

제7장 마술사를 이용한 코딩 연습

변영철

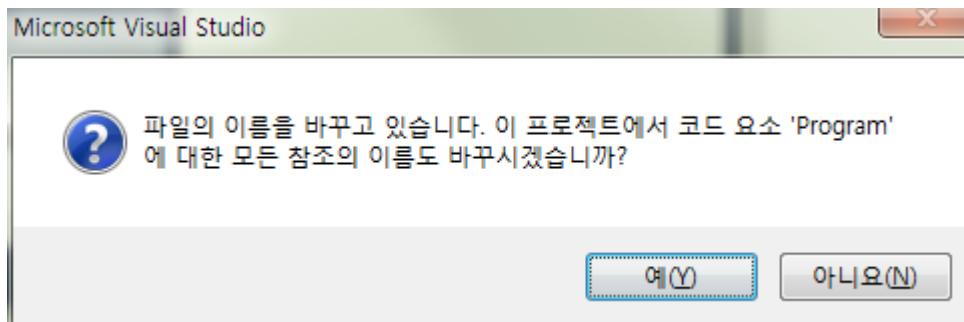
코딩 연습

- 간편하게 마술사로 코드를 생성
- 마술사가 만든 코드를 지워서 앞서 우리가 작성했던 코드로 단순화 해보기

01. 프로젝트 작성 및 사전작업

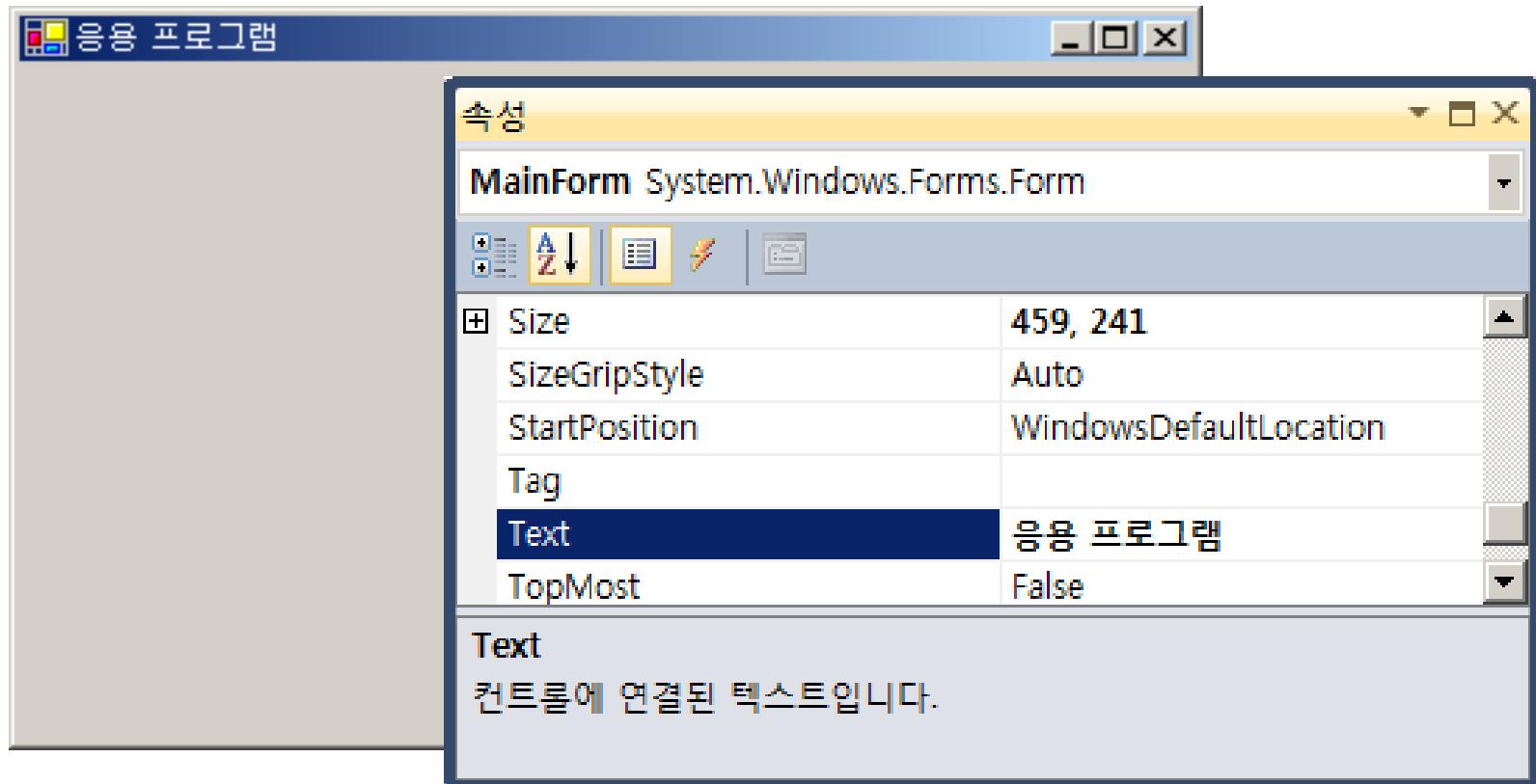
- My3 프로젝트 생성
 - Visual C# | Windows Forms 응용 프로그램
- 파일이름 바꾸기
 - Program.cs → My3.cs
 - Form1.cs → MainForm.cs

