

C# 프로그래밍 입문



11. 고급 컨트롤



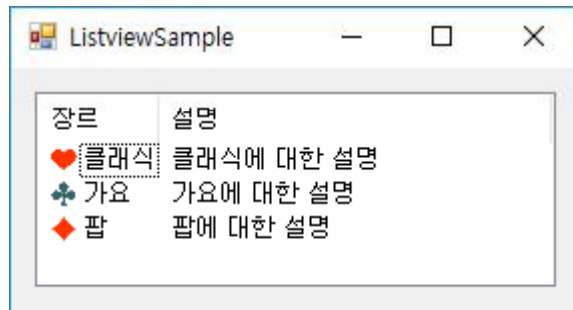
목차

- 리스트 뷰
- 트리 뷰
- 업다운 컨트롤
- 트랙 바
- 프로그레스 바
- 타이머 컴포넌트



리스트 뷰

- 리스트 상자와 유사한 형태를 지니며 목록을 구조적으로 장식할 수 있는 컨트롤
 - 리스트 상자 + 추가적인 정보 (아이콘, 설명)

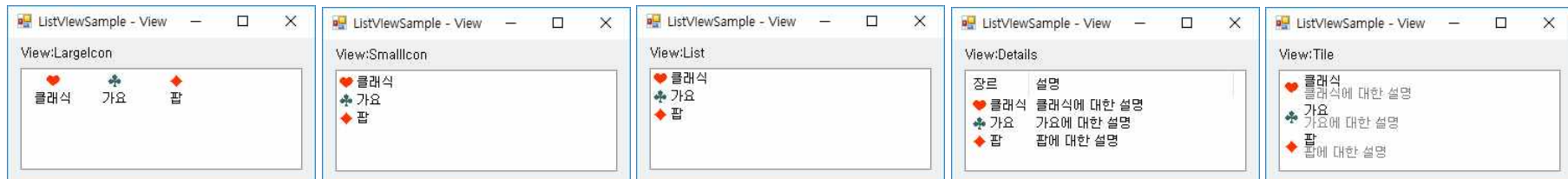




리스트 뷰의 형태

- View 프로퍼티의 값에 따라 다양한 형태를 가짐
 - System.Windows.Forms 네임스페이스에 포함된 View열거형을 값으로 가짐
 - View 열거형

기호상수	설명
Largelcon	큰 아이콘의 형태 (1)
SmallIcon	작은 아이콘의 형태 (2)
List	간단한 리스트 형태 (3)
Detail	자세한 리스트 형태 (4)
Tile	큰 아이콘이 표시되는 자세한 리스트 형태 (5)





리스트 뷰 항목의 선택 [1/6]

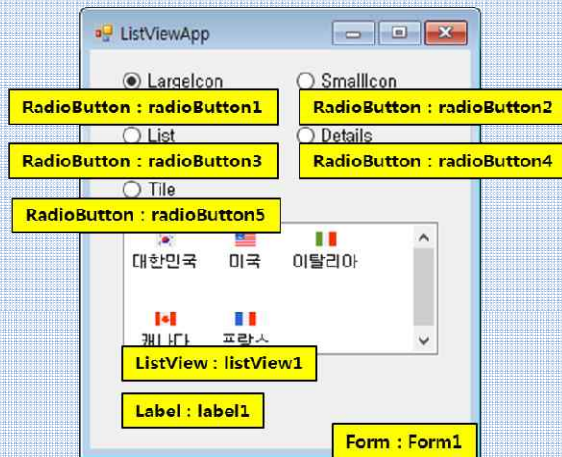
- SelectedItems 프로퍼티
 - 리스트 뷰에서 선택된 항목을 저장하는 프로퍼티
 - 반환형
 - ListViewItem 클래스형
 - 리스트 뷰의 MultiSelect 프로퍼티가 거짓일 경우
 - ListViewItem 클래스의 배열형
 - 리스트 뷰의 MultiSelect 프로퍼티가 참일 경우



리스트 뷰 항목의 선택 [2/6]

[예제 11.1 – ListViewApp.cs]

1) 디자인



컴포넌트 : (Name)	프로퍼티	인덱스	값
ImageList : imageList1	Images	0	South Korea.png
		1	USA.png
		2	Italy.png
		3	Canada.png
		4	France.png

