

## 7주차 프로젝트 (기업솔루션프로젝트)

### 예제\_1 - 프로세스 보기

윈도우 운영체제의 작업관리자의 일부 기능으로 ListView, StatusStrip, Button 컨트롤과 Thread를 이용하여 프로세스 보기 및 예제를 구현한다.

프로젝트 이름 : **ProcessView**

#### [1-1] 프로세스 디자인에 사용된 컨트롤의 주요 속성값

폼 컨트롤	속성	값
Form1	Name	Form1
	Text	프로세스 보기
	FormBorderStyle	FixedSingle
	MaximizeBox	False
	MinimizeBox	False
ListView1	Name	lvView
	Columns	[설정]
	FullRowSelect	True
	GridLines	True
	View	Details
Button1	Name	btnKill
	Text	프로세스 끝내기
StatusStrip1	Name	ssBar

#### [1-2] lvView 컨트롤의 Columns 속성값을 설정

- 1 lvView 컨트롤을 선택하고 속성 목록 창에서 [Columns] 항목의 ...(컬렉션) 버튼을 클릭
- 2 [ColumnHeader 컬렉션 편집기] 대화상자가 나타나면 [멤버] 항목 아래에서 [추가] 버튼을 클릭하여 필요한 4개 컬럼을 위한 멤버 항목을 멤버를 추가하고 멤버의 속성값 설정

폼 컨트롤	속성	값
ColumnHeader1	Name	chName
	Text	프로세스 이름
	Width	108
ColumnHeader2	Name	chPid
	Text	PID
	TextAlign	Center
	Width	60
ColumnHeader3	Name	chTime
	Text	Time
	TextAlign	Center
	Width	90
ColumnHeader4	Name	chMemory
	Text	메모리 사용
	TextAlign	Right
	Width	100

- ③ 상태 바를 설정하기 위해 ssBar 컨트롤을 선택하고 속성 창에서 [Items] 항목의 ...(컬렉션) 버튼을 클릭하고 [항목 컬렉션 편집기] 대화상자를 연다.
- ④ [항목 컬렉션 편집기] 대화상자의 왼쪽 위에 목록 상자에서 StatLabel을 선택하고 [추가] 버튼을 클릭하여 필요한 3개 항목을 추가

폼 컨트롤	속성	값
ToolStripStatusLabel1	Name	tsslProcess
	Text	프로세스 : 0개
ToolStripStatusLabel2	Name	tsslCpu
	Text	CPU 사용 : 0%
ToolStripStatusLabel3	Name	tsslMem
	Text	실제 메모리 : 0%

### [1-3] 프로세스 코드 구현하기

- ① using 키워드를 이용하여 필요한 네임스페이스를 추가  
 using System.Diagnostics; //Process 클래스 사용  
 using System.Collections; //ArrayList 사용  
 using System.Threading; //스레드 사용
- ② **ProcessUpdate 메서드** : ProcessThread 보조 스레드에 대입되어 실행되며, 프로세스 정보를 감시하여 프로세스가 생성되거나 삭제되어 변경되면 업데이트하는 작업 수행
- ③ **ProcessView() 메서드** : 실행 중인 프로세스 정보를 가져와 IvView 컨트롤에 나타내는 작업 수행
- ④ **NumFormat() 메서드** : Long 타입의 숫자를 산술식에 의해 계산한 후 string 타입으로 변환하여 반환
- ⑤ **getCPU Info() 메서드** : 보조 스레드에서 실행되는 대리자로 while 문을 이용하여 1초마다 CPU와 메모리 사용량을 체크하여 OnTotal 델리게이트를 호출하는 작업 수행
- ⑥ **TotalView() 메서드** : OnTotal 델리게이트에 의해 실행되며, CPU와 메모리 사용량을 확인하여 ToolStripStatusLabel 컨트롤에 나타내는 작업 수행
- ⑦ **btnKill Click 이벤트 핸들러** : [프로세스 끝내기] 버튼을 더블클릭하여 생성한 프로시저로, ProcessKill() 메서드를 호출하여 선택된 프로세스를 종료하는 작업 수행
- ⑧ **Form1 FormClosing 이벤트 핸들러** : 폼을 선택하고 이벤트 목록 창에서 [FormClosing] 항목을 더블 클릭하여 생성한 프로시저로, 폼을 종료할 때 발생하는 이벤트를 제어하는 작업 수행

### [1-4] 프로세스 보기 예제 실행

프로세스 목록을 확인하고 임의의 프로세스를 선택한 뒤 [프로세스 끝내기] 버튼을 클릭하여 선택한 프로세스가 종료되는 것을 확인해 본다.