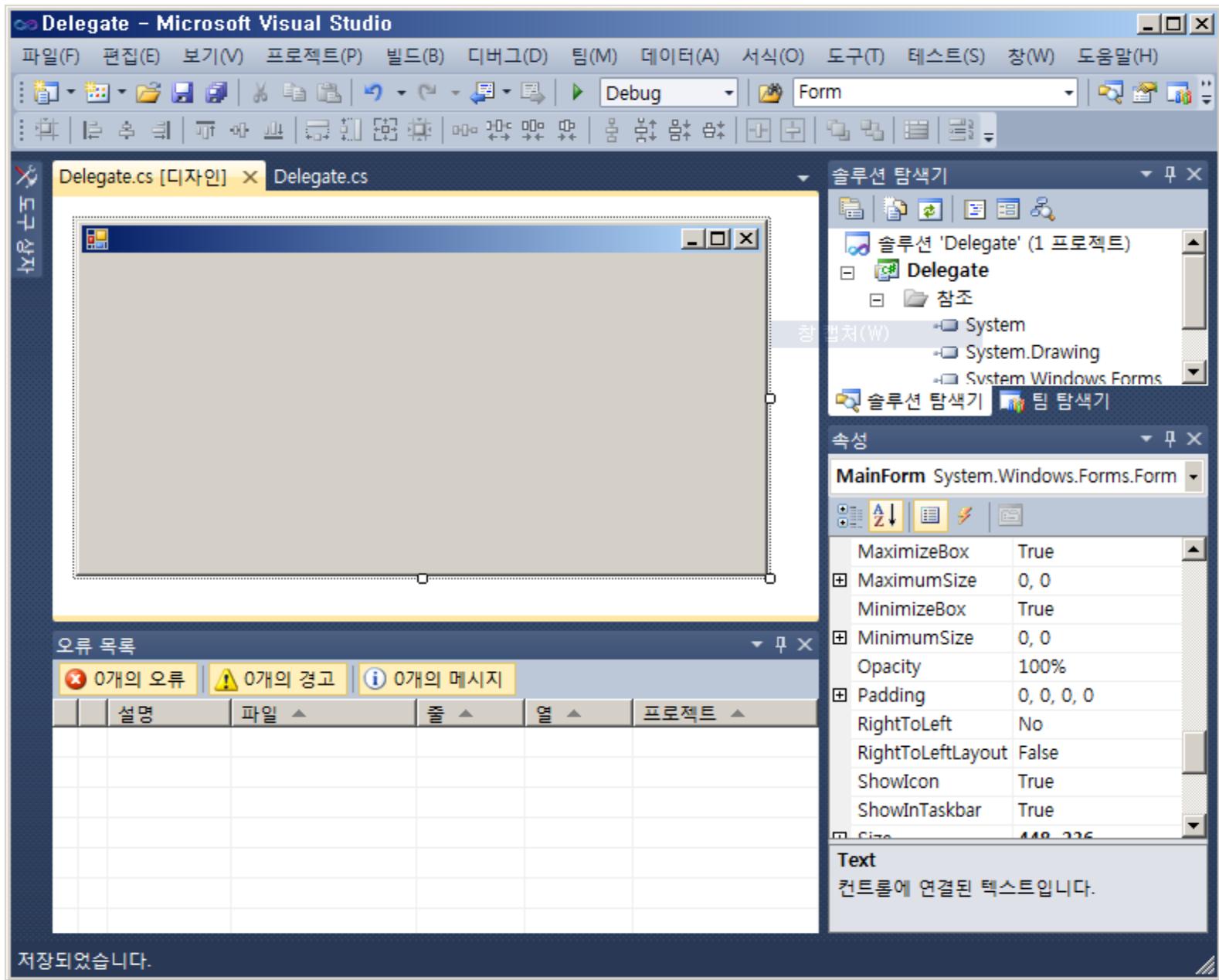
파일 이름 위에서 오른쪽
마우스 버튼 클릭하여
'이름 바꾸기'



