제 9 장 대화상자

- 대화상자
- 메시지 상자
- 공통 대화상자

대화상자

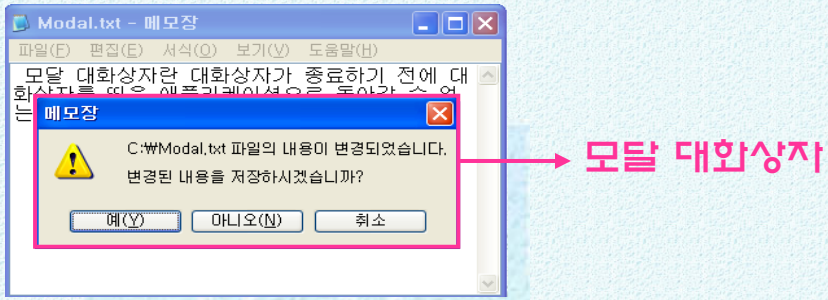
- 대화상자의 용도
  - 사용자와 애플리케이션간의 교량 역할 .
  - 주로 소량의 데이터를 입출력하기 위한 수단.
- 대화상자의 생성방법
  - 모달(modal) 대화상자
  - 모덜리스(modeless) 대화상자
- 대화상자의 종류
  - 메시지 대화상자
    - 사용자에게 간단한 메시지 표현.
  - 공통 대화상자
    - 윈도우 운영체제에서 기본적으로 제공.
    - 열기, 저장, 글꼴, 색, 인쇄, 페이지 설정 등.

대화상자 – 모달 대화상자

- 대화상자가 종료되기 전에 대화상자를 띄운 애플리케이션으로 돌아갈 수 없음.
- 모달 대화상자 만드는 방법
  - Form 클래스의 멤버인 ShowDialog() 메소드 이용.
  - 모달 대화상자 만들기 예

```
Form2 form2 = new Form2();
form2.ShowDialog(); // form2를 모달 방식으로 띄운다.
```

- 모달 대화상자 예
  - 메모장에서 편집내용을 저장하지 않고 종료할 때.

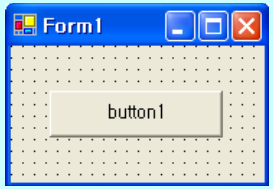


[예제 9.1 - ModalApp.cs]

- Form1에서 버튼을 클릭하여 Form2를 모달방식으로 띄우는 예제.

Form1

1) 폼 설계



2) 프로퍼티

컨트롤 : (Name)	프로퍼티	값
Form : Form1	Text	ModalApp
Button1 : button1	Text	Modal

3) 이벤트 처리기

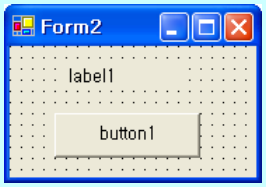
컨트롤 : (Name)	이벤트	메소드 명
Button : button1	Click	button1_Click()

```
private void button1_Click(object sender, EventArgs e)
{
    Form2 form2 = new Form2();
    form2.ShowDialog();                    // form2를 모달 방식으로 띄운다.
}
```

[예제 9.1 - ModalApp.cs]

❖ Form2

1) 폼 설계



2) 프로퍼티

컨트롤 : (Name)	프로퍼티	값
Form : Form2	Text	ModalDialogBox
Label : label1	Text	모달 대화상자
Button1 : button1	Text	닫기

