



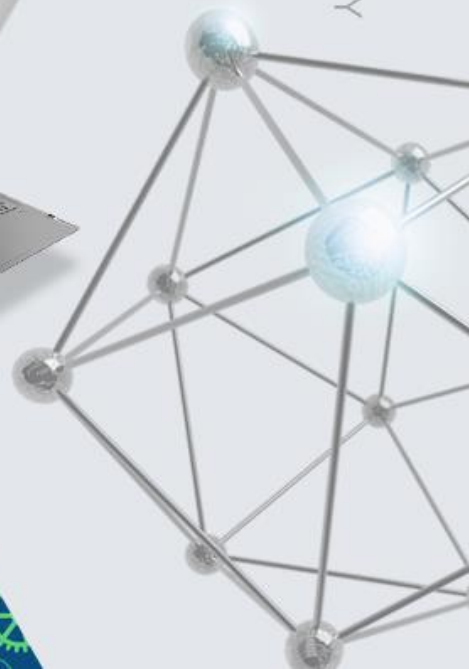
한국기술교육대학교
온라인평생교육원

The 4th Industrial Revolution is characterized by super connectivity and super intelligence, where various products and services are connected to the network, and artificial intelligence and information communication technologies are used in 3D printing, unmanned transportation, robotics. Of the world's most advanced technologies.

T
E
C
H
N
O
L
O
G
Y

C# 프로그래밍

컨트롤



The 4th Industrial Revolution is characterized by super connectivity and super intelligence, where various products and services are connected to the network, and artificial intelligence and information communication technologies are used in 3D printing, unmanned transportation, robotics. Of the world's most advanced technologies.

컨트롤



학/습/목/표

1. 버튼 기반 컨트롤을 이해하고 구현할 수 있다.
2. 레이블과 링크 레이블을 이해하고 구현할 수 있다.
3. 텍스트, 리스트, 콤보 그리고 체크리스트 박스를 이해하고 구현할 수 있다.



학/습/내/용

1. 버튼 기반 컨트롤
2. 레이블과 링크 레이블
3. 텍스트와 리스트 상자



1. 버튼 기반 컨트롤

1) 버튼 기반 컨트롤 개념

(1) 개념

- 버튼 기반 컨트롤의 개념
 - ButtonBase 추상 클래스를 상속받은 컨트롤

(2) 종류

- 버튼 기반 컨트롤의 종류



(3) 형태

- 버튼 기반 컨트롤의 기본적인 형태





1. 버튼 기반 컨트롤

2) 버튼

(1) 개념

- 버튼의 개념
 - 사용자 입력을 받는 가장 간단한 방법으로 버튼을 클릭함으로써 이벤트를 발생

(2) 프로퍼티

- 버튼의 프로퍼티
 - 이름과 글자색 등을 설정할 수 있음

프로퍼티	설명
Text	버튼 이름을 설정
Font	버튼 이름에 해당하는 문자열의 글꼴을 설정
ForeColor	버튼 이름에 해당하는 문자열의 색상을 설정
TextAlign	버튼 이름에 해당하는 문자열을 정렬
FlatStyle	버튼의 스타일(모양)을 설정
Size	버튼의 크기(픽셀 단위)를 설정
BackColor	버튼의 배경색을 설정
BackgroundImage	버튼의 배경 이미지를 설정
Image	버튼 형태 위에 이미지를 설정
ImageAlign	버튼 형태 위에 이미지를 정렬






1. 버튼 기반 컨트롤

2) 버튼

(3) 스타일

- 버튼의 스타일
 - 버튼의 스타일을 설정하기 위해서는 FlatStyle 프로퍼티를 지정해야 함
 - System.Windows.Forms 네임스페이스에 포함되어 있음
- 버튼의 스타일
 - 버튼의 4가지 종류
 - FlatStyle 열거형

기호 상수	버튼 모양	설명
Flat		버튼의 모양을 평면으로 표시
Standard		버튼의 모양을 입체적으로 표시
Popup		버튼의 모양이 처음에는 평면으로 표시되었다가 마우스 포인터가 버튼 위로 이동하면 입체적으로 변경
System		버튼의 모양을 시스템이 결정 일반적으로 System으로 지정한 버튼의 모양은 Standard로 지정한 것과 동일

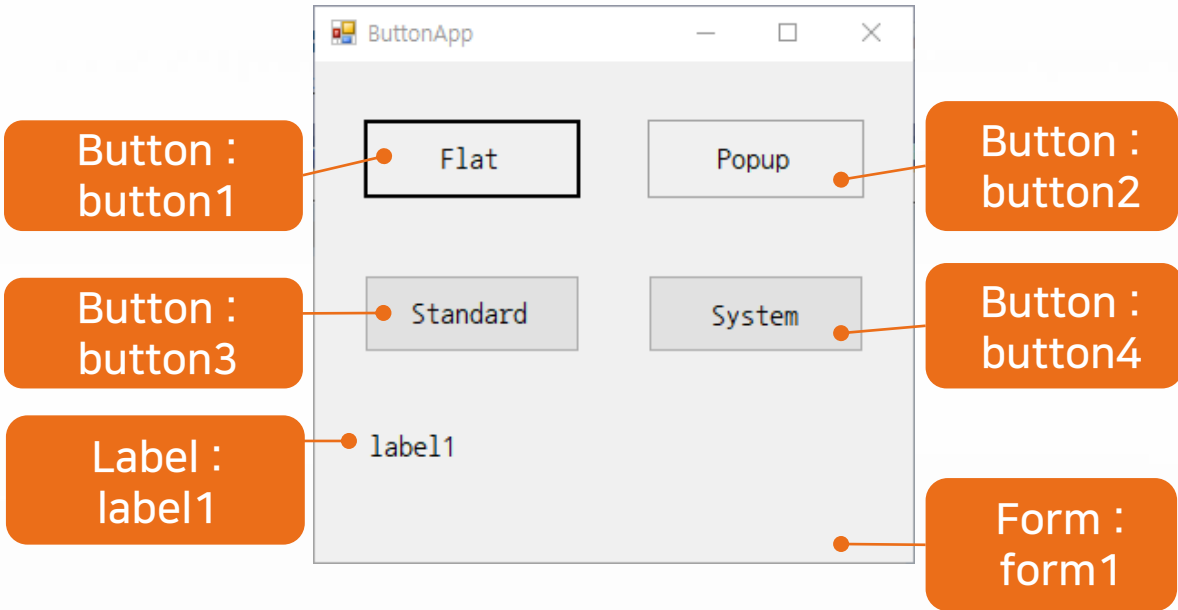


1. 버튼 기반 컨트롤

2) 버튼

(4) 예제

- 예제 : ButtonApp.cs



컨트롤 : (Name)	프로퍼티	값
Form : Form1	Text	ButtonApp
Button : Button1	FlatStyle, Text	Flat
Button : Button2	FlatStyle, Text	Popup
Button : Button3	FlatStyle, Text	Standard
Button : Button4	FlatStyle, Text	System