02. 폼 윈도우 비주얼 디자인

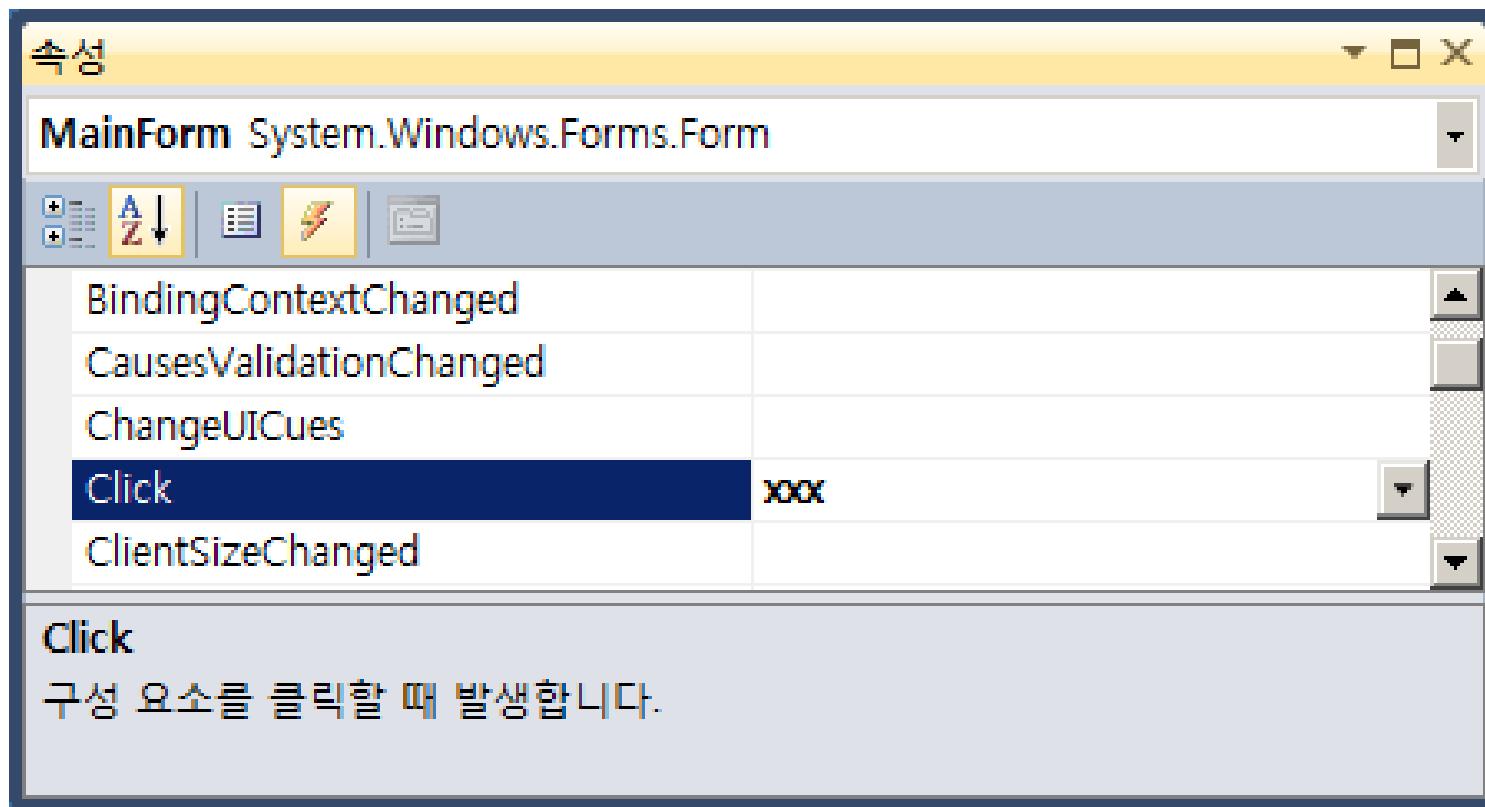
- MainForm.cs 클릭하여 폼 윈도우 크기 변경
- 보기 | 속성 메뉴 항목을 선택하여 속성창 표시
- 타이틀(Text 속성) 변경





