x, 시작 y, 끝 x, 끝 y 등 5개의 변수를 사용한다. pen에 지정한 색상과 두께로 선이 그려진다

04. 그림판 프로그램 작성

III. 간단 그림판 완성

[실습 2-7] 간단 그림판 완성하기

- ① 새로운 프로젝트를 생성하기 위해 실행 창 메뉴의 [파일] - [새로 만들기] - [프로젝트]를 클릭 한다. 만약, Visual Studio를 새로 실행했다면 창 오른쪽의 [새 프로젝트 만들기]를 클릭한다.
- ② [새 프로젝트 만들기] 창에서 언어를 [C#]으로 선택한다. 창 아래로 조금 스크롤해 [Windows Forms 앱(.NET Framework)]을 선택한 후 <다음>을 클릭한다.
- ③ [새 프로젝트 구성] 창에서 '프로젝트 이름'에 **Project02_4**라고 입력한 후, '위치'에서 <...>을 클릭해 **C:\W\CookC#** 폴더를 선택한다. 그리고, <솔루션 및 프로젝트를 같은 디렉터리에 배치>에 체크한 후 <만들기>를 클릭한다.

04. 그림판 프로그램 작성

III. 간단 그림판 완성

[실습 2-7] 간단 그림판 완성하기

- ④ 비어 있는 원폼이 준비된 상태로 프로젝트가 완성된다. 폼 크기를 조절해 정사각형으로 만들고, 창 왼쪽 상단에 Button을 하나 추가한 뒤 텍스트를 '화면 초기화'라고 쓴다.

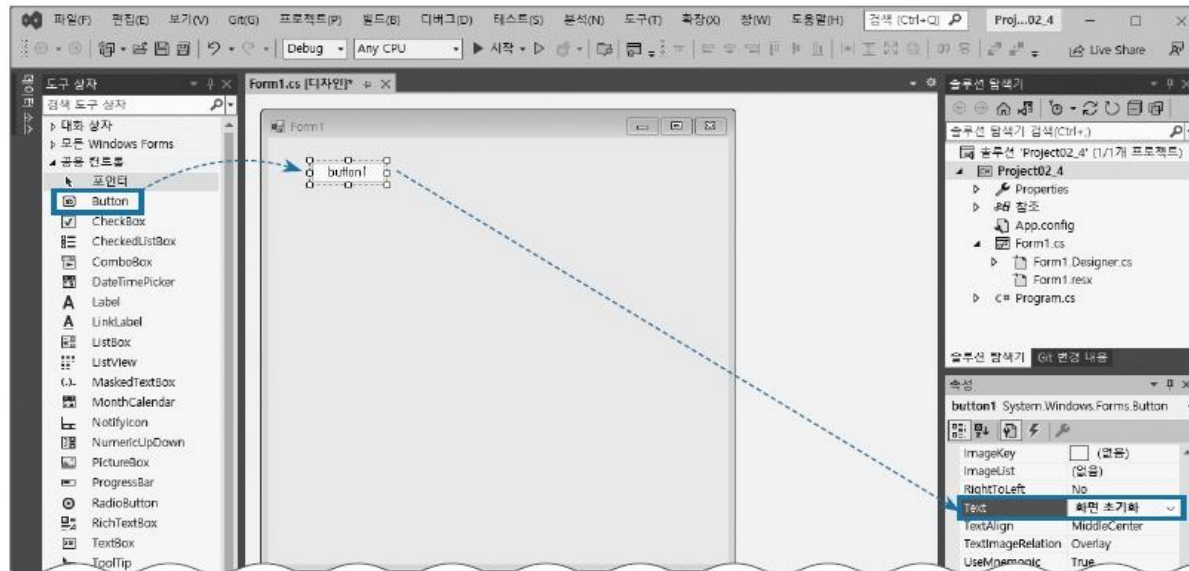


그림 2-32 완성된 폼에 Button 추가

04. 그림판 프로그램 작성

III. 간단 그림판 완성

[실습 2-7] 간단 그림판 완성하기

- ⑤ 폼의 빈 곳을 클릭하고 텍스트를 '간단 그림판(Ver 0.1)'이라고 적은 뒤 Enter 를 누르면 폼의 제목이 바뀐다.