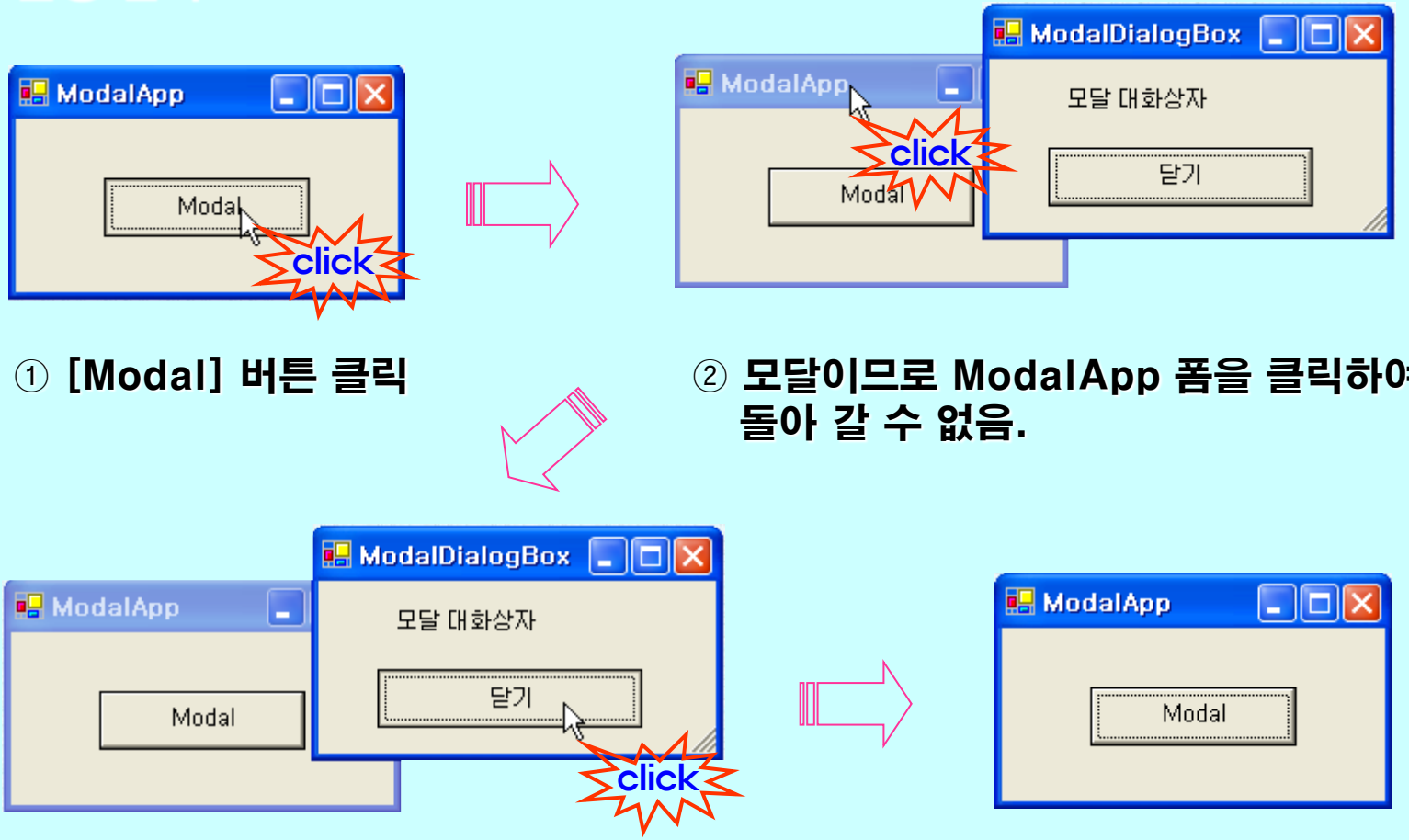
3) 이벤트 처리기

컨트롤 : (Name)	이벤트	메소드 명
Button : button1	Click	button1_Click()

```
private void button1_Click(object sender, EventArgs e)
{
        this.close();
 }
```

[예제 9.1 - ModalApp.cs]

❖ 실행 결과

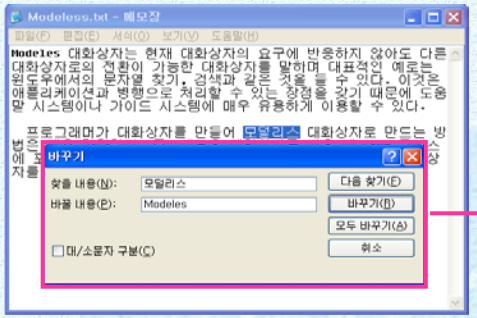


대화상자 – 모덜리스 대화상자

- 현재 대화상자의 요구에 반응하지 않아도 다른 대화상자로 전환 가능.
  - 문자열 찾기, 검색, 도움말 기능 구현에 유용.
- 모덜리스 대화상자 만드는 방법
  - Form 클래스의 멤버인 Show() 메소드 이용.
  - 모덜리스 대화상자 만들기 예

```
Form2 form2 = new Form2();
form2.Show();      // form2를 모덜리스 방식으로 띄운다.
```

- 모덜리스 대화상자 예
  - 메모장에서 문자열 바꾸기 대화상자.



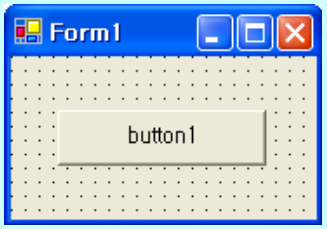
→ 모덜리스 대화상자

[예제 9.2 - ModelessApp.cs]

- Form1에서 버튼을 클릭하여 Form2를 모덜리스방식으로 띄우는 예제.