1. 버튼 기반 컨트롤

2) 버튼

(4) 예제

▪ 예제 : ButtonApp.cs

컨트롤 : (Name)	이벤트	메소드명
Button : Button1	Click	Button1_Click()
Button : Button2	Click	Button2_Click()
Button : Button3	Click	Button3_Click()
Button : Button4	Click	Button4_Click()

▪ 예제:버튼 컨트롤

정의 형태

```
private void button1_Click(object sender, EventArgs e) {  
    label1.Text = FlatStyle.Flat.ToString();  
}  
private void button2_Click(object sender, EventArgs e) {  
    label1.Text = FlatStyle.Popup.ToString();  
}  
private void button3_Click(object sender, EventArgs e) {  
    label1.Text = FlatStyle.Standard.ToString();  
}  
private void button4_Click(object sender, EventArgs e) {  
    label1.Text = FlatStyle.System.ToString();  
}
```

1. 버튼 기반 컨트롤

2) 버튼

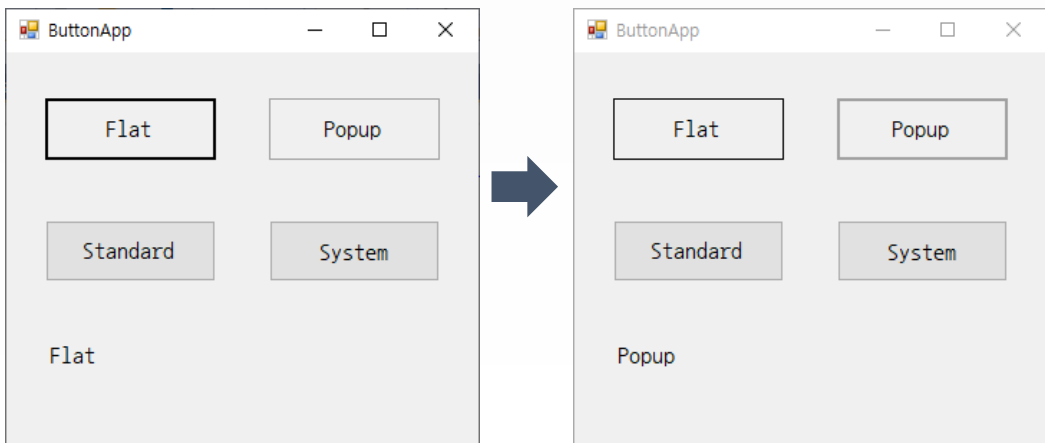
(4) 예제

- 예제: 버튼 컨트롤

실행 방법

각각의 버튼을 클릭

실행 결과





1. 버튼 기반 컨트롤

3) 체크 상자

(1) 개념

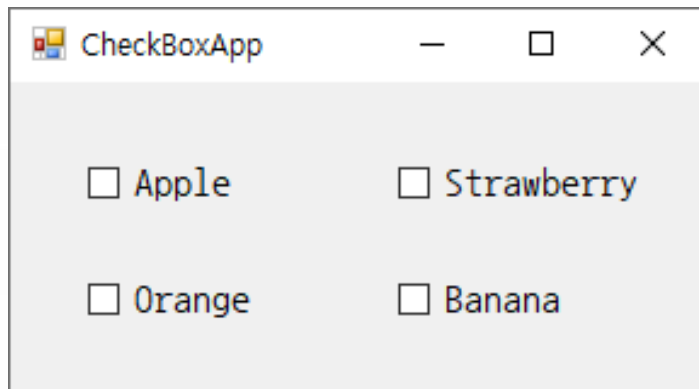
- 체크 상자의 개념

→ 주어진 항목들 중에서 선택할 수 있는 컨트롤

→ 특징 : 주어진 항목을 복수로 선택할 수 있음

(2) 형태

- 체크 상자의 기본적인 형태



(3) 체크 상자의 프로퍼티

- 체크 상자의 프로퍼티

→ Control 클래스를 상속받은 컨트롤이기 때문에 버튼과 대부분 동일

→ 체크 상자의 체크 상태를 나타내는 Checked 프로퍼티가 추가로 제공

1. 버튼 기반 컨트롤

3) 체크 상자

(4) checked 프로퍼티

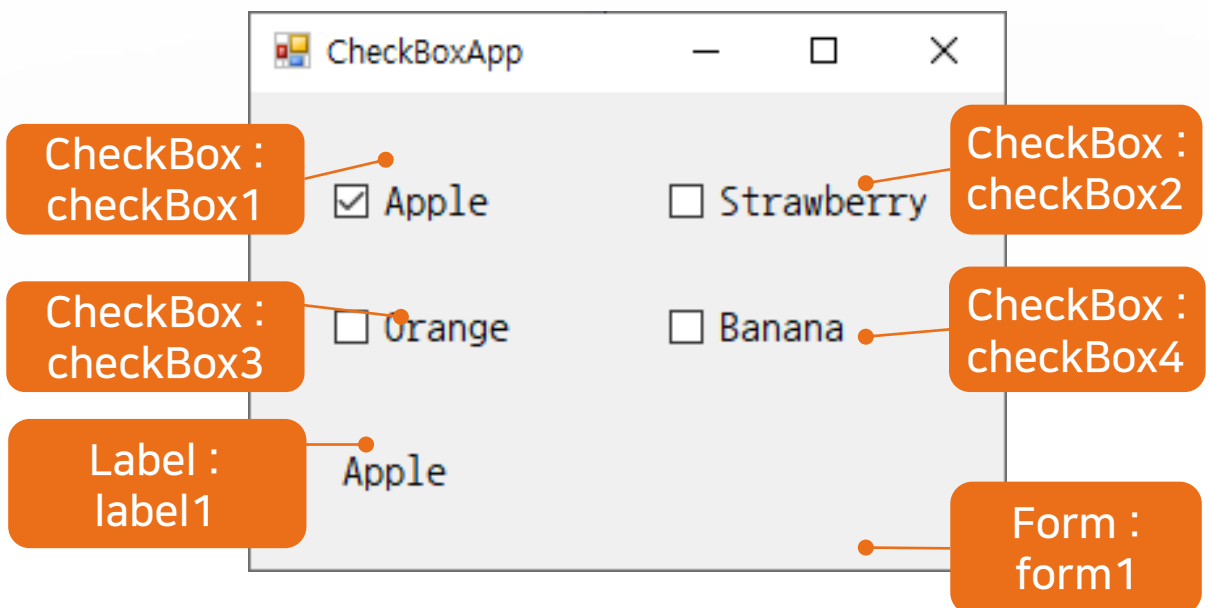
▪ checked 프로퍼티

→ 참으로 설정하면 네모부분에 '✓'가 표시

→ Checked 프로퍼티의 값이 변경될 때마다 CheckChanged 이벤트 발생

(5) 예제

▪ 예제 : CheckBoxApp.cs





1. 버튼 기반 컨트롤

3) 체크 상자

(5) 예제

- 예제 : CheckBoxApp.cs

컨트롤 : (Name)	프로퍼티	값
Form : Form1	Text	CheckBoxApp
CheckBox : checkBox1	Text	Apple
	Checked	True
CheckBox : checkBox2	Text	Strawberry
	Checked	False
CheckBox : checkBox3	Text	Orange
	Checked	False
CheckBox : checkBox4	Text	Banana
	Checked	False
Label : label1	Text	Apple



