Task Manager

Grupa 30431

Rus Tudor

Contents

1. Introduction
   1.1 Context
   1.2 Specifications
   1.3 Objectives
2. Bibliographic study
3. Analysis
4. Design and Implementation
5. Testing
6. Conclusion
7. Bibliography

1. Introduction

1.1 Context

The goal of this project is to design, implement and test a task manager software for windows similar to the existing task manager.

1.2 Specifications

The software will be developed using Visual Studio (C#).

1.3 Objectives

List tasks
End tasks
Memory usage graph
CPU usage graph
Information about : cache memory, system architecture, cpu etc

2. Bibliographic study

I studied real time data graphs in C#, the Process class in C#, (forms, labels, buttons, tables etc) for the UI, timed execution of a thread and how to extract data from system files.

3. Analysis

I will need a list containing the active tasks. That list will be inserted in the table to be easily viewed and manipulated by the user.

Two graphs that show in real time the memory and cpu usage on a different tab of the frame.

The last tab will contain information.

4. Design and Implementation

Using the process class, the processes are stored in a list that is inserted in the listview. An updateprocess function gets a new list of active tasks and updates the process list and the listview by comparing the new list to the old one.

For the graphs, performance counters (for cpu and ram) update the cpu and mem arrays with the new value.

For the system info, we retrieve the information from system files using management object searchers.

Function that calls the process update function at an interval.

```
private void timerTick(object sender, EventArgs e)
        {
            updateProcessList();
           }
```

Function that updates the cpu usage and puts the new values in the cpuArray used in the next function that updates the graph.

```
private void getPerfCounters()
        {
            var cpuPerfCounter = new PerformanceCounter("Processor Information","%
Processor Time","_Total");

            while (true)
            {
                cpuArray[cpuArray.Length - 1] = Math.Round(cpuPerfCounter.NextValue(),
0);

                Array.Copy(cpuArray, 1, cpuArray, 0, cpuArray.Length - 1);

                if (chart1.IsHandleCreated)
                    this.Invoke((MethodInvoker)delegate { UpdateCpuChart(); });
                Thread.Sleep(500);
            }
               }

private void UpdateCpuChart()
        {
            chart1.Series["CPU%"].Points.Clear();

            for (int i = 0; i < cpuArray.Length - 1; ++i)
            {
                chart1.Series["CPU%"].Points.AddY(cpuArray[i]);
            }
           }
```

The function that ends the selected task from the tasklist.

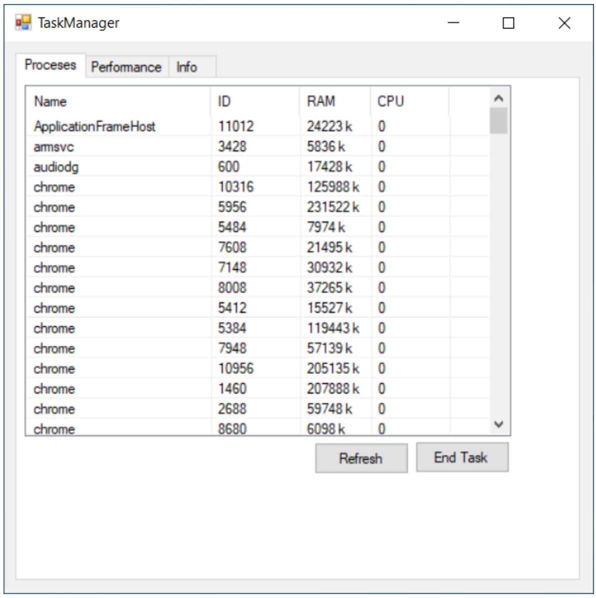
```
private void end_Click(object sender, EventArgs e)
        {
            ListViewItem item = listView1.SelectedItems[0];
            Process p = (Process)item.Tag;
            p.Kill();
            loadProcessList();
           }
```

The management object searcher that retrieves data about the system.

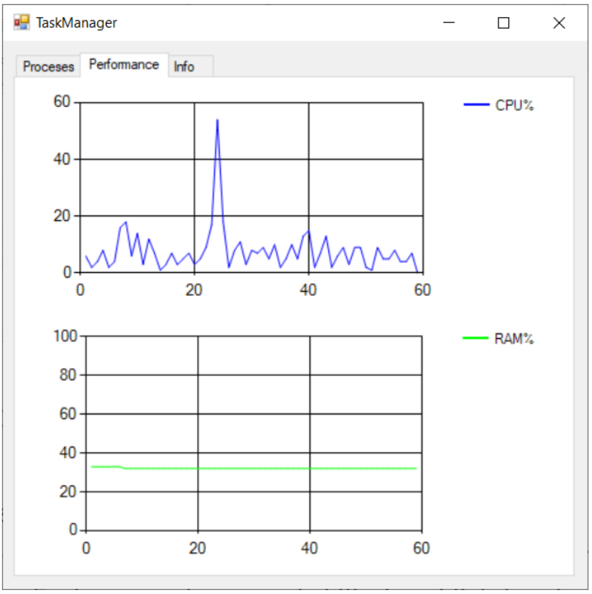
We then select what data we need using this object.