Form1

1) 폼 설계



2) 프로퍼티

컨트롤 : (Name)	프로퍼티	값
Form : Form1	Text	ModelessApp
Button1 : button1	Text	Modeless

3) 이벤트 처리기

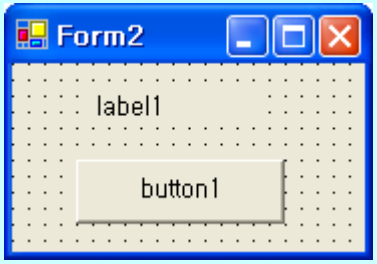
컨트롤 : (Name)	이벤트	메소드 명
Button : button1	Click	button1_Click()

```
private void button1_Click(object sender, EventArgs e){
    Form2 form2 = new Form2();
    form2.Show();          // form2를 모덜리스 방식으로 띄운다.
}
```

[예제 9.2 - ModelessApp.cs]

- Form2

1) 폼 설계



2) 프로퍼티

컨트롤 : (Name)	프로퍼티	값
Form : Form2	Text	ModelessDialogBox
Label : label1	Text	**모덜리스 대화상자**
Button1 : button1	Text	**닫기**

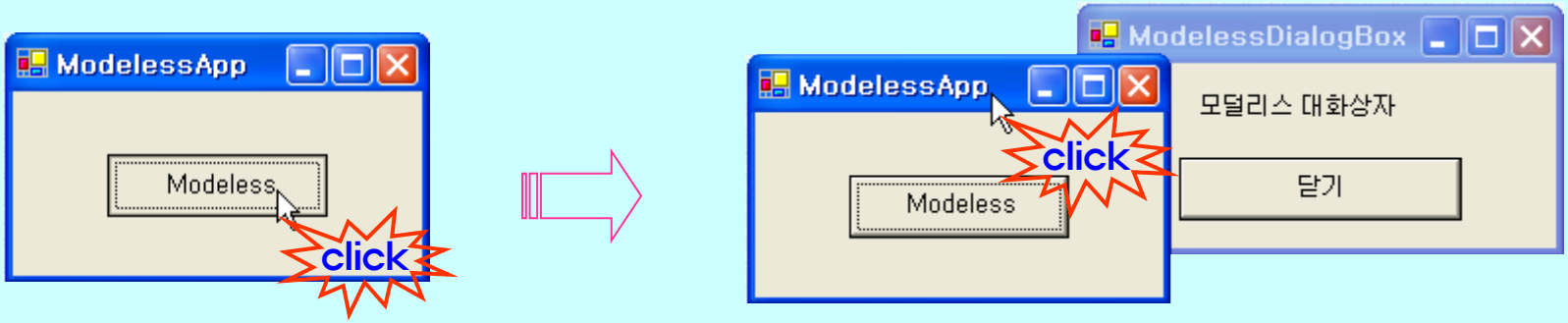
3) 이벤트 처리기

컨트롤 : (Name)	이벤트	메소드 명
Button : button1	Click	button1_Click()

```
private void button1_Click(object sender, EventArgs e) {
        this.close();
 }
```

[예제 9.1 - ModalApp.cs]

❖ 실행 결과



① [Modeless] 버튼 클릭.

② 모덜리스이므로 ModelessApp 폼으로 돌아 갈 수 있음.

메시지 상자

- 사용자에게 간단한 메시지를 전달할 때 사용
- MessageBox 클래스의 멤버인 Show() 메소드 이용
- 12개의 중복된 Show() 메소드 중 기본 형식

```
MessageBox.Show(message);                                    ①
MessageBox.Show(message, caption);                           ②
MessageBox.Show(message, caption, buttonKind);               ③
MessageBox.Show(message, caption, buttonKind, iconKind);     ④
```

메시지 상자 - 버튼

- 버튼의 종류
  - MessageBoxButtons 열거형의 멤버로서 5가지

기호상수 (멤버이름)	순서 값	버 튼 모 양	설 명
OK	0	확인	OK 버튼
OKCancel	1	확인 취소	OK, Cancel 버튼
AbortRetryIgnore	2	중단(A) 다시시도(R) 무시(I)	Abort, Retry, Ignore 버튼
YesNoCancel	3	예(Y) 아니오(N) 취소	Yes, No, Cancel 버튼
YesNo	4	예(Y) 아니오(N)	Yes, No 버튼
RetryCancel	5	다시시도(R) 취소	Retry, Cancel 버튼

메시지 상자 - 아이콘

- 아이콘의 종류
  - MessageBoxIcon 열거형의 멤버로서 9개의 기호상수.
  - 아이콘의 모양은 4가지

기호상수 (멤버이름)	순서 값	아이콘 모양	설 명
None	0		기호 없음.
Error Hand Stop	16		빨간색 배경의 원 안에 흰색 X가 포함된 기호.
Question	32		풍선 안에 물음표가 포함된 기호.
Exclamation Warning	48		노란색 배경의 삼각형 안에 느낌표가 있는 기호.
Asterisk Information	64		풍선 안에 소문자 i 가 포함된 기호.