1. 버튼 기반 컨트롤

3) 체크 상자

(5) 예제

- 예제 : CheckBoxApp.cs

컨트롤 : (Name)	이벤트	메소드명
CheckBox : checkBox1	CheckedChange	checkBox1_Checked dChanged()
CheckBox : checkBox2	CheckedChange	checkBox2_Checked dChanged()
CheckBox : checkBox3	CheckedChange	checkBox3_Checked dChanged()
CheckBox : checkBox4	CheckedChange	checkBox4_Checked dChanged()

1. 버튼 기반 컨트롤

3) 체크 상자

(5) 예제

- 예제 : CheckBoxApp.cs

코드

```
private string strTemp;
private void UpdateLabel1(string s, bool b) {
    if (b) { label1.Text += s;
    } else {

strTemp = label1.Text; int i = strTemp.IndexOf(s.Substring(0, 1));
    int j = i + s.Length; label1.Text = strTemp.Remove(i, j - i);
    }
}
private void checkbox1_CheckedChanged(object sender, EventArgs e)
{
    UpdateLabel(checkboxBox1.Text, checkBox1.Checked);
}
private void checkbox2_CheckedChanged(object sender, EventArgs e)
{
    UpdateLabel(checkboxBox2.Text, checkBox2.Checked);
}
private void checkbox3_CheckedChanged(object sender, EventArgs e)
{
    UpdateLabel(checkboxBox3.Text, checkBox3.Checked);
}
private void checkbox4_CheckedChanged(object sender, EventArgs e)
{
    UpdateLabel(checkboxBox4.Text, checkBox4.Checked);
}
```

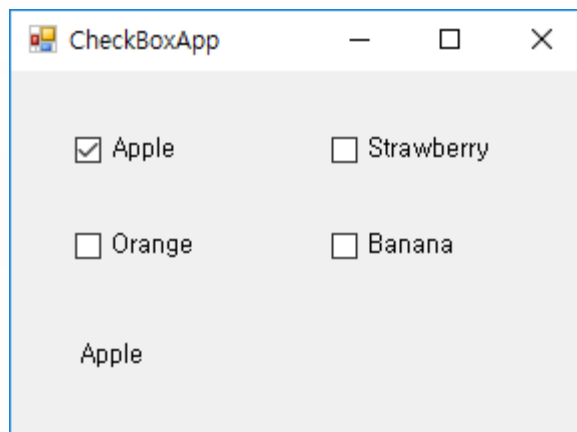
1. 버튼 기반 컨트롤

3) 체크 상자

(5) 예제

- 예제 : CheckBoxApp.cs

실행 결과



1. 버튼 기반 컨트롤

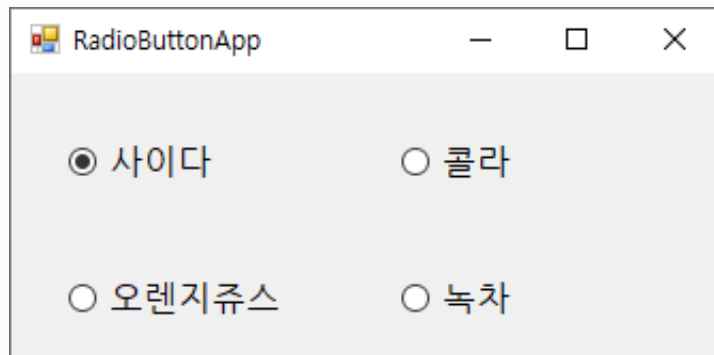
4) 라디오 버튼

(1) 개념

- 라디오 버튼의 개념
→ 주어진 항목들 중에서 오직 한개만을 선택할 수 있는 컨트롤

(2) 형태

- 라디오 버튼의 기본적인 형태



(3) 프로퍼티

- 라디오 버튼의 프로퍼티
→ 참으로 설정하면 동그란 부분에 '●'가 표시
→ Checked 프로퍼티의 값이 변경될 때마다 CheckedChanged 이벤트 발생

1. 버튼 기반 컨트롤

4) 라디오 버튼

(4) 예제

- 예제 : RadioButtonApp.cs

The screenshot shows a Windows Form titled "RadioButtonApp" with a light gray background. It contains four radio buttons arranged in a 2x2 grid. The top-left radio button is selected and is labeled "사이다". The top-right radio button is labeled "콜라". The bottom-left radio button is labeled "오렌지쥬스". The bottom-right radio button is labeled "녹차". Below the radio buttons, there is a label with the text "사이다". Orange callout boxes with lines pointing to the controls identify them as follows:

- RadioButton: radioButton1 (points to the selected "사이다" radio button)
- RadioButton: radioButton2 (points to the "콜라" radio button)
- RadioButton: radioButton3 (points to the "오렌지쥬스" radio button)
- RadioButton: radioButton4 (points to the "녹차" radio button)
- Label: label1 (points to the "사이다" label)
- Form: form1 (points to the form window)



1. 버튼 기반 컨트롤

4) 라디오 버튼

(4) 예제

- 예제 : RadioButtonApp.cs

컨트롤 : (Name)	프로퍼티	값
Form : Form1	Text	RadioButtonApp
RadioButton : radioButton1	Text	사이다
	Checked	True
RadioButton : radioButton2	Text	콜라
	Checked	False
RadioButton : radioButton3	Text	오렌지쥬스
	Checked	False
RadioButton : radioButton4	Text	녹차
	Checked	False
Label : label1	Text	사이다



1. 버튼 기반 컨트롤

4) 라디오 버튼

(4) 예제

- 예제 : RadioButtonApp.cs

컨트롤 : (Name)	이벤트	메소드명
RadioButton : radioButton1	CheckedChange	radioButton1_CheckedChanged()
RadioButton : radioButton2	CheckedChange	radioButton2_CheckedChanged()
RadioButton : radioButton3	CheckedChange	radioButton3_CheckedChanged()
RadioButton : radioButton4	CheckedChange	radioButton4_CheckedChanged()



1. 버튼 기반 컨트롤

4) 라디오 버튼

(4) 예제

- 예제 : RadioButtonApp.cs

코드

```
private void radioButton1_CheckedChanged(object sender, EventArgs e) {  
    label1.Text = radioButton1.Text;  
}  
private void radioButton2_CheckedChanged(object sender, EventArgs e) {  
    label1.Text = radioButton2.Text;  
}  
private void radioButton3_CheckedChanged(object sender, EventArgs e) {  
    label1.Text = radioButton3.Text;  
}  
private void radioButton4_CheckedChanged(object sender, EventArgs e) {  
    label1.Text = radioButton4.Text;  
}
```