03. 핸들러 함수 추가

- Click 이벤트 핸들러 함수 xxx 추가
 - 속성 창 | 번개 아이콘 | Click 이벤트 옆에 xxx 입력

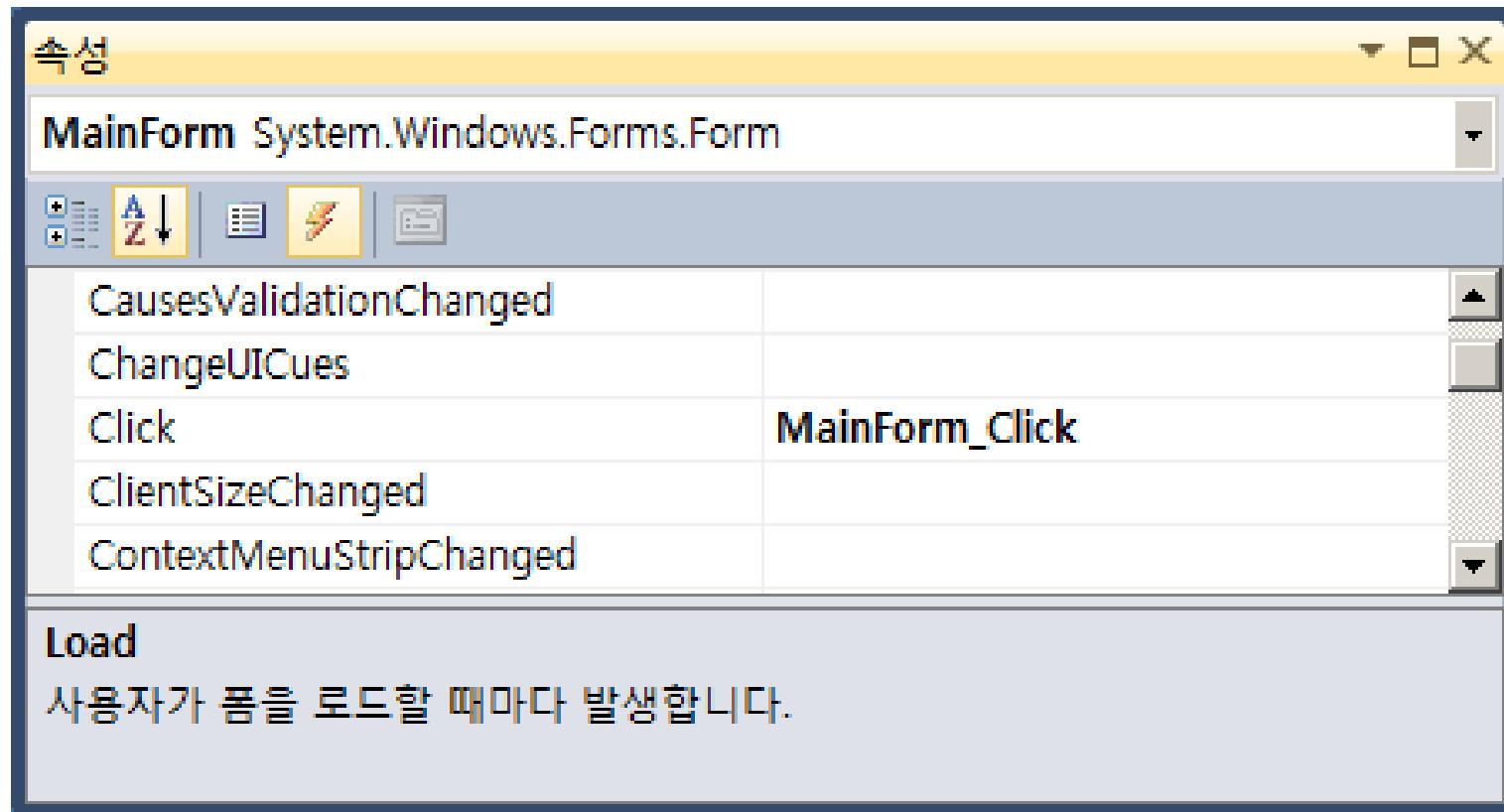


04. 자동으로 생성된 코드 확인

- Click 델리게이트 초기화 코드
 - InitializeComponent 멤버함수 내
- xxx 핸들러 함수 (사용자 코드)

05. 핸들러 함수 이름 변경 연습하기

- 속성 창에서 xxx → MainForm_Click으로 수정



06. 핸들러 함수 코드 작성 연습하기