메시지 상자 – 기본 버튼 설정

- 기본 버튼
  - 메시지 상자가 활성화 될 때 초기에 입력포커스를 갖는 버튼.
  - 기본 버튼을 명시하지 않으면 첫 번째 버튼이 기본 버튼.

- 기본 버튼 설정 방법
  - MessageBoxDefaultButton 열거형 멤버를 매개변수로 갖는 Show() 메소드 이용.

```
Show(message, caption, buttonKind, iconKind, MessageBoxDefaultButton);
```

  - MessageBoxDefaultButton 열거형

기호상수 (멤버이름)	순서 값	설 명
Button1	0x000	왼쪽을 기준으로 첫 번째 버튼을 기본으로 설정
Button2	0x100	왼쪽을 기준으로 두 번째 버튼을 기본으로 설정
Button3	0x200	왼쪽을 기준으로 세 번째 버튼을 기본으로 설정

[예제 9.6 - MessageBoxDefaultButtonApp.cs]

- **폼을 클릭하여 메시지 박스를 띄우는 예제.**

1) 폼 설계

2) 프로퍼티

컨트롤 : (Name)	프로퍼티	값
Form : Form1	Text	MessageBoxDefaultButtonApp

3) 이벤트 처리기

컨트롤 : (Name)	이벤트	메소드 명
Form : Form1	Click	Form1_Click()

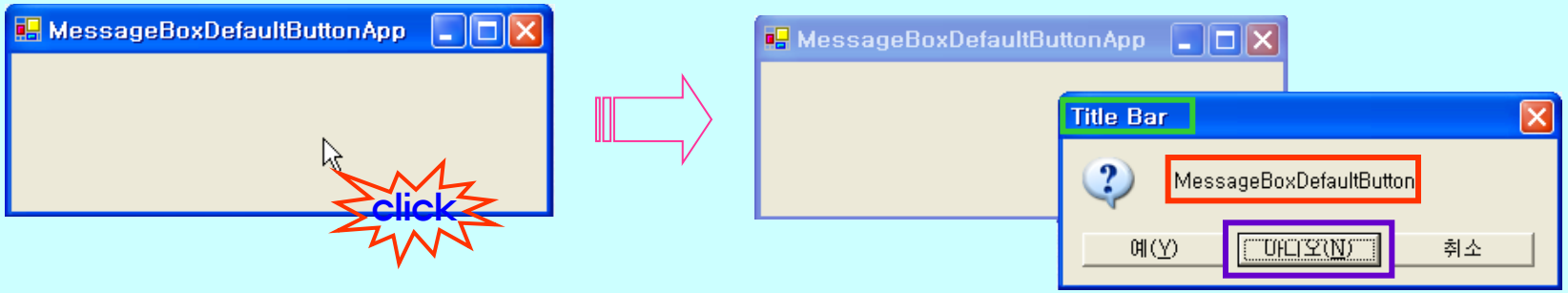
```
private void Form1_Click(object sender, EventArgs e) {
    MessageBox.Show("MessageBoxDefaultButton", "Title Bar",
                    MessageBoxButtons.YesNoCancel,
                    MessageBoxIcon.Question,
                    MessageBoxDefaultButton.Button2);
}
```

[예제 9.6 - MessageBoxDefaultButtonApp.cs]

☺ 참 고

❖ 실행 결과

```
MessageBox.Show("MessageBoxDefaultButton",
                "Title Bar",
                 MessageBoxButtons.YesNoCancel,
                 MessageBoxIcon.Question,
                 MessageBoxDefaultButton.Button2);
```



① MessageBoxDefaultButtonApp 폼 클릭

공통 대화상자

- 윈도우 운영체제에서 기본적으로 제공.
- 정형화된 대화상자
- CommonDialog 클래스의 파생 클래스
  - FileDialog
    - OpenFileDialog (파일 열기)
    - SaveFileDialog (파일 저장)
  - FontDialog (글꼴)
  - ColorDialog (색)
  - PrintDialog (인쇄)
  - PageSetupDialog (페이지 설정)
  - FolderBrowserDialog (폴더 탐색/생성)

공통 대화상자 – 열기 대화상자

- 드라이브, 폴더, 파일 확장자를 설정하여 원하는 형식의 파일을 찾을 수 있는 기능 제공.