실행 방법

라디오 버튼을 클릭하여 선택

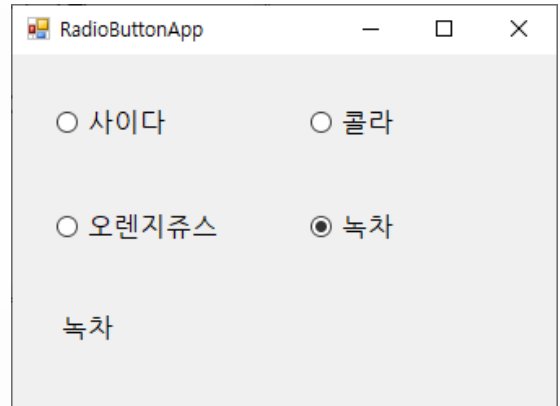
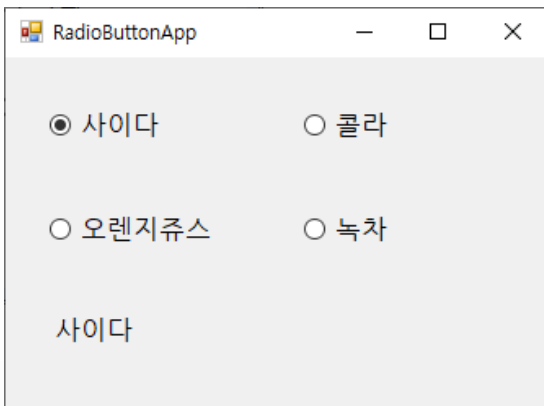
1. 버튼 기반 컨트롤

4) 라디오 버튼

(4) 예제

- 예제 : RadioButtonApp.cs

실행 결과



2. 레이블과 링크 레이블

1) 레이블

(1) 개념

- 레이블의 개념

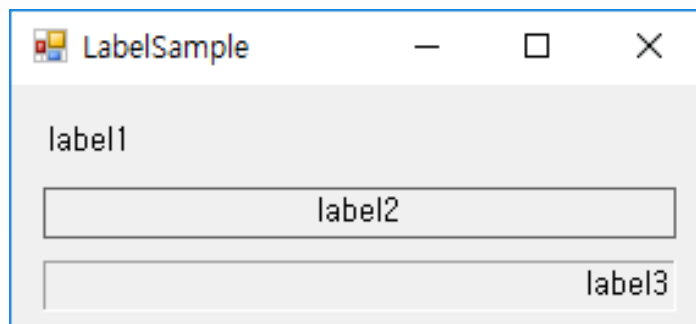
→ 각종 정보를 폼에 표시할 때 사용하는 컨트롤

→ 레이블의 용도

- 폼에 있는 컨트롤을 식별하는 정보를 표시
- 애플리케이션의 실행에 대한 응답 정보를 표시
- 특정 컨트롤을 클릭했을 때 실행되는 작업을 설명하는 메시지를 표시

(2) 형태

- 레이블의 기본적인 형태



(3) 프로퍼티

- 레이블의 프로퍼티

→ Font, BorderStyle, TextAlign

2. 레이블과 링크 레이블

1) 레이블

(3) 프로퍼티

- 레이블 프로퍼티 - BorderStyle

→ 레이블의 테두리를 설정할 때 사용하는 프로퍼티

→ BorderStyle 열거형

기호상수	순서 값	설명
None	0	테두리가 없음
FixedSingle	1	단일 선 테두리
Fixed3D	2	3차원 테두리

- 레이블 프로퍼티 - TextAlign

→ 레이블과 같은 컨트롤에서 표시되는 문자열에 대한 정렬을 위한 프로퍼티

→ System.Drawing 네임스페이스에 포함된

ContentAlignment 열거형을 사용하여 지정할 수 있음

- ContentAlignment 열거형



2. 레이블과 링크 레이블

1) 레이블

(3) 프로퍼티

- 레이블 프로퍼티 - TextAlign

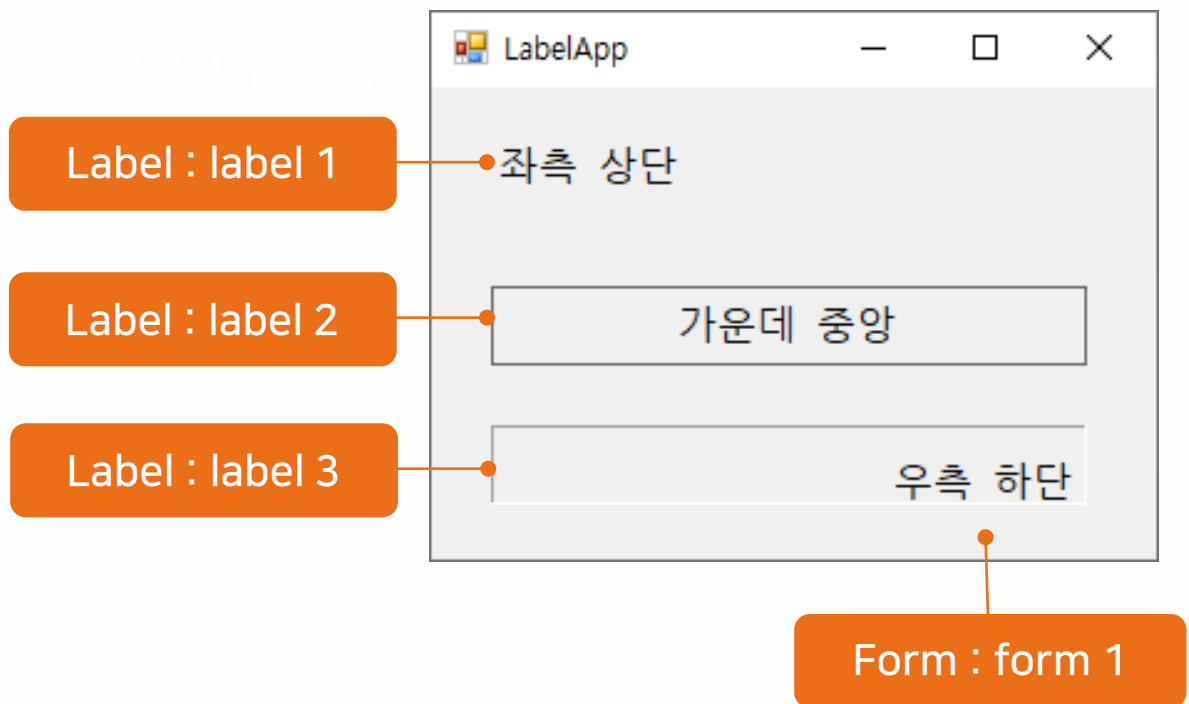
기호상수	순서 값	설명
TopLeft	0x0001	좌측 상단 정렬
TopCneter	0x0002	가운데 상단 정렬
TopRight	0x0004	우측 상단 정렬
MiddleLeft	0x0010	좌측 중앙 정렬
MiddleCenter	0x0020	가운데 중앙 정렬
MiddleRight	0x0040	우측 중앙 정렬
BottomLeft	0x0100	좌측 하단 정렬
BottomCenter	0x0200	가운데 하단 정렬
BottomRight	0x0400	우측 중앙 정렬
TopLeft	0x0001	좌측 상단 정렬
TopCneter	0x0002	가운데 상단 정렬
TopRight	0x0004	우측 상단 정렬