- 폼 윈도(길동이 얼굴) 클릭할 경우 “Click!”
문자열 출력

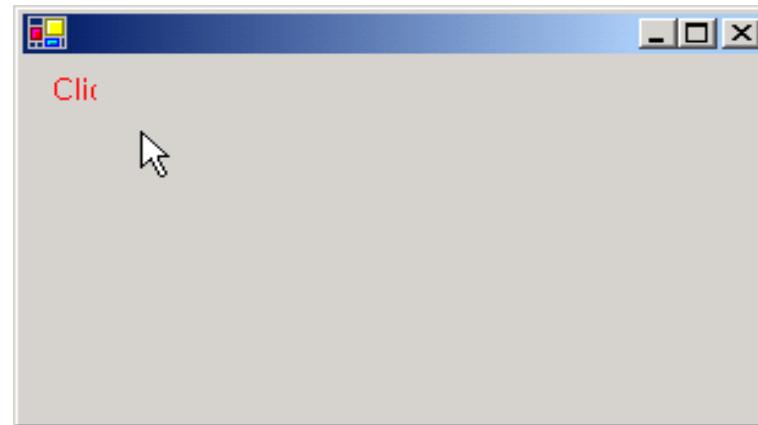
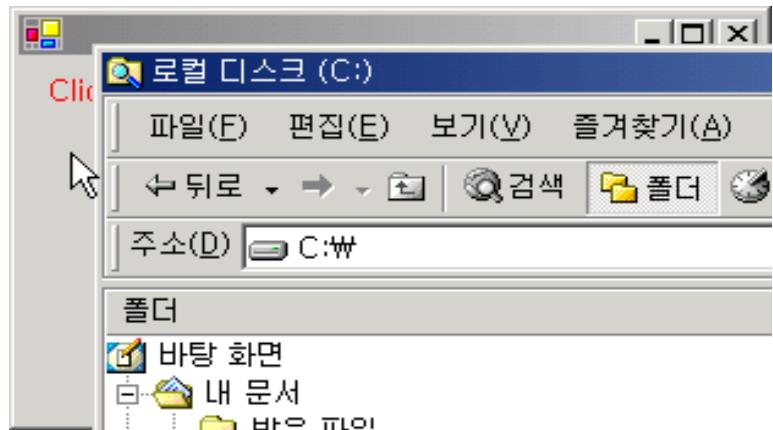
```
private void MainForm_Click(object sender, EventArgs e)
{
    Graphics cheolsu = this.CreateGraphics();
    cheolsu.DrawString("Click!", Font,
        new SolidBrush(Color.Blue), 10, 10);
}
```