이미지 경로 : <https://icons8.com/web-app/category/all/Flags>



리스트 뷰 항목의 선택 [3/6]

컨트롤 : (Name)	프로퍼티	값
Form : Form1	Text	ListViewApp
RadioButton : radiobutton1	Text	LargeIcon
	Checked	True
RadioButton : radioButton2	Text	SmallIcon
RadioButton : radioButton3	Text	List
RadioButton : radioButton4	Text	Details
RadioButton : radioButton5	Text	Details
Label : label1	Text	
ListView : listView1	Columns	columnHeader1
		columnHeader2
	LargeImageList	imageList1
	SmallImageList	imageList1
ColumnHeader : columnHeader1	Text	국가
ColumnHeader : columnHeader2	Text	국가번호



리스트 뷰 항목의 선택 [4/6]

listView1의 Items 프로퍼티

Items	프로퍼티	값	프로퍼티	값
ListViewItem0	ImageIndex	0		
	SubItems	ListViewSubItem0	Text	대한민국
		ListViewSubItem1	Text	82
ListViewItem1	ImageIndex	1		
	SubItems	ListViewSubItem0	Text	미국
		ListViewSubItem1	Text	1
ListViewItem2	ImageIndex	2		
	SubItems	ListViewSubItem0	Text	이탈리아
		ListViewSubItem1	Text	39
ListViewItem3	ImageIndex	3		
	SubItems	ListViewSubItem0	Text	캐나다
		ListViewSubItem1	Text	1
ListViewItem4	ImageIndex	4		
	SubItems	ListViewSubItem0	Text	프랑스
		ListViewSubItem1	Text	33



리스트 뷰 항목의 선택 [5/6]

컨트롤 : (Name)	이벤트	메소드명
RadioButton : radioButton1	CheckedChanged	radioButton1_CheckedChanged
RadioButton : radioButton2	CheckedChanged	radioButton2_CheckedChanged
RadioButton : radioButton3	CheckedChanged	radioButton3_CheckedChanged
RadioButton : radioButton4	CheckedChanged	radioButton4_CheckedChanged
RadioButton : radioButton5	CheckedChanged	radioButton5_CheckedChanged
ListView : listView1	Click	listView1_Click()

2) 코드

```
private void radioButton1_CheckedChanged(object sender, EventArgs e) {
    if (radioButton1.Checked)
        // 리스트 뷰의 항목을 큰 아이콘 형태로 보여준다.
        listView1.View = View.LargeIcon;
}
private void radioButton2_CheckedChanged(object sender, EventArgs e) {
    if (radioButton2.Checked)
        // 리스트 뷰의 항목을 작은 아이콘 형태로 보여준다.
        listView1.View = View.SmallIcon;
}
private void radioButton3_CheckedChanged(object sender, EventArgs e) {
    if (radioButton3.Checked)
        // 리스트 뷰의 항목을 간단한 리스트 형태로 보여준다.
        listView1.View = View.List;
}
```



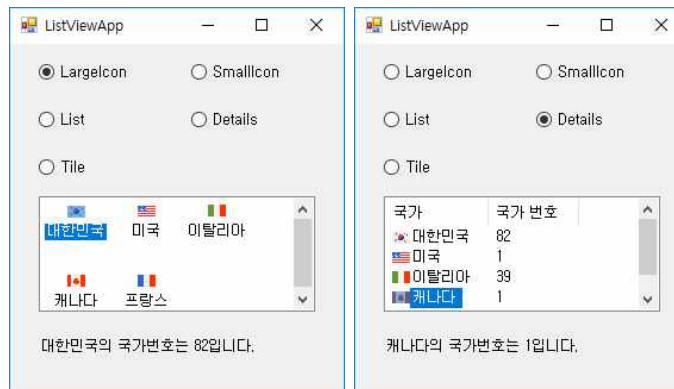
리스트 뷰 항목의 선택 [6/6]

```
private void radioButton4_CheckedChanged(object sender, EventArgs e) {
    if (radioButton4.Checked)
        // 리스트 뷰의 항목을 자세한 리스트 형태로 보여준다.
        listView1.View = View.Details;
}
private void radioButton5_CheckedChanged(object sender, EventArgs e) {
    if (radioButton5.Checked)
        // 리스트 뷰의 항목을 타일 형태로 보여준다.
        listView1.View = View.Details;
}
private void listView1_Click(object sender, EventArgs e) {
    foreach (ListViewItem item in listView1.SelectedItems) {
        ListViewItem.ListViewSubItemCollection subItem = item.SubItems;
        // 각 항목에 대한 부항목을 얻기 위해 SubItems 프로퍼티를 사용
        label1.Text = subItem[0].Text + "의 국가번호는 " + subItem[1].Text + "입니다.";
    }
}
```