- OpenFileDialog 컴포넌트의 주요 프로퍼티

프로퍼티	설 명
FileName	대화상자에서 선택된 절대경로 형태로 구성된 파일명.
FileNames	Multiselect 프로퍼티가 참으로 설정된 경우에 파일명들을 나타내는 스트링 배열.
Filter	콤보 상자에 표시될 문자열(파일 형식)과 해당 파일 형식을 선택할 때 사용하게 될 확장자. "파일형식1|확장자1|파일형식2|확장자2..." 형식으로 명시.
FilterIndex	대화상자에서 현재 선택된 Filter 프로퍼티의 인덱스.
InitialDirectory	대화상자에 표시하는 초기 디렉토리.
RestoreDirectory	종료 전에 초기 디렉토리로 되돌아갈 지의 여부.
Multiselect	대화상자에서 여러 파일들을 선택할 수 있는 지의 여부.

[예제 9.8 - OpenFileDialogApp.cs]

- 버튼을 클릭하여 열기대화상자를 띄우고 선택한 파일의 경로와 이름을 텍스트 상자에 출력하는 예제.

1) 폼 설계

2) 컴포넌트

컴포넌트 : (Name)	프로퍼티	값
OpenFileDialog : openFileDialog1	ShowColor	True

3) 프로퍼티

컨트롤 : (Name)	프로퍼티	값
Form : Form1	Text	MultiSelectApp
Label : label1	Text	**파일 찾기**
TextBox : textbox1	Text	
	Multiline	True

[예제 9.8 - OpenFileDialogApp.cs]

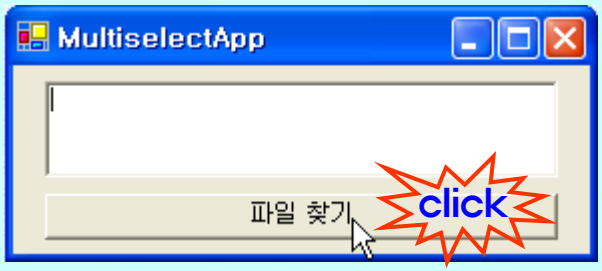
4) 이벤트 처리기

컨트롤 : (Name)	이벤트	메소드 명
Button : button1	Click	button1_Click()

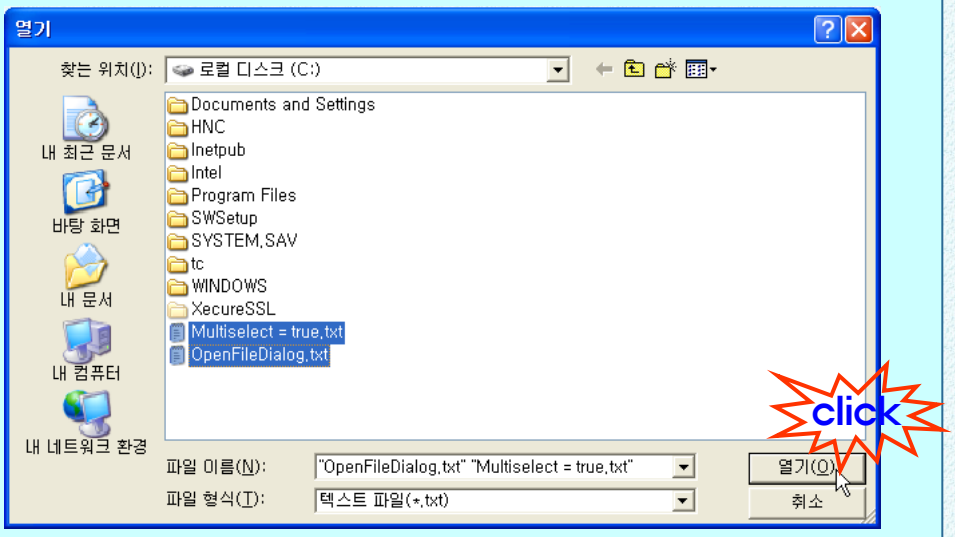
```
private void button1_Click(object sender, EventArgs e)
{
    openFileDialog1.InitialDirectory = @"C:\";
    openFileDialog1.Filter = "텍스트 파일(*.txt)|*.txt|모든 파일(*.*)|*.*";
    openFileDialog1.FilterIndex = 1;
    openFileDialog1.RestoreDirectory = true;
    openFileDialog1.Multiselect = true;
    openFileDialog1.ShowDialog();
    foreach(string strTmp in openFileDialog1.FileNames)
    {
        textBox1.Text += strTmp;
        textBox1.Text += "\r\n";
    }
}
```

[예제 9.8 - OpenFileDialogApp.cs]

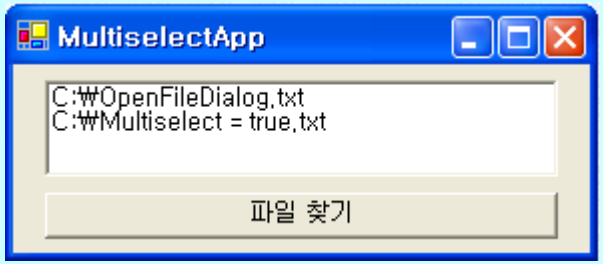
❖ 실행 결과



① [파일 찾기] 버튼 클릭.