화가 객체

스텐실(Stencil) 기법



07. 문제점



08. Paint 이벤트와 핸들러 함수 작성

- Paint 이벤트
 - 폼 윈도가 다시 그려져야 할 때마다 ‘자동으로’ 발생하는 이벤트
- 속성창에서 핸들러 함수 MainForm_Paint 작성 후 코드입력

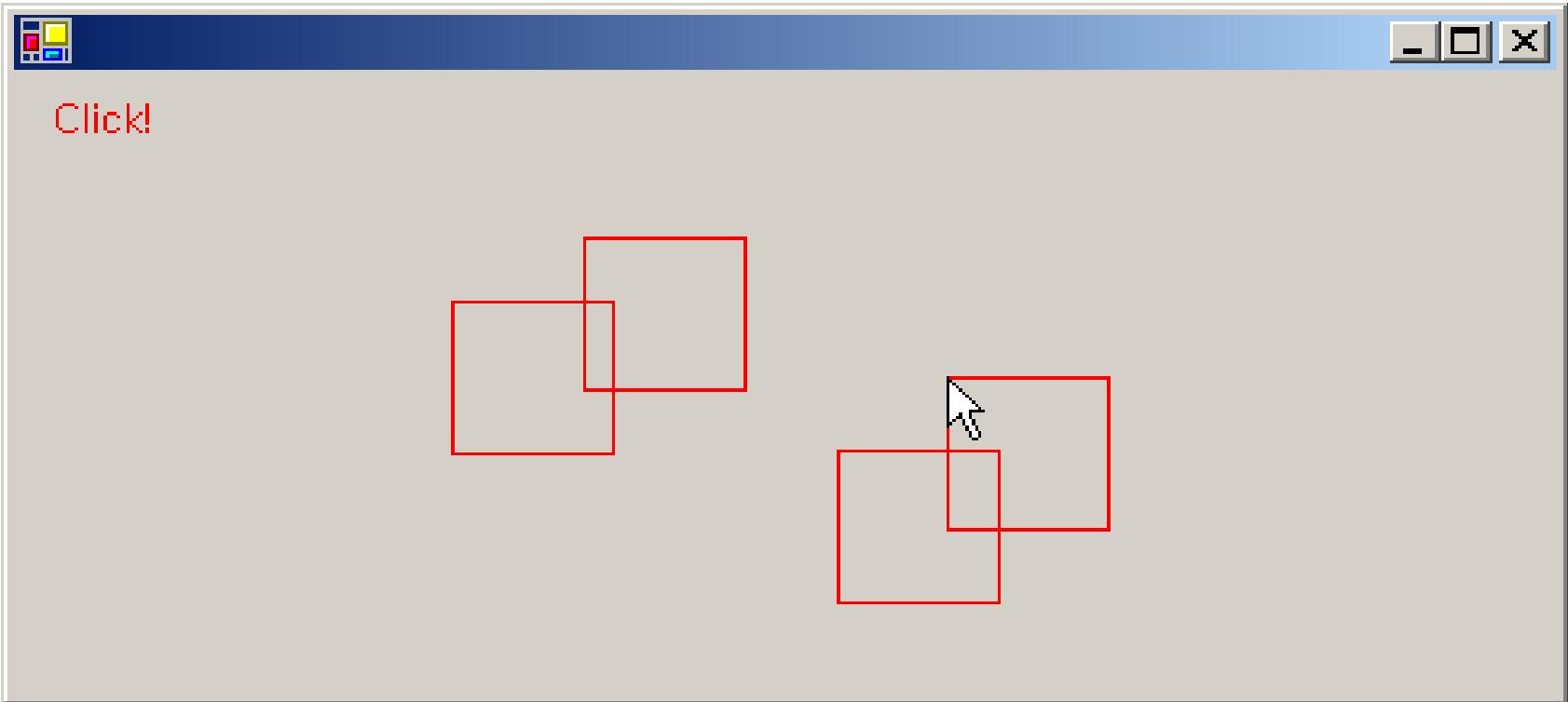
```
Graphics cheolsu = this.CreateGraphics(); //화가
```

```
cheolsu.DrawString("Click!", Font,  
new SolidBrush(Color.Blue), 10, 10);
```