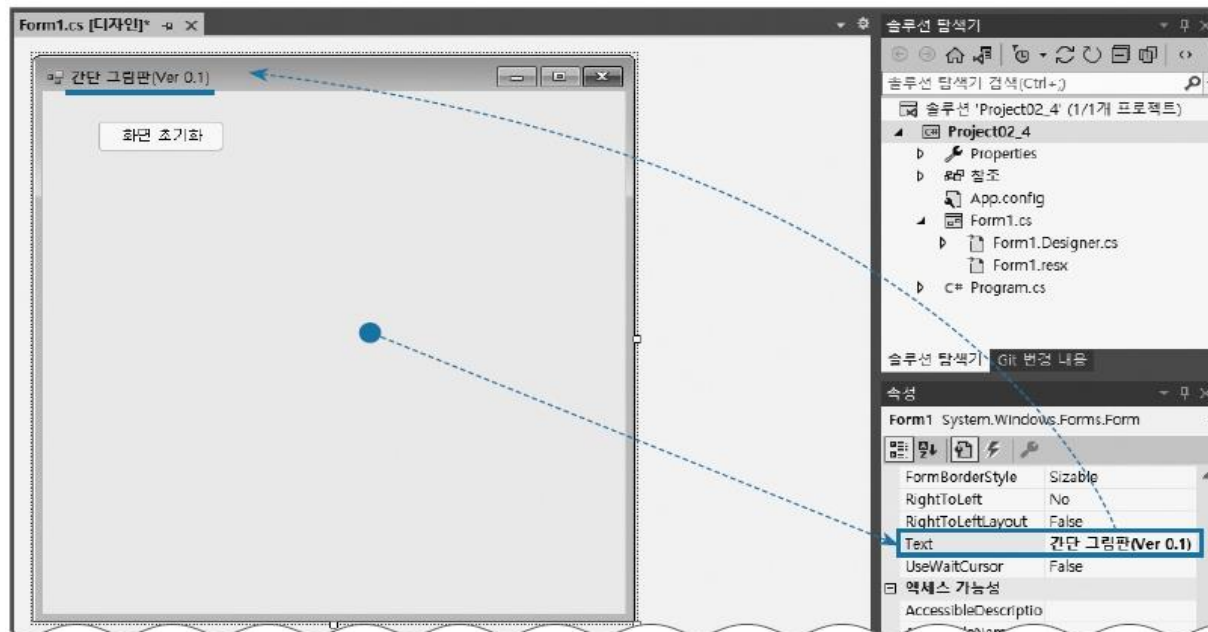


그림 2-33 폼 제목 변경

04. 그림판 프로그램 작성

III. 간단 그림판 완성

[실습 2-7] 간단 그림판 완성하기

- ⑥ 마우스 클릭 이벤트가 발생했을 때 처리하는 이벤트 메서드를 추가해보자. 폼의 빈 곳을 클릭하고 오른쪽 [속성] 창에서 번개 모양의 이벤트 아이콘을 클릭한다. 그리고 [마우스] 창에서 [MouseDown] 오른쪽 빈 곳을 더블클릭한다.