실행 방법 : ① 라디오 버튼 중 하나를 선택한다.

② 리스트 뷰의 항목을 선택한다.

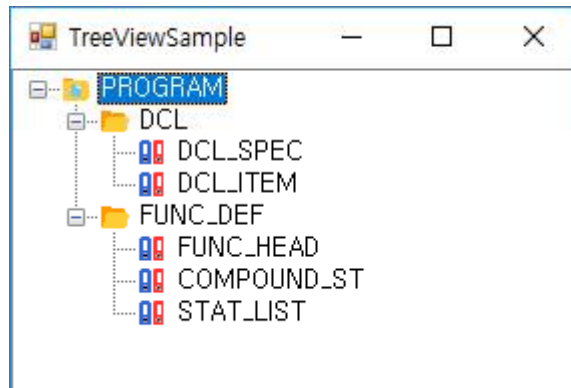
실행 결과 :





트리 뷰

- 목록을 계층적으로 보여주기 위한 컨트롤
 - 노드를 계층적으로 표시
 - 노드에 이미지 아이콘을 추가할 수 있음





트리 뷰의 노드

- 트리 노드 편집기를 통해 생성
- TreeNode 클래스의 객체
- TreeView 컨트롤의 Nodes 프로퍼티에 TreeNodeCollection 형으로 저장
- TreeNodeCollection 클래스의 메소드를 통해 노드의 편집이 가능함
 - TreeNodeCollection 클래스의 메소드

메소드	설명
Add(TreeNode node)	트리 뷰에 새로운 노드를 추가
Clear()	트리 뷰의 모든 노드를 삭제
Insert(int index, TreeNode node)	트리 뷰의 지정된 인덱스에 노드를 삽입
Remove(TreeNode node)	트리 뷰의 노드 중 매개 변수에 해당하는 노드를 삭제



TreeNode 클래스 [1/4]

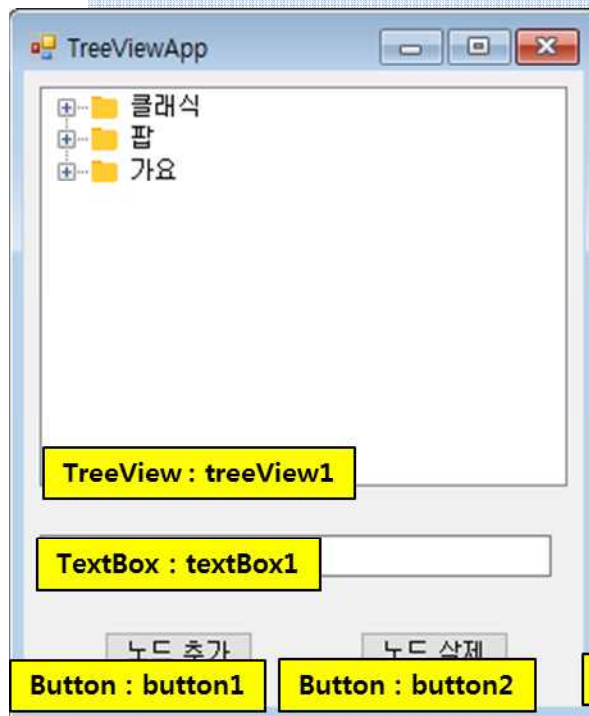
- 트리 뷰의 노드는 TreeNode 클래스의 객체
 - TreeNodeCollection 클래스의 메소드를 사용할 경우
 - TreeNode 클래스의 객체를 생성해야 함
 - TreeNode 클래스의 생성자
 - public TreeNode(string label);
 - public TreeNode(string label, int idx1, int idx2);
 - label : 노드 이름에 해당하는 문자열
 - idx1 : 노드가 선택되지 않았을 때의 이미지 인덱스
 - idx2 : 노드가 선택되었을 때의 이미지 인덱스