2. 레이블과 링크 레이블

1) 레이블

(4) 예제

▪ LabelApp.cs





2. 레이블과 링크 레이블

1) 레이블

(4) 예제

▪ LabelApp.cs

컨트롤 : (Name)	프로퍼티	값
Form : Form1	Text	LabelApp
Label : label1	Text	좌측 상단
	BorderStyle	None
	TextAlign	TopLeft
Label : label2	Text	가운데 중앙
	BorderStyle	FixedSingle
	TextAlign	MiddleCenter
Label : label3	Text	우측 하단
	BorderStyle	Fixed3D
	TextAlign	BottomRight

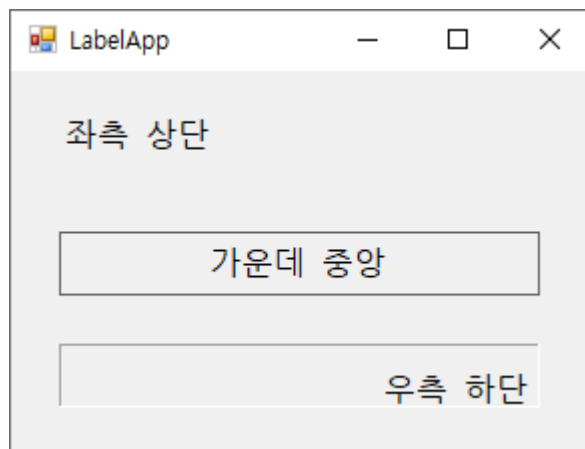
2. 레이블과 링크 레이블

1) 레이블

(4) 예제

- LabelApp.cs

실행 결과



2. 레이블과 링크 레이블

2) 링크 레이블

(1) 개념

- 링크 레이블의 개념

- 하이퍼링크(Hyper Link)를 설정할 수 있는 레이블 컨트롤
- 레이블 컨트롤을 상속받았기 때문에 레이블의 기본적인 기능을 모두 보유
- 링크를 클릭 했을 때 발생하는 LinkClicked 이벤트에 대한 이벤트 처리기를 작성
 - 이벤트를 처리하기 위해서는 Process 클래스의 정적 메소드인 Start() 메소드를 사용

(2) 형태

- 링크 레이블의 기본적인 형태





2. 레이블과 링크 레이블

2) 링크 레이블

(3) 예제

- 예제 : LinkLabelApp.cs

LinkLabel : Linklabel 1

LinkLabel : Linklabel 1

LinkLabel : Linklabel 1

http://www.abcdefg.com

mailto:nobody@example.com

c:\Wlog.txt

Form : form 1

컨트롤 : (Name)	프로퍼티	값
Form : Form1	Text	LinkLabelApp
LinkLabel : linkLabel1	Text	http://www.example.com
LinkLabel : linkLabel2	Text	mailto:nobody@example.com
LinkLabel : linkLabel3	Text	C:\Wlog.txt

2. 레이블과 링크 레이블

2) 링크 레이블

(3) 예제

- 예제 : LinkLabelApp.cs

컨트롤 : (Name)	이벤트	메소드명
LinkLabel : linkLabel1	LinkClicked	linkLabel1_LinkClicked()
LinkLabel : linkLabel2	LinkClicked	linkLabel2_LinkClicked()
LinkLabel : linkLabel3	LinkClicked	linkLabel3_LinkClicked()

코드

```
private void linkLabel1_LinkClicked
(object sender, LinkLabelLinkClickedEventArgs e)
{
    Process.Start(linkLabel1.Text);
}
private void linkLabel2_LinkClicked
(object sender, LinkLabelLinkClickedEventArgs e)
{
    Process.Start(linkLabel2.Text);
}
private void linkLabel3_LinkClicked
(object sender, LinkLabelLinkClickedEventArgs e)
{
    Process.Start(linkLabel3.Text);
}
```

2. 레이블과 링크 레이블

2) 링크 레이블

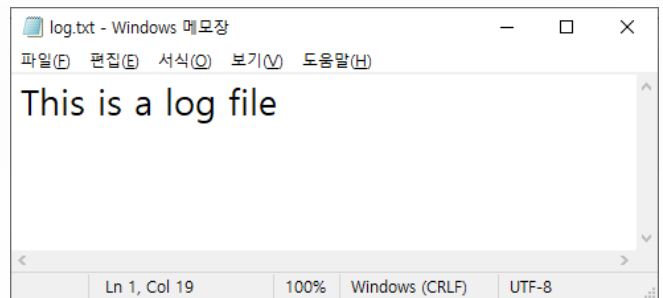
(3) 예제

- 예제 : LinkLabelApp.cs

실행 방법

각각의 링크를 클릭

실행 결과



3. 텍스트와 리스트 상자

1) 텍스트 상자

(1) 개념

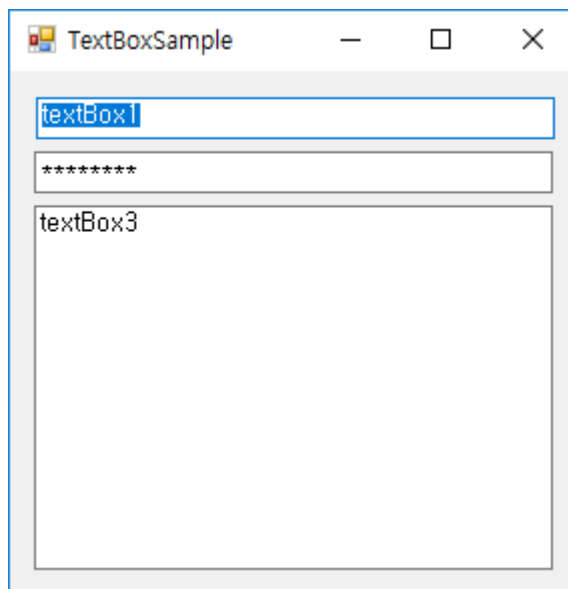
- 텍스트 상자의 개념

→ 사용자가 직접 텍스트를 입력할 수 있는 기본적인 컨트롤로서
사용자로부터 값을 입력 받을 때 사용

→ 애플리케이션의 실행 결과를 출력하고자 할 때 유용

(2) 형태

- 텍스트 상자의 기본적인 형태





3. 텍스트와 리스트 상자

1) 텍스트 상자

(3) 프로퍼티

- 텍스트 상자의 프로퍼티

프로퍼티	설명
MaxLength	텍스트 상자에 입력할 수 있는 최대문자수를 설정
Multiline	텍스트 상자의 영역을 여러 줄로 설정
PasswordChar	텍스트 상자의 암호 입력에 사용할 문자를 설정
ReadOnly	텍스트 상자의 텍스트를 변경하지 못하도록 설정
WordWrap	텍스트 상자의 텍스트가 영역을 초과할 경우 자동으로 줄을 바꾸도록 설정
ScrollBars	Multiline 프로퍼티가 참인 경우 텍스트 상자에 스크롤 바를 설정(None Horizontal Vertical Both)