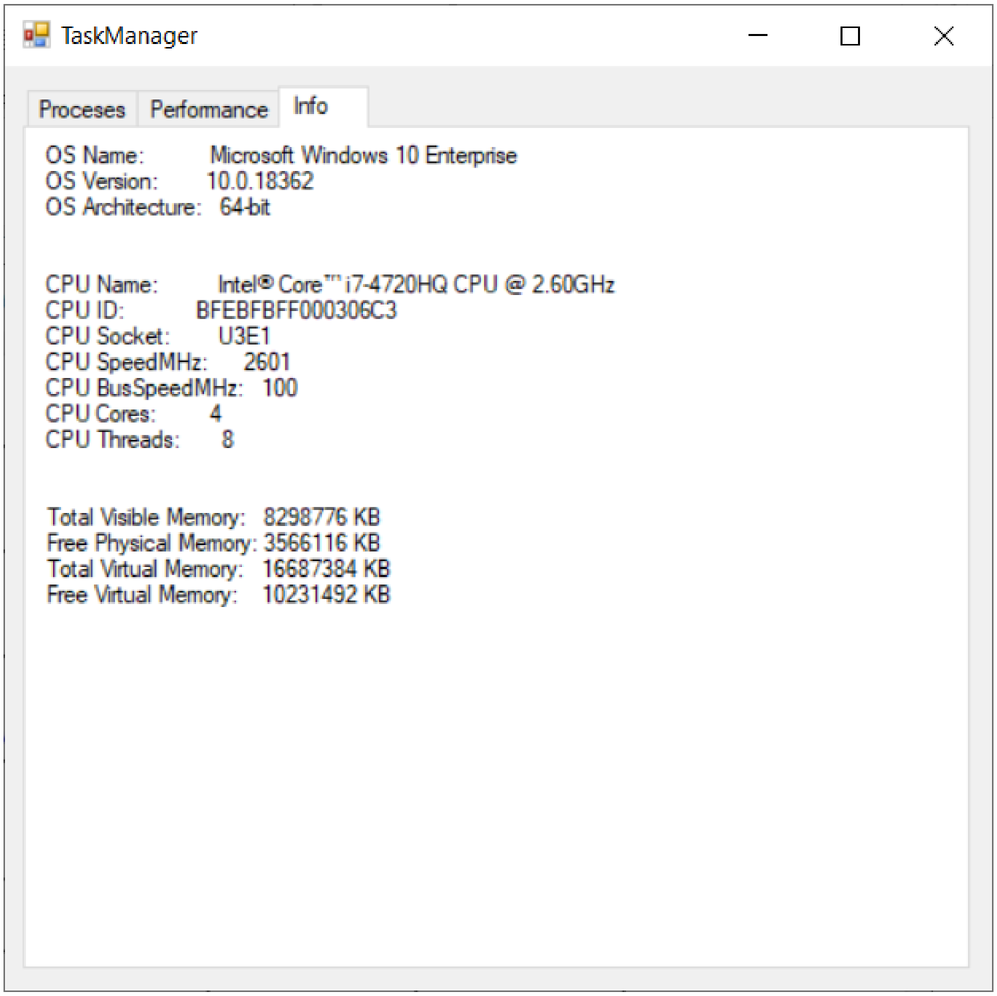
```
var wmi =
            new ManagementObjectSearcher("select * from Win32_OperatingSystem")
            .Get()
            .Cast<ManagementObject>()
                .First();
```

Screenshots



Name	ID	RAM	CPU
ApplicationFrameHost	11012	24223 k	0
armsvc	3428	5836 k	0
audiodg	600	17428 k	0
chrome	10316	125988 k	0
chrome	5956	231522 k	0
chrome	5484	7974 k	0
chrome	7608	21495 k	0
chrome	7148	30932 k	0
chrome	8008	37265 k	0
chrome	5412	15527 k	0
chrome	5384	119443 k	0
chrome	7948	57139 k	0
chrome	10956	205135 k	0
chrome	1460	207888 k	0
chrome	2688	59748 k	0
chrome	8680	6098 k	0





OS Name: Microsoft Windows 10 Enterprise
OS Version: 10.0.18362
OS Architecture: 64-bit

CPU Name: Intel® Core™ i7-4720HQ CPU @ 2.60GHz
CPU ID: BFEBFBFF000306C3
CPU Socket: U3E1
CPU SpeedMHz: 2601
CPU BusSpeedMHz: 100
CPU Cores: 4
CPU Threads: 8

Total Visible Memory: 8298776 KB
Free Physical Memory: 3566116 KB
Total Virtual Memory: 16687384 KB
Free Virtual Memory: 10231492 KB

5. Testing

Cases tested:

- Task listing (comparing to real task manager)
- End process(process is killed and listview is updated)
- Change in cpu and ram(by opening and closing apps and comparing values and graphs to the real task manager)
- System information(by running the program on a different system)

6. Conclusion

The project was completed successfully.

Future possible improvements: better UI, better graph representation, more data about the system.

7. Bibliography

youtube.com (graph representation and UI)
stackoverflow.com (extracting data from system files, timed threads, process class)