② 2개 이상의 파일을 선택하여 [열기]버튼 클릭.



③ 텍스트 박스에 선택된 파일의 리스트가 출력됨.

공통 대화상자 – 글꼴 대화상자

- 글꼴, 글자의 크기, 글자의 색상, 형태 등을 설정할 수 있는 기능 제공.

- FontDialog 컴포넌트의 주요 프로퍼티

프로퍼티	설 명
Color	글꼴 대화상자에서 선택된 색상
Font	글꼴 대화상자에서 선택한 글꼴 및 글꼴의 크기
ShowApply	글꼴 대화상자에 [적용] 버튼의 추가 여부 제어 True : 추가, False : 추가하지 않음
ShowColor	글꼴 대화상자에서 색 콤보박스 추가 여부 제어 True : 추가, False : 추가하지 않음

[예제 9.9 - FontDialogApp.cs]

- **버튼을 클릭하여 글꼴대화상자를 띄우고 텍스트 상자의 글 속성들을 변경시키는 예제.**

1) 폼 설계

2) 컴포넌트

컴포넌트 : (Name)	프로퍼티	값
FontDialog : fontDialog1	ShowColor	True

3) 프로퍼티

컨트롤 : (Name)	프로퍼티	값
Form : Form1	Text	FontDialogApp
Label : label1	Text	글꼴 변경
TextBox : textbox1	Text	
	Multiline	True

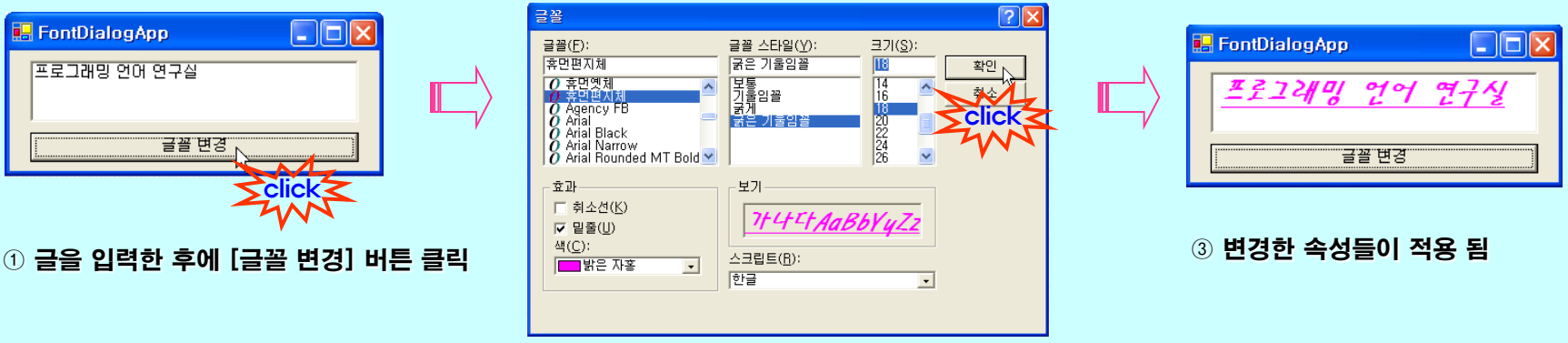
[예제 9.9 - FontDialogApp.cs]

4) 이벤트 처리기

컨트롤 : (Name)	이벤트	메소드 명
Button : button1	Click	button1_Click()

```
private void button1_Click(object sender, EventArgs e) {
        fontDialog1.ShowDialog();
        textBox1.Font = fontDialog1.Font;
        textBox1.ForeColor = fontDialog1.Color;
}
```