3. 텍스트와 리스트 상자

2) 리스트 상자

(1) 개념

- 리스트 기반 컨트롤
→ ListControl 클래스를 상속받은 컨트롤을 가리킴

(2) 종류

- 리스트 기반 컨트롤의 종류

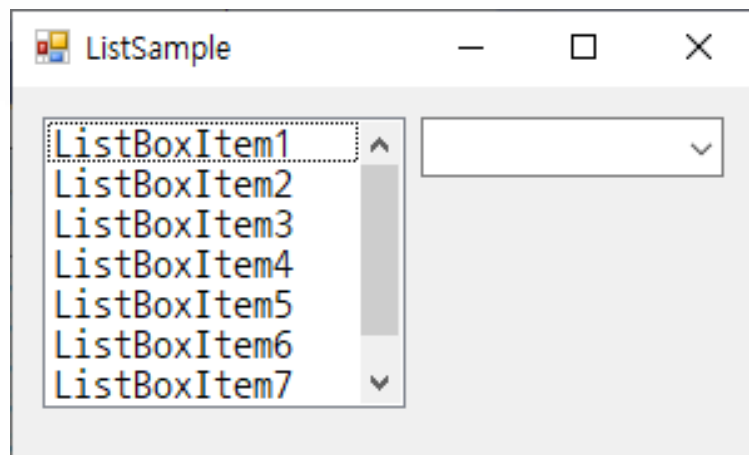
리스트 상자

콤보 상자

체크리스트 상자

(3) 형태

- 리스트 기반 컨트롤의 기본적인 형태





3. 텍스트와 리스트 상자

2) 리스트 상자

(4) 개념

- 리스트상자의개념
→ 사용자가 선택할 수 있는 항목들의 목록을 표시해 주는 컨트롤

(5) 프로퍼티

- 리스트 상자의 프로퍼티

프로퍼티	설명
Item	리스트 상자의 항목을 설정
MultiColumn	리스트 상자가 여러 열을 표시할 수 있도록 설정
ScrollAlwatsVisible	항상 세로 스크롤바가 표시되도록 설정
SelectionMode	리스트 상자에서 항목을 선택하는 방법을 설정
SelectionIndex	리스트 상자에서 현재 선택된 아이템이 인덱스를 반환 (0부터 시작)
SelectedItem	리스트 상자에서 현재 선택된 아이템을 반환

3. 텍스트와 리스트 상자

2) 리스트 상자

(6) 항목 선택

- 리스트 상자의 항목 선택 방법
 - SelectionMode 프로퍼티 값에 따라 변경
 - SelectionMode 열거형

기호상수	설명
None	항목을 선택할 수 없음
One	하나의 항목만 선택할 수 있음
MultiSimple	여러 항목을 선택할 수 있음
MultiExtended	여러 항목을 선택할 수 있으며, <Shift>, <Ctrl>, 마우스 포인터를 이용해 선택할 수 있음

(7) 항목

- 리스트 상자의 항목
 - Items 프로퍼티를 통하여 추가 및 삭제 가능
 - 문자열 컬렉션 편집기
 - ObjectCollection 클래스 메소드



3. 텍스트와 리스트 상자

2) 리스트 상자

(8) 메소드

- ObjectCollection 클래스의 메소드

메소드	설명
Add(object item)	리스트 상자에 항목을 추가
Clear()	리스트 상자의 모든 항목을 제거
FindString(string s, int index)	리스트 상자의 항목 중 지정된 인덱스 다음부터 지정된 문자열로 시작하는 항목의 인덱스를 반환
IndexOf(object item)	지정한 항목의 인덱스를 반환
Insert(int index, object item)	지정된 인덱스에 항목을 추가
Remove(object item)	지정된 항목을 제거
RemoveAt(int index)	지정된 인덱스의 항목을 제거

3. 텍스트와 리스트 상자

3) 콤보 상자

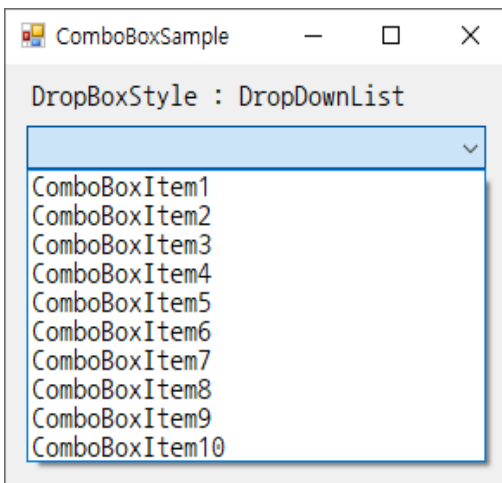
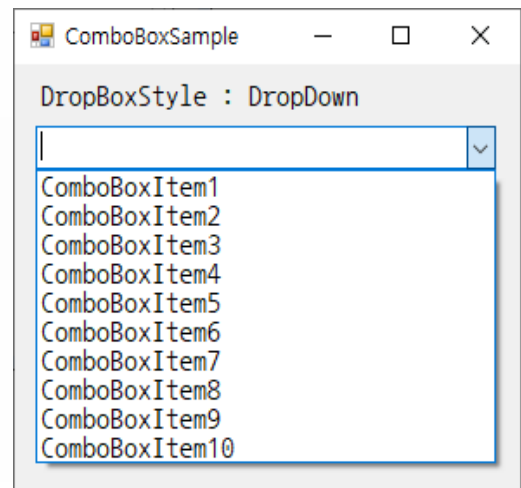
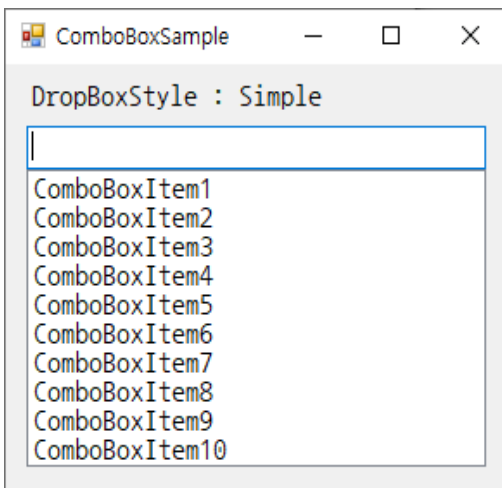
(1) 개념

- 콤보 상자의 개념

→ 사용자가 상자를 클릭하면 목록이 나타나는 드롭다운(drop-down) 형식의 컨트롤

(2) 형태

- 콤보 상자의 기본적인 형태





3. 텍스트와 리스트 상자

3) 콤보 상자

(3) 정의

- DropDownStyle의 정의
 - 콤보 상자의 형태를 결정
 - ComboBoxStyle을 열거형으로 가짐
 - ComboBoxStyle 열거형