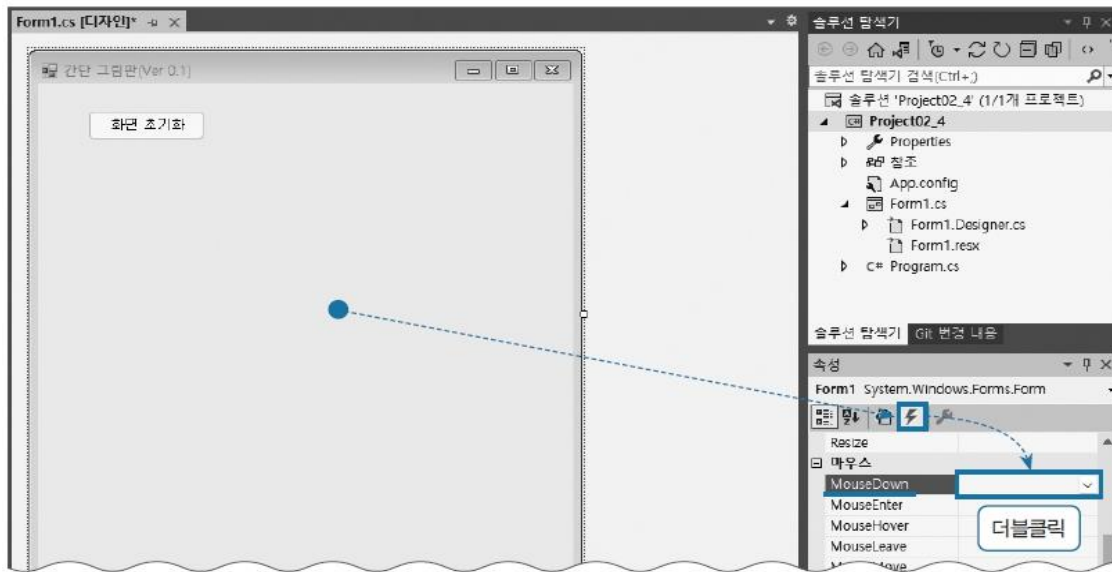


그림 2-34 MouseDown 이벤트 추가

04. 그림판 프로그램 작성

III. 간단 그림판 완성

[실습 2-7] 간단 그림판 완성하기

- ⑦ 소스 코드 창이 뜨면서 자동으로 `Form1_MouseDown()` 메서드가 완성된다. 시작 x , 시작 y , 끝 x , 끝 y 를 저장할 전역 변수를 추가하면 마우스를 클릭(= 마우스다운)하여 시작 x , 시작 y 에 좌표를 대입할 수 있다.

[소스 2-6] 마우스다운 이벤트 처리하기 (프로젝트명 : Project02_4)

```
1 namespace Project02_4
2 {
3     public partial class Form1 : Form
4     {
5         public Form1()
6         {
7             InitializeComponent();
8         }
9     }
```


04. 그림판 프로그램 작성

III. 간단 그림판 완성

[실습 2-7] 간단 그림판 완성하기

```
10     int x1, y1, x2, y2;
11     private void Form1_MouseDown(object sender, MouseEventArgs e)
12     {
13         x1 = e.X;
14         y1 = e.Y;
15     }
16 }
17 }
```

- 10행은 선의 시작 x , 시작 y , 끝 x , 끝 y 에 대한 변수를 전역 변수로 선언한다.
- 11~15행은 마우스를 클릭(=다운)하면 실행되는 메서드다.
- 중요한 건 11행의 파라미터 중 **MouseEventArgs** 형식의 e 변수이다. e 변수에는 마우스를 눌렀을 때의 좌표 등 다양한 정보가 들어 있다.
- 13, 14행의 $e.X$ 와 $e.Y$ 에는 마우스를 클릭한 위치에 대한 x , y 좌표 값이 들어 있다.

04. 그림판 프로그램 작성

III. 간단 그림판 완성

[실습 2-7] 간단 그림판 완성하기

- ⑧ 마우스를 떼었을 때(= 마우스업) 작동하는 동작을 만들어보자. 창 위쪽의 [Form1.cs [디자인]] 탭을 클릭하고, 폼의 빈 곳을 클릭해 오른쪽 [속성] 창에서 번개 모양의 이벤트 아이콘을 클릭한다. 그리고 [마우스] 창에서 [MouseUp] 오른쪽 빈 곳을 더블클릭한다.

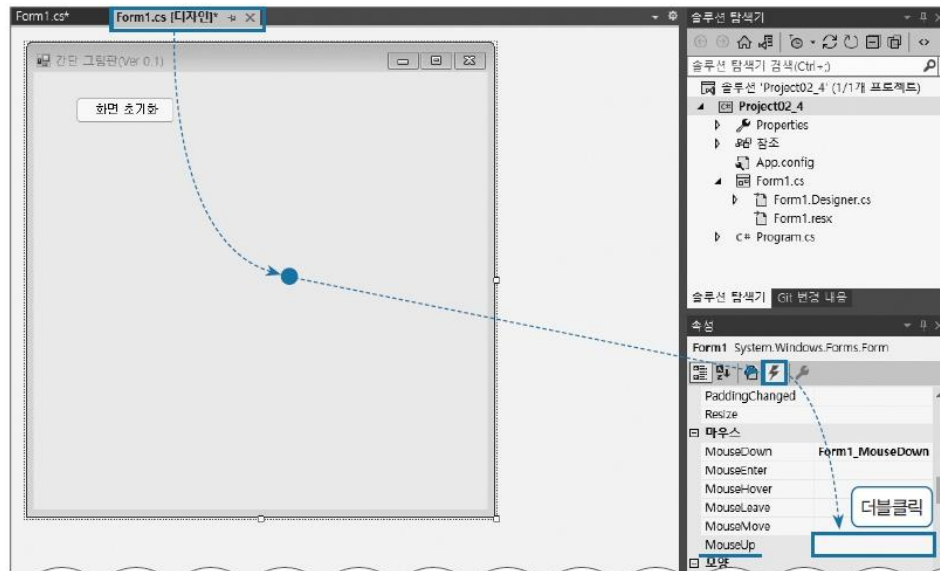


그림 2-35 MouseUp 이벤트 추가

04. 그림판 프로그램 작성

III. 간단 그림판 완성

[실습 2-7] 간단 그림판 완성하기

- ⑨ 소스 코드 창이 뜨면 Form1_MouseUp() 메서드를 작성하자. 끝점을 저장하고, 시작점부터 끝점까지 그리는 코드를 추가하는 것이다.