#### [Source]

```
using System;
using System.Threading.Tasks;
using System.Windows.Forms;

using System.Diagnostics; //Process 클래스 사용
using System.Collections; //ArrayList 사용
using System.Threading; //스레드 사용
```

**namespace ProcessView**

```
{
    public partial class Form1 : Form
    {
        public Form1()
        {
            InitializeComponent();
        }

        private Thread ProcessThread; // 프로세스 나타내기 스레드
        Thread checkThread = null; //실시간 시스템 정보 체크

        private delegate void ProcessUpdateDelegate(); // 델리게이트 생성
        private ProcessUpdateDelegate UpProc = null;

        private delegate void TotalUpdateDelegate(int m, int n); // 델리게이트 생성
        private TotalUpdateDelegate OnTotal = null;

        private PerformanceCounter oCPU =
            new PerformanceCounter("Processor", "% Processor Time", "_Total");
            //시스템 CPU 성능 카운터

        private PerformanceCounter oMem =
            new PerformanceCounter("Memory", "% Committed Bytes In Use");
            //시스템 Mem 성능 카운터

        private PerformanceCounter pCPU = new PerformanceCounter();

        bool bExit = false;
        int cp = 0;

        private void Form1_Load(object sender, EventArgs e)
        {
            ProcessView(); //프로세스 출력

            UpProc = new ProcessUpdateDelegate(ProcessView); //델리게이트에 구동 메서드 입력
            OnTotal = new TotalUpdateDelegate(TotalView);

            ProcessThread = new Thread(ProcessUpdate); //스레드 대리자에 구동 메서드 입력
            ProcessThread.Start(); //스레드 시작

            checkThread = new Thread(getCPU_Info);
            checkThread.Start();
        }

        private void ProcessUpdate()
        {
            try
            {
                while (true)
                {
                    var oldlist = new ArrayList();

                    foreach (var oldproc in Process.GetProcesses())
                    {
                        oldlist.Add(oldproc.Id.ToString());
                    }
                }
            }
            catch { }
        }
    }
}
```

```

Thread.Sleep(1000);
var newproc = Process.GetProcesses();

if (oldlist.Count != newproc.Length)
{
    Invoke(UpProc);
    continue;
}

int i = 0;

foreach (var rewproc in Process.GetProcesses())
{
    if (oldlist[i++].ToString() != rewproc.Id.ToString())
    {
        Invoke(UpProc);
        break;
    }
}
}
}
catch { }
}

```

#### private void ProcessView()

```

{
    try
    {
        this.lvView.Items.Clear();
        cp = 0;
        foreach (var proc in Process.GetProcesses())
        {
            string[] str;
            try
            {
                str = proc.TotalProcessorTime.ToString().Split(new Char[] { '.' });
            }
            catch { str = new string[] { "" }; }

            var strArray = new string[] { proc.ProcessName.ToString(), proc.Id.ToString(),
                str[0], NumFormat(proc.WorkingSet64) };

            var lvt = new ListViewItem(strArray);
            this.lvView.Items.Add(lvt);
            cp++;
        }
    }
    catch { }
    this.tsslProcess.Text = "프로세스 : " + cp.ToString() + "개";
}

```

#### private void TotalView(int m, int n)

```

{
    this.tsslCpu.Text = "CPU 사용: " + m.ToString() + " %";
    this.tsslMem.Text = "실제 메모리 : " + n.ToString() + " %";
}

```

```

private string NumFormat(long MemNum)
{
    MemNum = MemNum / 1024;
    return String.Format("{0:N}", MemNum) + " KB";
}

private void getCPU_Info()
{
    while (!bExit)
    {
        int iCPU = (int)oCPU.NextValue();
        int iMem = (int)oMem.NextValue();
        Invoke(OnTotal, iCPU, iMem);
        Thread.Sleep(1000);
    }
}

private void btnKill_Click(object sender, EventArgs e)
{
    ProccessKill();
}

private void ProccessKill()
{
    try
    {
        int PID = Convert.ToInt32(this.lvView.SelectedItems[0].SubItems[1].Text);
        Process tProcess = Process.GetProcessById(PID);
        if (!(tProcess == null))
        {
            var dlr = MessageBox.Show(this.lvView.SelectedItems[0].SubItems[0].Text +
                " 프로세스를 끝내시겠습니까?", "알림", MessageBoxButtons.YesNo,
                MessageBoxIcon.Warning);
            if (dlr == DialogResult.Yes)
            {
                tProcess.Kill();
                ProcessView();
            }
        }
        else
        {
            MessageBox.Show(this.lvView.SelectedItems[0].SubItems[0].Text +
                "프로세스는 존재하지 않습니다", "알림", MessageBoxButtons.OK,
                MessageBoxIcon.Error);
            ProcessView();
        }
    }
    catch
    {
        return;
    }
}

private void Form1_FormClosing(object sender, FormClosingEventArgs e)
{
    if (!(ProcessThread == null))

```

```
        ProcessThread.Abort(); //스레드 종료
    if (!(checkThread == null))
        checkThread.Abort(); //스레드 종료
    }
}
```