❖ 실행 결과



① 글을 입력한 후에 [글꼴 변경] 버튼 클릭

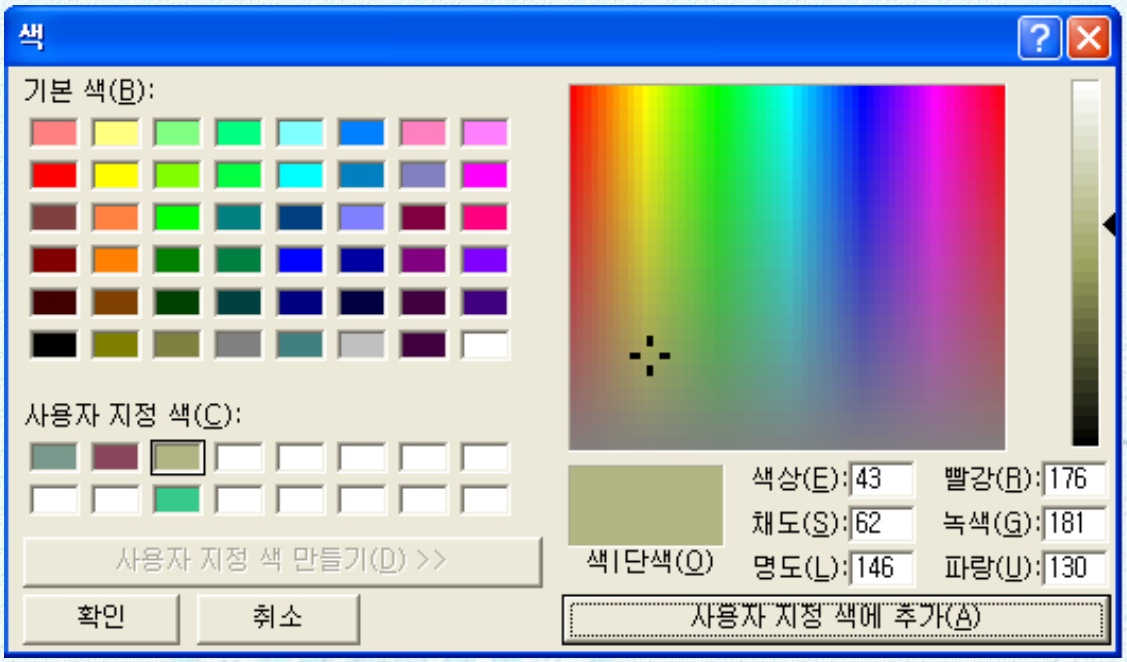
② 글의 속성들을 변경한 후에 [확인] 버튼 클릭

③ 변경한 속성들이 적용 됨

공통 대화상자 – 색 대화상자

- 색상표에서 기본 색을 선택하거나 사용자 지정 색을 만들어 사용할 수 있는 기능 제공.

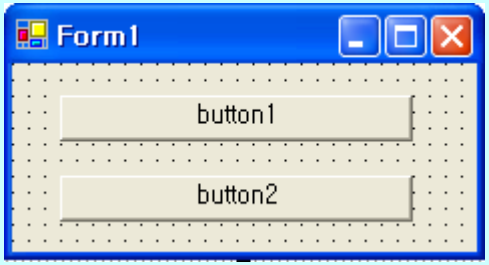
- ColorDialog 컴포넌트 이용



[예제 9.10 - ColorDialogApp.cs]

- button1을 클릭하여 폼의 배경색을 변경하고 button2를 클릭하여 버튼의 배경색을 변경하는 예제.

1) 폼 설계



2) 컴포넌트

컴포넌트 : (Name)	프로퍼티	값
ColorDialog : colorDialog1		

3) 프로퍼티

컨트롤 : (Name)	프로퍼티	값
Form : Form1	Text	ColorDialogApp
Button : button1	Text	폼 색상 변경
Button : button2	Text	버튼 색상 변경

[예제 9.10 - ColorDialogApp.cs]

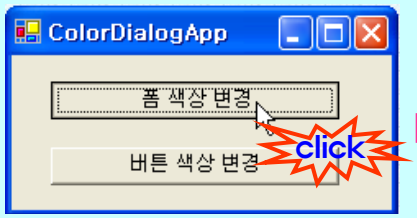
4) 이벤트 처리기

컨트롤 : (Name)	이벤트	메소드 명
Button : button1	Click	button1_Click()
Button : button2	Click	button2_Click()

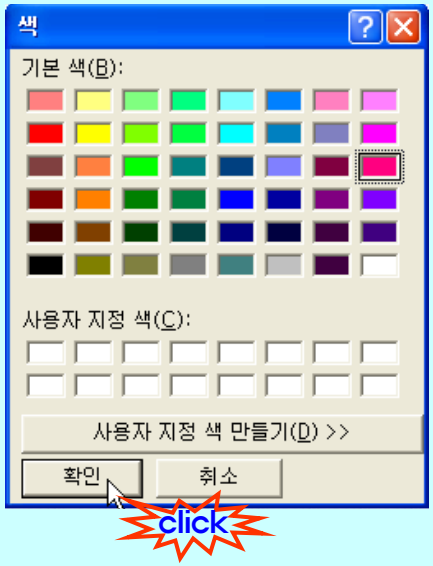
```
private void button1_Click(object sender, EventArgs e)
{
    colorDialog1.ShowDialog();
    this.BackColor = colorDialog1.Color;        // 폼의 배경 색
}

private void button2_Click(object sender, EventArgs e)
{
    colorDialog1.ShowDialog();
    button1.BackColor = colorDialog1.Color;  //  버튼의 배경 색
    button2.BackColor = colorDialog1.Color;
}
```

[예제 9.10 - ColorDialogApp.cs]

❖ 실행 결과



① [폼 색상 변경] 버튼 클릭.