기호상수	설명
Simple	선택 항목을 항상 볼 수 있음
DropDown	화살표 버튼을 클릭해야 선택항목을 볼 수 있음
DropDownList	화살표 버튼뿐만 아니라 텍스트 부분을 클릭하여도 선택 항목을 볼 수 있음

(4) 항목

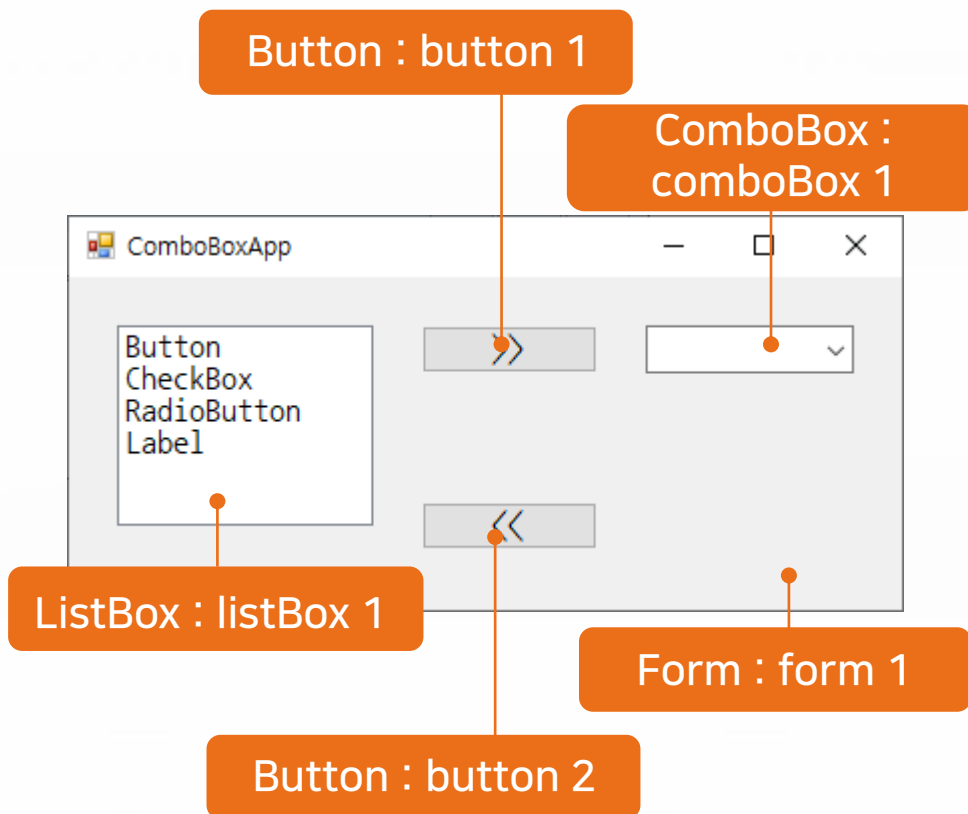
- 콤보 상자의 항목
 - Items 프로퍼티를 통하여 추가 및 삭제 가능
 - 문자열 컬렉션 편집기
 - ObjectCollect 클래스 메소드

3. 텍스트와 리스트 상자

3) 콤보 상자

(5) 예제

- 예제 : ComboBoxApp.cs





3. 텍스트와 리스트 상자

3) 콤보 상자

(5) 예제

- 예제 : ComboBoxApp.cs

컨트롤 : (Name)	프로퍼티	값
Form : Form1	Text	ComboBoxApp
ListBox : listBox1	Items	Button CheckBox RadioButton Lable
ComboBox : comboBox1	Items	LinkLabel TextBox ListBox ComboBox
Button : button1	Text	>>
Button : button2	Text	<<

컨트롤 : (Name)	이벤트	메소드명
Button : button1	Click	Button1_Click()
Button : button2	Click	Button2_Click()

3. 텍스트와 리스트 상자

3) 콤보 상자

(5) 예제

- 예제 : ComboBoxApp.cs

코드

```
private void button1_Click(object sender, EventArgs e) {  
    if (listBox1.SelectedMode != null) {  
        comboBox1.Items.Add(listBox1.SelectedItem);  
        listBox1.Item.Remove(listBox1.SelectedItem) :  
    }  
}  
private void button2_Click(object sender, EventArgs e) {  
    if (comboBox1.SelectedMode != null) {  
        listBox1.Items.Add(comboBox1.SelectedItem);  
        comboBox1.Item.Remove(comboBox1.SelectedItem) :  
    }  
}
```