TreeNode 클래스 [2/4]

[예제 11.2 – TreeViewApp]

1) 디자인



컴포넌트 : (Name)	프로퍼티	인덱스	값
ImageList : imageList1	Images	0	Folder.png
		1	CD.png

이미지 경로 : <https://icons8.com/>

컨트롤 : (Name)	프로퍼티	값
Form : Form1	Text	TreeViewApp
TextBox : textBox1	Text	
Button : button1	Text	노드 추가
Button : button2	Text	노드 삭제
TreeView : treeView1	Nodes	



TreeNode 클래스 [3/4]

* treeView1의 Nodes

Node의 레이블	프로퍼티	이미지(인덱스)	선택한 이미지(인덱스)
클래식		0(Folder)	0(Folder)
	베토벤	1(CD)	1(CD)
	슈베르트	1(CD)	1(CD)
	모짜르트	1(CD)	1(CD)
팝		0(Folder)	0(Folder)
	Britney Spears	1(CD)	1(CD)
	Mariah Carey	1(CD)	1(CD)
	Capenters	1(CD)	1(CD)
가요		0(Folder)	0(Folder)
	이승환	1(CD)	1(CD)
	전인권	1(CD)	1(CD)
	이효리	1(CD)	1(CD)

컨트롤 : (Name)	이벤트	메소드명
Form : Form1	Load	Form1_Load()
Button : button1	Click	button1_Click()
Button : button2	Click	button2_Click()

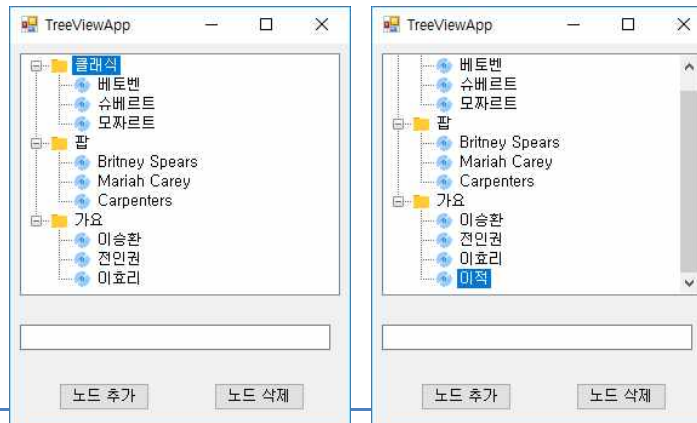


TreeNode 클래스 [4/4]

```
private void Form1_Load(object sender, EventArgs e) {
    treeView1.ExpandAll(); // 트리 뷰의 모든 노드를 펼침.
}
private void button1_CheckedChanged(object sender, EventArgs e) {
    if (textBox1.Text != "" && treeView1.SelectedNode != null) {
        // 선택된 노드가 있으면, 그 노드의 자식 노드로 추가한다.
        treeView1.SelectedNode.Nodes.Add(new TreeNode(textBox1.Text, 1, 1));
        textBox1.Text = "";
        textBox1.Focus();
    }
}
private void button2_Click(object sender, EventArgs e) {
    treeView1.Nodes.Remove(treeView1.SelectedNode);
}
```

- 실행 방법 : ① 자식 노드를 추가할 노드를 선택한다.
 ② 텍스트 상자에 텍스트를 입력한다.
 ③ 노드 추가 버튼을 클릭한다.