② 색을 선택하고 [확인] 버튼 클릭.

③ [버튼 색상 변경] 버튼을 클릭하여 ②와 같이 색 선택.

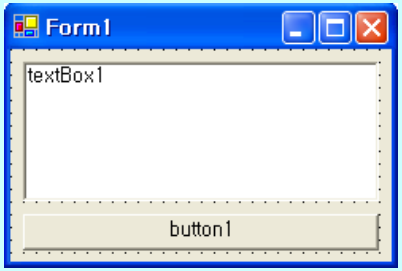
공통 대화상자 – 인쇄 대화상자

- 인쇄할 프린터, 인쇄 범위, 인쇄 매수 등을 선택할 수 있는 기능 제공.
- 인쇄 대화상자 만들기
  - PrintDialog 컴포넌트와 두 개의 클래스가 더 필요.
    - System.Drawing.Printing.**PrinterSettings** (기본프린터 설정)
    - System.Drawing.Printing.**PrintDocument** (출력물 설정)
      - PrintPage 이벤트에 PrintPageEventHandler 델리게이트 등록.
  - VS .NET IDE 환경에서 System.Drawing.Printing 네임스페이스 추가.

[예제 9.11 - PrintDialogApp.cs]

- **버튼을 클릭하여 인쇄 대화상자를 띄우고 테스트 상자의 내용을 프린터로 출력하는 예제.**

1) 폼 설계



2) 컴포넌트

컴포넌트 : (Name)	프로퍼티	값
PrintDialog : printDialog1		

3) 프로퍼티

컨트롤 : (Name)	프로퍼티	값
Form : Form1	Text	PrintDialogApp
TextBox : textBox1	Multiline	True
	Text	
Button : button1	Text	출력

[예제 9.10 - ColorDialogApp.cs]

4) 이벤트 처리기

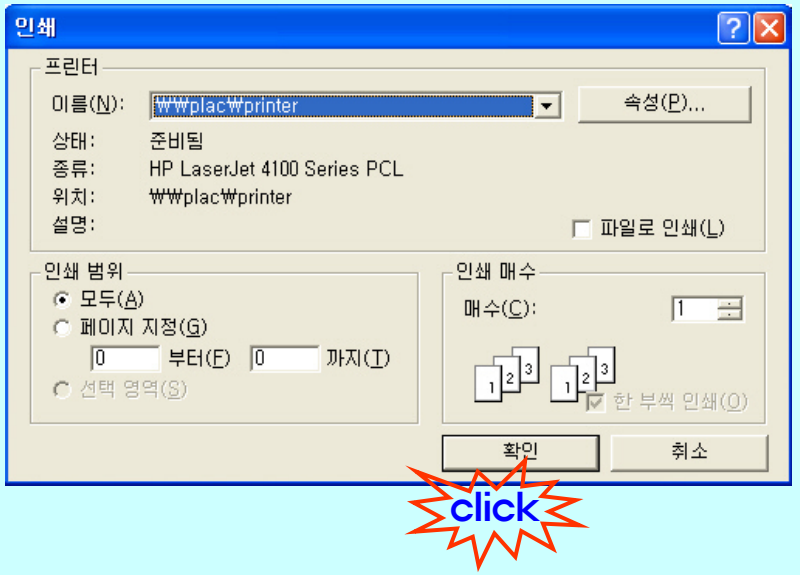
컨트롤 : (Name)	이벤트	메소드 명
Button : button1	Click	button1_Click()

```
private void button1_Click(object sender, EventArgs e){
    PrinterSettings printer = new PrinterSettings();
    PrintDocument pd = new PrintDocument();
    printDialog1.PrinterSettings = printer;
    printDialog1.Document = pd;
                    // PrintPage 이벤트는 Print() 메소드가 호출되기 직전 발생
    pd.PrintPage += new PrintPageEventHandler(this.pd_PrintPage);
    DialogResult result = printDialog1.ShowDialog();
    if (result==DialogResult.OK){
        pd.Print();
    }
}
private void pd_PrintPage(object sender, PrintPageEventArgs e){
    string text = textBox1.Text;
    Font printFont = new Font("Arial", 10, FontStyle.Regular);
    e.Graphics.DrawString(text, printFont, Brushes.Black, 10, 10);
}
```

[예제 9.10 - ColorDialogApp.cs]

❖ 실행 결과

① 텍스트 상자에 글자를 입력하고 [출력] 버튼 클릭.



② 프린터를 선택하고 [확인] 버튼 클릭하면 문서가 출력됨.