09. MouseDown 이벤트 핸들러 작성

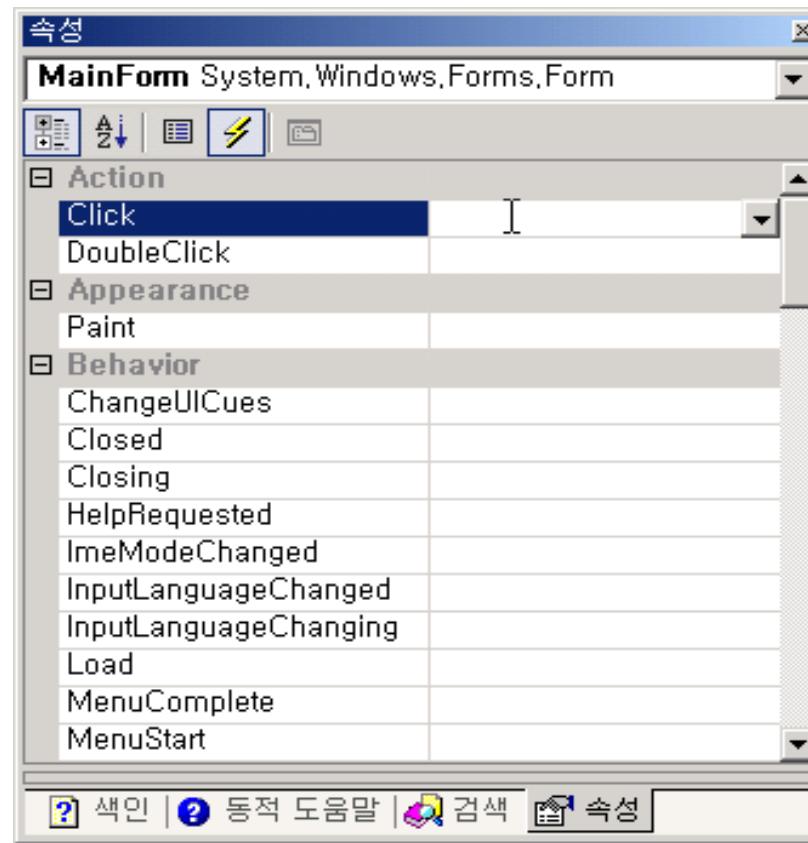
- Click과 마찬가지로 마우스 버튼을 누르면 발생하는 이벤트
 - MouseUp 이벤트는?
- 결국 클릭하면 발생하는 이벤트들은?

```
private void MainForm_MouseDown(object sender,  
MouseEventArgs e)  
{  
    Graphics cheolsu = this.CreateGraphics();  
    cheolsu.DrawRectangle(new Pen(Color.Red), e.X, e.Y, 50, 50);  
}
```