실행 결과 :





트리 뷰 항목의 선택

■ SelectedNode 프로퍼티

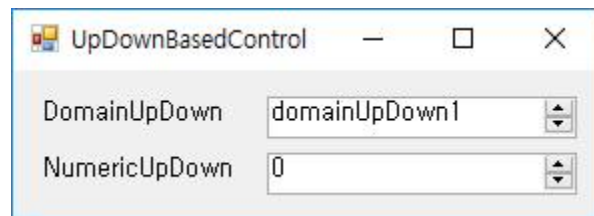
- 트리 뷰에서 선택된 노드를 저장하는 프로퍼티
- 반환형
 - TreeNode 클래스형
- TreeNode 클래스의 프로퍼티를 이용하면 선택된 노드를 기준으로 부모, 이전 형제, 다음 형제, 자식 노드를 참조할 수 있음
 - TreeNode 클래스의 프로퍼티 (p.486, 예제 11.3 참조)

메소드	설 명
Parent	현재 트리 노드의 부모 노드
PrevNode	현재 트리 노드의 이전 형제 노드
NextNode	현재 트리 노드의 다음 형제 노드
Nodes	현재 트리 노드의 자식 노드들



업다운 컨트롤

- 주어진 목록에서 항목을 선택할 수 있는 컨트롤
 - 업다운 버튼을 이용하여 필요한 값을 선택
 - 스피ن 컨트롤(spin control)
- 영역 업다운 컨트롤
 - 문자열로 이루어진 항목에서 특정한 항목을 선택할 수 있는 컨트롤
- 수치적 업다운 컨트롤
 - 지정한 범위 내에서 수치적 값을 선택할 수 있는 컨트롤

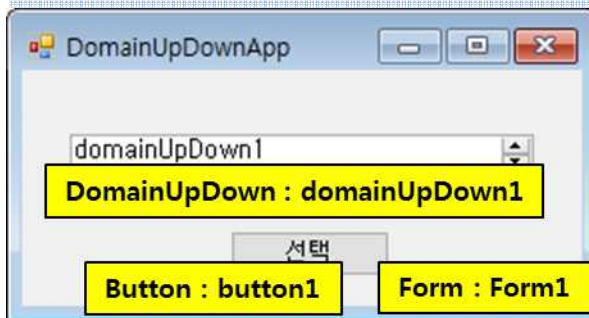




영역 업다운 컨트롤의 작성

[예제 11.4 – DomainUpDownApp]

1) 디자인



컨트롤 : (Name)	프로퍼티	값
Form : Form1	Text	DomainUpDownApp
Button : button1	Text	선택
DomainUpDown : domainUpDown1	Items	프로그래밍언어 컴파일러 컴퓨터구성 알고리즘 데이터베이스 운영체제
	Wrap	True

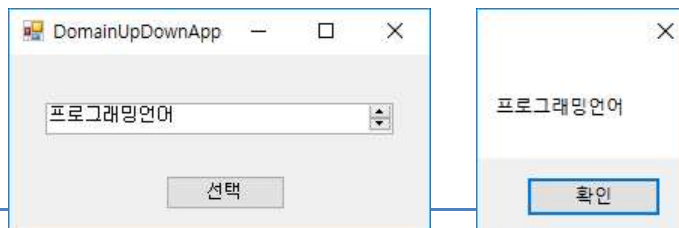
컨트롤 : (Name)	이벤트	메소드명
Button : button1	Click	Button1_Click()

2) 코드

```
private void button1_Click(object sender, EventArgs e) {
    MessageBox.Show(domainUpDown1.SelectedItem.ToString());
}
```

실행 방법 : 도메인 업다운 컨트롤에서 항목을 선택한 후, 선택 버튼을 클릭한다.

실행 결과 :





수치적 업다운 컨트롤의 작성 [1/3]

- 수치적 업다운 컨트롤의 추가
 - 【도구상자】∅【NumericUpDown】을 선택하여 폼에 추가
- 수치적 업다운 컨트롤의 항목에 대한 범위와 증가/감소량을 설정
 - 수치적 업다운 컨트롤의 프로퍼티를 통해 설정