[소스 2-7] 마우스업 이벤트 처리하기 (프로젝트명 : Project02_4)

```
1 private void Form1_MouseUp(object sender, MouseEventArgs e)
2 {
3     x2 = e.X;
4     y2 = e.Y;
5
6     Graphics g = CreateGraphics();
7     Pen pen = new Pen(Color.Blue, 10);
8     g.DrawLine(pen, x1, y1, x2, y2);
9 }
```

04. 그림판 프로그램 작성

III. 간단 그림판 완성

[실습 2-7] 간단 그림판 완성하기

- ⑩ 화면을 초기화하는 기능을 추가해보자. 창 위쪽의 [Form1.cs [디자인]] 탭을 클릭하고, <화면 초기화> 버튼을 선택한다. 그리고 화면 오른쪽 [속성] 창 - [속성] 아이콘 - [모양] 부분의 **BackColor**를 클릭한다. **BackColor** 오른쪽 칸 화살표(▼)를 클릭하면 [사용자 지정] 탭이 뜨는데 그 안에서 색상을 다양하게 선택할 수 있다.

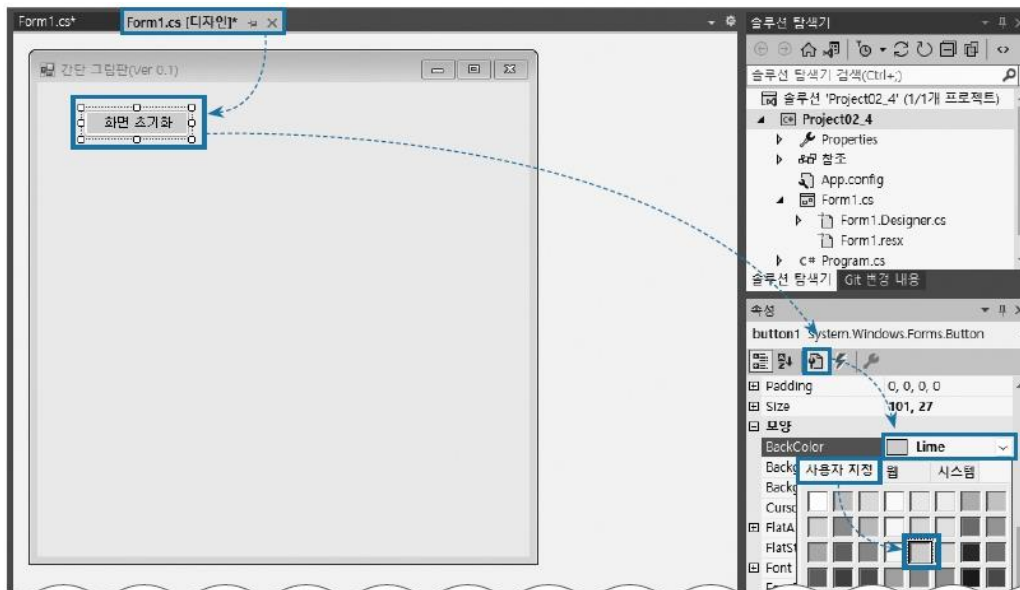


그림 2-36 버튼의 배경색 변경

04. 그림판 프로그램 작성

III. 간단 그림판 완성

[실습 2-7] 간단 그림판 완성하기

- ⑪ 이번에는 버튼을 더블클릭한 후, 화면을 초기화하기 위한 코드를 추가한다.

[소스 2-8] 화면 초기화 소스 코드 (프로젝트명 : Project02_4)

```
1 private void button1_Click(object sender, EventArgs e)
2 {
3     this.Refresh();
4 }
```

- 3행은 화면을 깨끗하게 지우는 코드이다. **this**는 현재 원폼 화면을 의미한다.

04. 그림판 프로그램 작성

III. 간단 그림판 완성

[실습 2-7] 간단 그림판 완성하기

- ⑫ 코딩을 마쳤으므로 Ctrl+F5 키를 눌러서 빌드 및 실행을 동시에 진행한다. 빌드에 성공하면 실행 화면이 뜬다. 마우스를 이용해 적당한 글자를 써보자.



그림 2-37 프로그램 실행 화면

04. 그림판 프로그램 작성

III. 간단 그림판 완성

SELF STUDY 2-4

[그림 2-37]의 실행 화면에서 글자의 색상이 빨간색으로 나오고, 선의 두께는 5로 그려지도록 소스 코드를 수정해보자.

Thank You !