10. 핸들러 함수 제거 연습하기

- 속성 창에서 Click 핸들러 함수 이름 MainForm_Click을 지운 후 Ctrl + S 키 눌러 저장

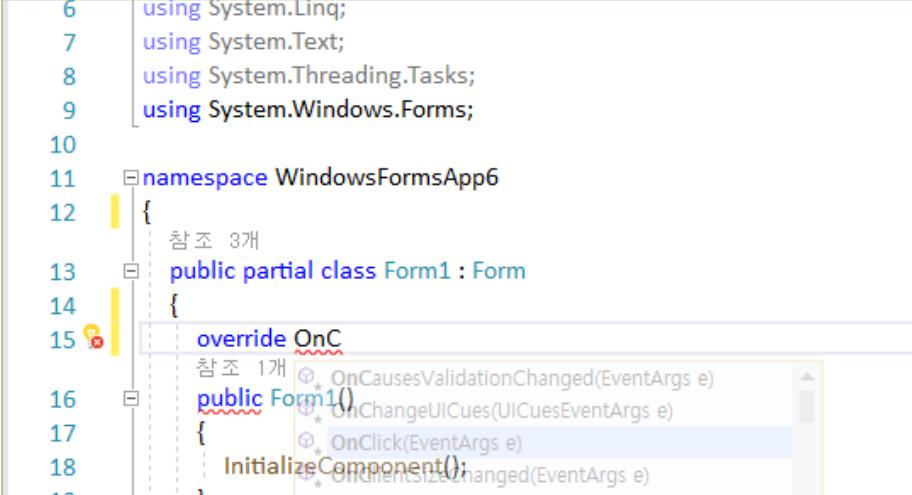


10. 핸들러 함수 제거 연습하기

- 하지만,
 - MainForm_Click 핸들러 함수 자체는 제거되지 않음.
- 왜냐하면
 - 혹시나 실수로 힘들게 작성한 코드가 삭제될 수도 있으므로...
- 따라서
 - 핸들러 함수를 확인하고 직접 본인이 삭제하라는 의미

11. 가상함수 오버라이딩 연습하기

- override를 입력하여 선택하면 자동완성



The screenshot shows a code editor with the following C# code:

```
6  using System.Linq;
7  using System.Text;
8  using System.Threading.Tasks;
9  using System.Windows.Forms;
10
11 namespace WindowsFormsApp6
12 {
13     public partial class Form1 : Form
14     {
15         override OnC
16         public Form1()
17         {
18             InitializeComponent();
19         }
20     }
21 }
```

The cursor is at line 15, position 10, where the code `override OnC` is being typed. A tooltip appears, listing event handlers for the `OnC` prefix:

- OnCausesValidationChanged(EventArgs e)
- OnChangeUICues(UICuesEventArgs e)
- OnClick(EventArgs e) — This item is highlighted.
- OnClientSizeChanged(EventArgs e)

```
protected override void OnClick(EventArgs e)
{
    //base.OnClick(e);
}
```

12. 폼 윈도에 버튼 추가 (직접 코딩)

```
using System.Windows.Forms;
using System.Drawing;
using System;

class MainForm : Form
{
    Button btn;

    public void InitializeComponent()
    {
        btn = new Button();
        btn.Text = "Hello";
        btn.Location = new Point(100, 100);
        btn.Size = new Size(100, 30);
        Controls.Add(btn);
    }
}
```

“직접 작성하니 매우
귀찮다!”

13. 버튼 Click에 핸들러 함수 연결(직접 코딩)

```
using System.Windows.Forms;  
using System.Drawing;  
using System;  
  
class MainForm : Form  
{  
    Button btn;  
  
    public void InitializeComponent()  
    {  
        btn = new Button();  
        btn.Text = "Hello";  
        btn.Location = new Point(100, 100);  
        btn.Size = new Size(100, 30);  
        btn.Click += new System.EventHandler(this.yyy);  
        Controls.Add(btn);  
    }  
}
```

“직접 작성하니 매우
귀찮다!”

버튼 및 관련 코드
제거해 보기

14. 폼 윈도에 버튼 추가하기 (도구 이용)

- 비주얼 디자인
 - 폼 디자인 열기
 - 보기 | 도구 상자 메뉴 선택
 - 고정 핀 눌러 도구 상자 항상 표시되도록 고정
 - 버튼(Button) 드래그-앤-드롭

“마우스로 비주얼 디자인
하니 쉽고 편하다!”