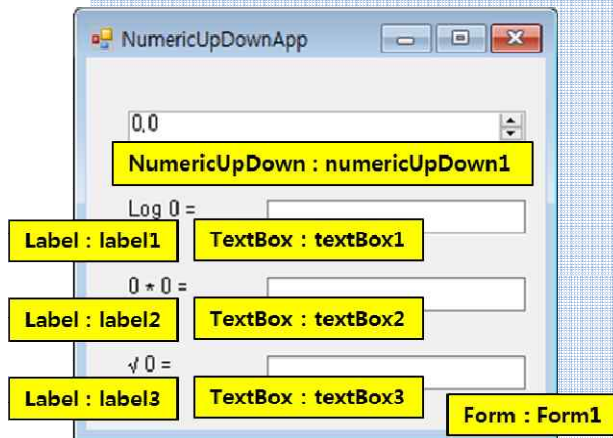
프로퍼티	설 명
Minimum	수치적 업다운 컨트롤의 최소 값.
Maximum	수치적 업다운 컨트롤의 최대 값.
Increment	수치적 업다운 컨트롤의 증가/감소 양.
Value	수치적 업다운 컨트롤의 현재 값.
DecimalPlaces	수치적 업다운 컨트롤에 표시할 소수 자릿수.
ThousandsSeparator	10진수 3자리마다 구분 기호를 삽입 여부.
Hexadecimal	수치적 업다운 컨트롤의 값을 16진수로 표시.



수치적 업다운 컨트롤의 작성 [2/3]

[예제 11.5 – NumericUpDownApp]

1) 폼 설계



2) 프로퍼티

컨트롤 : (Name)	프로퍼티	값
Form : Form1	Text	NumericUpDownApp
Label : label1	Text	Log 0 =
Label : label2	Text	0 * 0 =
Label : label3	Text	√ 0 =
TextBox : textBox1	Text	
TextBox : textBox2	Text	
TextBox : textBox3	Text	
NumericUpDown : numericUpDown1	Minimum	0
	Maximum	1000
	Increment	0.5
	DecimalPlaces	1

컨트롤 : (Name)	이벤트	메소드명
NumericUpDown : numericUpDown1	ValueChanged	numericUpDown1_ValueChanged()



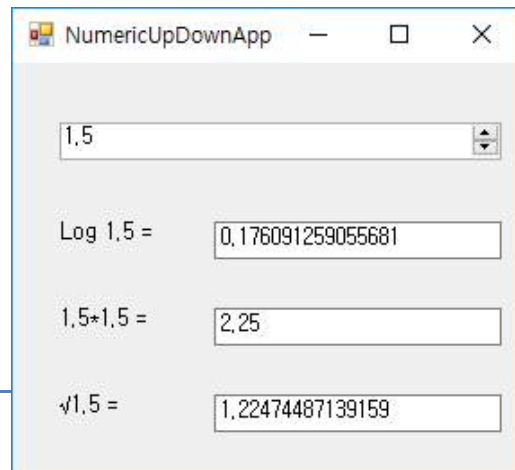
수치적 업다운 컨트롤의 작성 [3/3]

2) 코드

```
private void numericUpDown1_ValueChanged(object sender, EventArgs e) {  
    decimal d = numericUpDown1.Value;  
    label1.Text = "Log " + d + " = ";  
    textBox1.Text = System.Math.Log10((double)d).ToString();  
    label2.Text = d + "*" + d + " = ";  
    textBox2.Text = System.Math.Pow((double)d,2).ToString();  
    label3.Text = "√" + d + " = ";  
    textBox3.Text = System.Math.Sqrt((double)d).ToString();  
}
```

실행 방법 : 수치적 업다운 컨트롤의 값을 변경한다.

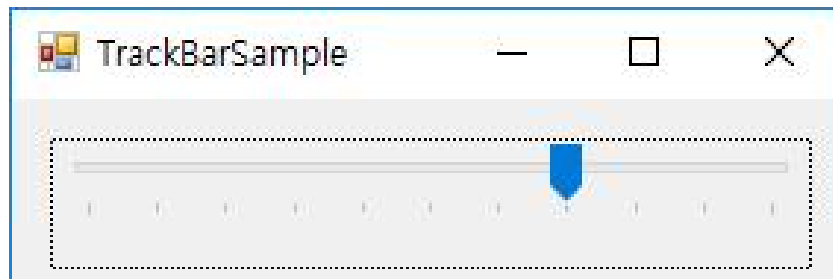
실행 결과 :





트랙 바

- 범위 내에서 값을 선택할 수 있는 컨트롤
 - 슬라이더와 눈금으로 구성



- 슬라이더의 이동
 - 마우스 드래그
 - 슬라이더의 좌우 공간 클릭
 - 마우스 휠의 회전
 - 키보드의 좌우 방향키, 페이지 업다운키



트랙 바의 작성 [1/4]