3. 텍스트와 리스트 상자

3) 콤보 상자

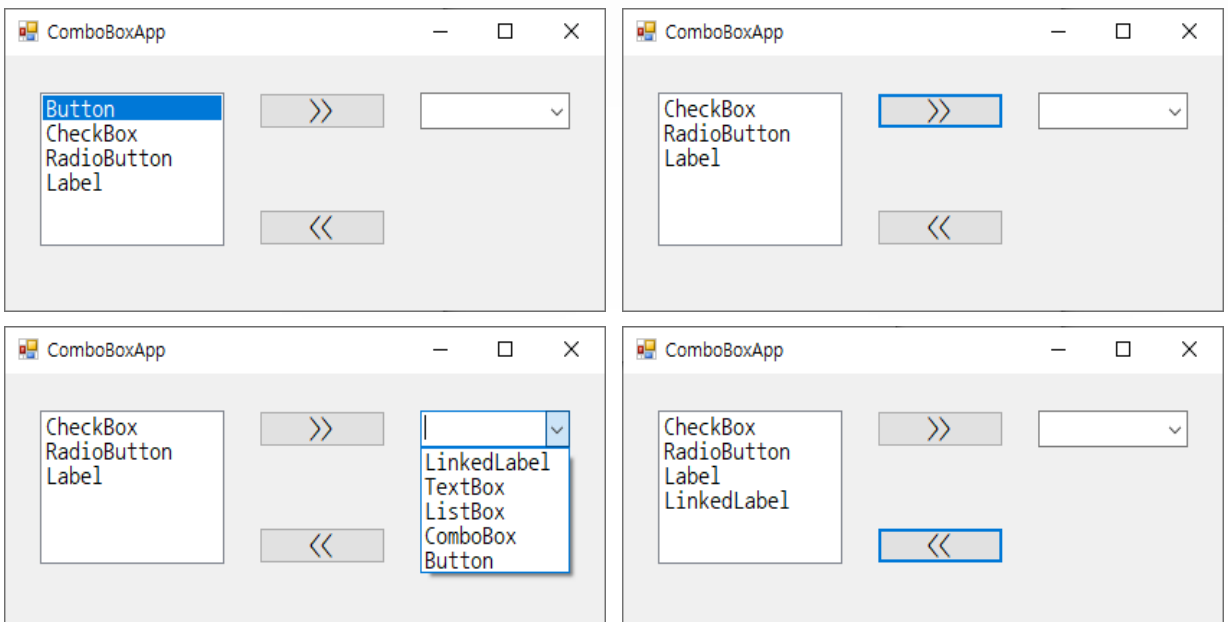
(5) 예제

- 예제 : ComboBoxApp.cs

실행 방법

- ① 리스트 상자의 항목을 선택한 후, >> 버튼을 클릭
- ② 콤보 상자의 항목을 선택한 후, << 버튼을 클릭

실행 결과



3. 텍스트와 리스트 상자

4) 체크리스트

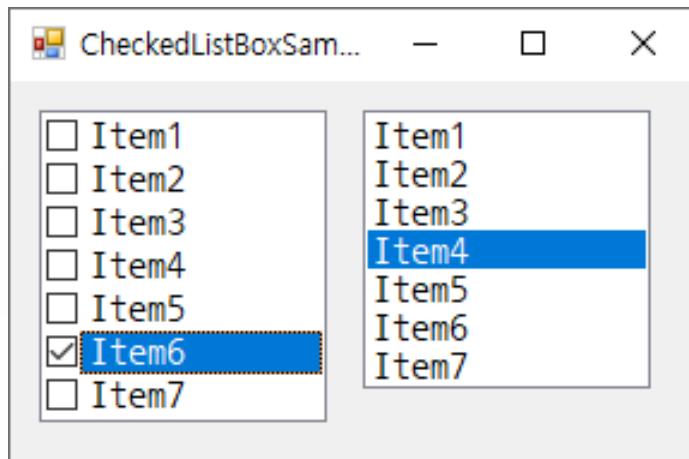
(1) 개념

- 체크 리스트의 개념

→ 리스트 상자의 항목에 체크 상자를 추가한 형태의 컨트롤

(2) 형태

- 체크 리스트 상자의 기본적인 형태



(3) 프로퍼티

- 체크 리스트 상자의 프로퍼티

→ 리스트 상자와 대부분 동일

→ CheckOnClick 프로퍼티가 추가

- 항목을 클릭했을 때 체크 상자에 '✓' 표시가 나타나도록 설정

3. 텍스트와 리스트 상자

4) 체크리스트

(4) 예제

- 예제 : CheckedExceptionApp.cs

CheckedListBoxApp

취미를 선택하시오

<input type="checkbox"/> 영화감상	<input type="checkbox"/> 스키
<input type="checkbox"/> 음악감상	<input type="checkbox"/> 스노우보드
<input type="checkbox"/> 축구	<input type="checkbox"/> 수영
<input type="checkbox"/> 농구	<input type="checkbox"/> 장기
<input type="checkbox"/> 골프	<input type="checkbox"/> 바둑

선택

Label : label 1

CheckedListBox : checkedListBox 1

Form : form 1

Button : button 2



3. 텍스트와 리스트 상자

4) 체크리스트

(4) 예제

- 예제 : CheckedExceptionApp.cs

컨트롤 : (Name)	프로퍼티	값	
Form : Form1	Text	CheckedListBoxApp	
Label : label1	Items	취미를 선택하십시오	
CheckedListBox : checkedListBox1	Items	영화감상 음악감상 축구 농구 골프	스키 스노우보드 수영 장기 바둑
	Multi Column	True	
Button : button1	Text	선택	

컨트롤 : (Name)	이벤트	메소드명
Button : button1	Click	Button1_Click()

3. 텍스트와 리스트 상자

4) 체크리스트

(4) 예제

- 예제 : CheckBoxApp.cs

코드

```
private void button1_Click(object sender, EventArgs e)
{
    string strTemp = "";
    foreach(object obj in checkedListBox1.CheckedItems) {
        strTemp += obj.ToString();
        strTemp += " " :
    }
    MessageBox.Show("당신의 취미는" + strTemp + "입니다.");
}
```

3. 텍스트와 리스트 상자

4) 체크리스트

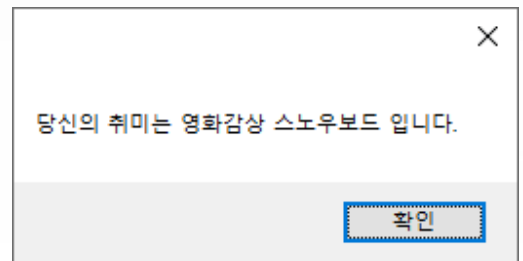
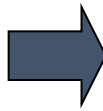
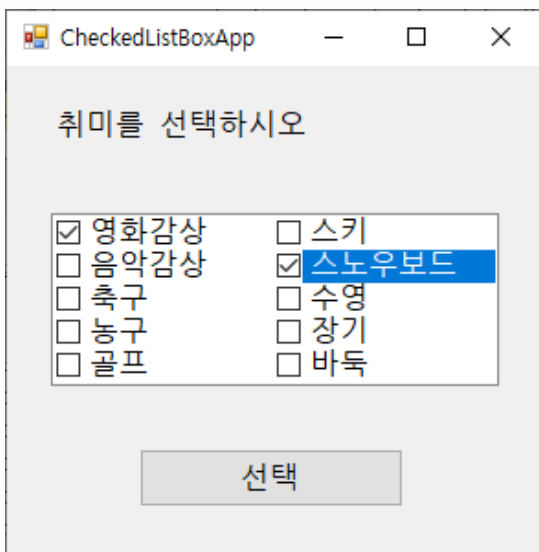
(4) 예제

- 예제 : CheckBoxApp.cs

실행 방법

체크리스트상자의 항목을 선택한 후, 선택 버튼을 클릭

실행 결과





1. 버튼 기반 컨트롤

- 컨트롤 : 화면에 표시되어 사용자와 상호작용을 수행하는 컴포넌트
- 버튼 기반 컨트롤 : ButtonBase 추상 클래스를 상속받은 컨트롤을 의미
- 버튼 기반 컨트롤의 종류에는 버튼, 체크상자, 라디오 버튼 등이 있음

2. 레이블과 링크 레이블

- 레이블
 - 각종 정보를 폼에 표시할 때 사용하는 컨트롤
 - 폼에 있는 컨트롤을 식별하는 정보를 표시
 - 애플리케이션의 실행에 대한 응답 정보를 표시
 - 특정 컨트롤을 클릭했을 때 실행되는 작업을 설명하는 메시지를 표시



2. 레이블과 링크 레이블

- 링크 레이블
 - 하이퍼링크를 설정할 수 있는 레이블 컨트롤
 - 레이블 컨트롤을 상속받았기 때문에 레이블의 기본적인 기능을 모두 보유
 - 링크를 클릭했을 때 발생하는 LinkClicked 이벤트에 대한 이벤트 처리기를 작성

3. 텍스트와 리스트 상자

- 텍스트 상자 : 사용자가 직접 텍스트를 입력할 수 있는 기본적인 컨트롤
- 리스트 기반 컨트롤 : ListControl 클래스를 상속받은 컨트롤
- 콤보 상자 : 사용자가 상자를 클릭하면 목록이 나타나는 드롭다운(drop-down) 형식의 컨트롤
- 체크 리스트 상자 : 리스트 상자의 항목에 체크 상자를 추가한 형태의 컨트롤