- 트랙 바의 추가
 - 【도구상자】∅【TrackBar】를 선택하여 폼에 추가
- 트랙 바의 값에 대한 범위와 이동량을 설정
 - 트랙바의 프로퍼티를 통해 설정

프로퍼티	설 명
Minimum	트랙 바의 최소 값
Maximum	트랙 바의 최대 값
Value	트랙 바의 현재 값
LargeChange	마우스 클릭이나 PageUp/PageDown 키에 대한 이동량
SmallChange	마우스 휠의 회전이나 키보드의 방향키에 대한 이동량
TickFrequency	눈금이 표시되는 값의 범위
TickStyle	트랙 바에 눈금이 표시되는 위치
Orientation	트랙 바의 방향(Horizontal Vertical)



트랙 바의 작성 [2/4]

- 슬라이더 형태와 눈금이 표시되는 위치 설정
 - TickStyle 프로퍼티에 TickStyle 열거형 값을 배정하여 설정
 - TickStyle 열거형

기호상수	슬라이더	설 명
None		눈금을 표시하지 않음.
TopLeft		트랙 바의 Orientation 프로퍼티가 Horizontal로 설정된 경우 슬라이더의 상단에 눈금 표시. 트랙 바의 Orientation 프로퍼티가 Vertical로 설정된 경우 슬라이더의 좌측에 눈금 표시.
BottomRight		트랙 바의 Orientation 프로퍼티가 Horizontal로 설정된 경우 슬라이더의 하단에 눈금 표시. 트랙 바의 Orientation 프로퍼티가 vertical로 설정된 경우 슬라이더의 우측에 눈금 표시.
Both		슬라이더의 양쪽에 눈금 표시.



트랙 바의 작성 [3/4]

[예제 11.6 – TrackBarApp]

1) 디자인



컨트롤 : (Name)	프로퍼티	값
Form : Form1	Text	TrackBarApp
TrackBar : trackBar1	Minimum	0
	Maximum	100
	LargeChange	5
	SmallChange	1
	TickFrequency	10
	TickStyle	Both
	Orientation	Horizontal
NumericUpDown : numericUpDown1	Minimum	0
	Maximum	100

컨트롤 : (Name)	이벤트	메소드명
TrackBar : trackBar1	Scroll	trackBar1_Scroll()
NumericUpDown : numericUpDown1	ValueChanged	numericUpDown1_ValueChanged()



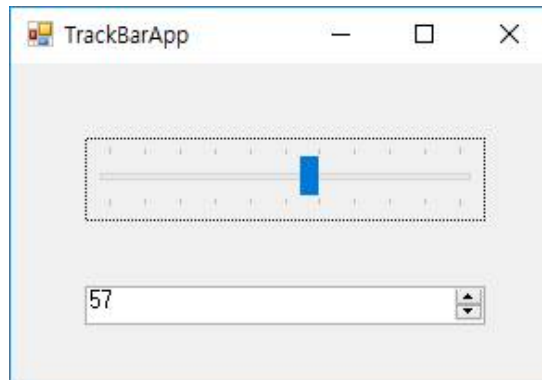
트랙 바의 작성 [4/4]

2) 코드

```
private void trackBar1_Scroll(object sender, EventArgs e) {  
    numericUpDown1.Value = trackBar1.Value;  
}  
private void numericUpDown1_ValueChanged(object sender, EventArgs e) {  
    trackBar1.Value = (int)numericUpDown1.Value;  
    // 수치적 업다운 컨트롤의 Value프로퍼티는 Decimal 형  
    // trackBar의 Value 프로퍼티는 정수형  
}
```

실행 방법 : 트랙 바의 슬라이더를 이동하거나 수치적 업다운 컨트롤의 값을 변경한다.

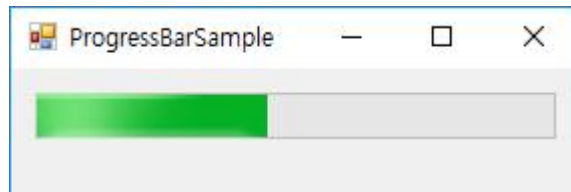
실행 결과 :





프로그레스 바

- 작업의 진행상황을 보여주는 컨트롤
 - 좌측에서 우측으로 사각형의 조각을 채우면서 진행
 - 애플리케이션의 설치과정이나 파일 복사과정에서 사용





프로그레스 바의 작성 [1/2]

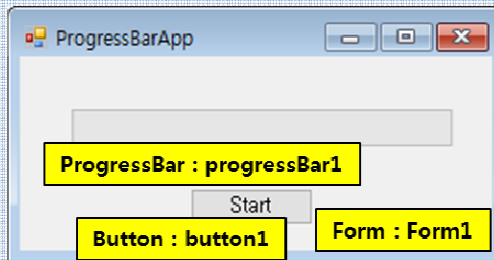
- 프로그레스 바의 추가
 - 【도구상자】∅【ProgressBar】를 선택하여 폼에 추가
- 프로그레스 바의 값에 대한 범위를 설정
 - 프로그레스 바의 프로퍼티를 통해 설정
 - Maximum
 - 프로그레스 바의 최대값
 - Minimum
 - 프로그레스 바의 최소값



프로그레스 바의 작성 [2/2]

[예제 11.7 – ProgressBarApp]

1) 디자인



컨트롤 : (Name)	프로퍼티	값
Form : Form1	Text	ProgressBarApp
Button : button1	Text	Start
ProgressBar : progressBar1	Minimum	0
	Maximum	100000
컨트롤 : (Name)	이벤트	메소드명
Button : button1	Click	button1_Click()

2) 코드

```
private void button1_Click(object sender, EventArgs e) {
    for (int i = 0; i < 100000; i++)
        progressBar1.Value += i;
}
```

실행 방법 : Start 버튼을 클릭한다.

실행 결과 :





타이머 [1/3]

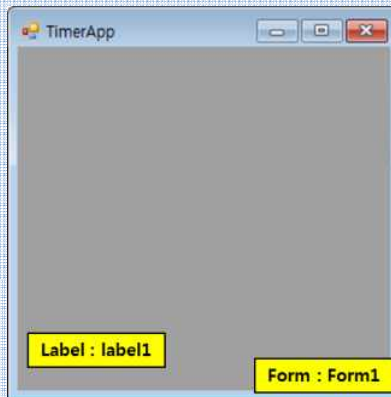
- 주기적인 간격으로 이벤트를 발생시키는 컴포넌트
 - 배경작업을 처리할 때 주로 사용
 - 일정한 간격에 따라 **Tick 이벤트**를 발생
 - **Interval 프로퍼티**를 통해 간격을 설정
 - **밀리 초**(millisecond, 1/1000초)를 사용
 - 주기적으로 발생시키기 위해서는 **Enable 프로퍼티**를 참으로 설정
 - 항상 Interval 프로퍼티의 간격에 따라 Tick 이벤트가 발생하는 것은 아님
 - Tick 이벤트가 다른 이벤트에 비해 우선순위가 낮기 때문
 - 타이머 컴포넌트의 추가
 - **【도구상자】** ∅ **【Timer】**를 선택하여 폼에 추가



타이머 [2/3]

[예제] 11.8 – TimerApp

1) 디자인



컴포넌트 : (Name)	프로퍼티	값
Timer : timer1	Enable	True
	Interval	100
ImageList : imageList1	Images	frame-1.png frame-2.png frame-3.png frame-4.png frame-5.png frame-6.png frame-7.png frame-8.png

* 이미지 경로:

<http://opengameart.org/content/game-character-blue-flappy-bird-sprite-sheets>

컨트롤 : (Name)	프로퍼티	값
Form : Form1	Text	TimerApp
Label : label1	Text	
	Dock	Fill
	BackColor	ButtonShadow
컴포넌트 : (Name)	이벤트	메소드명
Timer : timer1	Tick	timer1_Tick()



타이머 [3/3]

2) 코드

```
private int index = 0;  
private void timer1_Tick(object sender, EventArgs e) {  
    index %= imageList1.Images.Count;  
    label1.Image = imageList1.Images[index++];  
}
```

실행 결과 :

