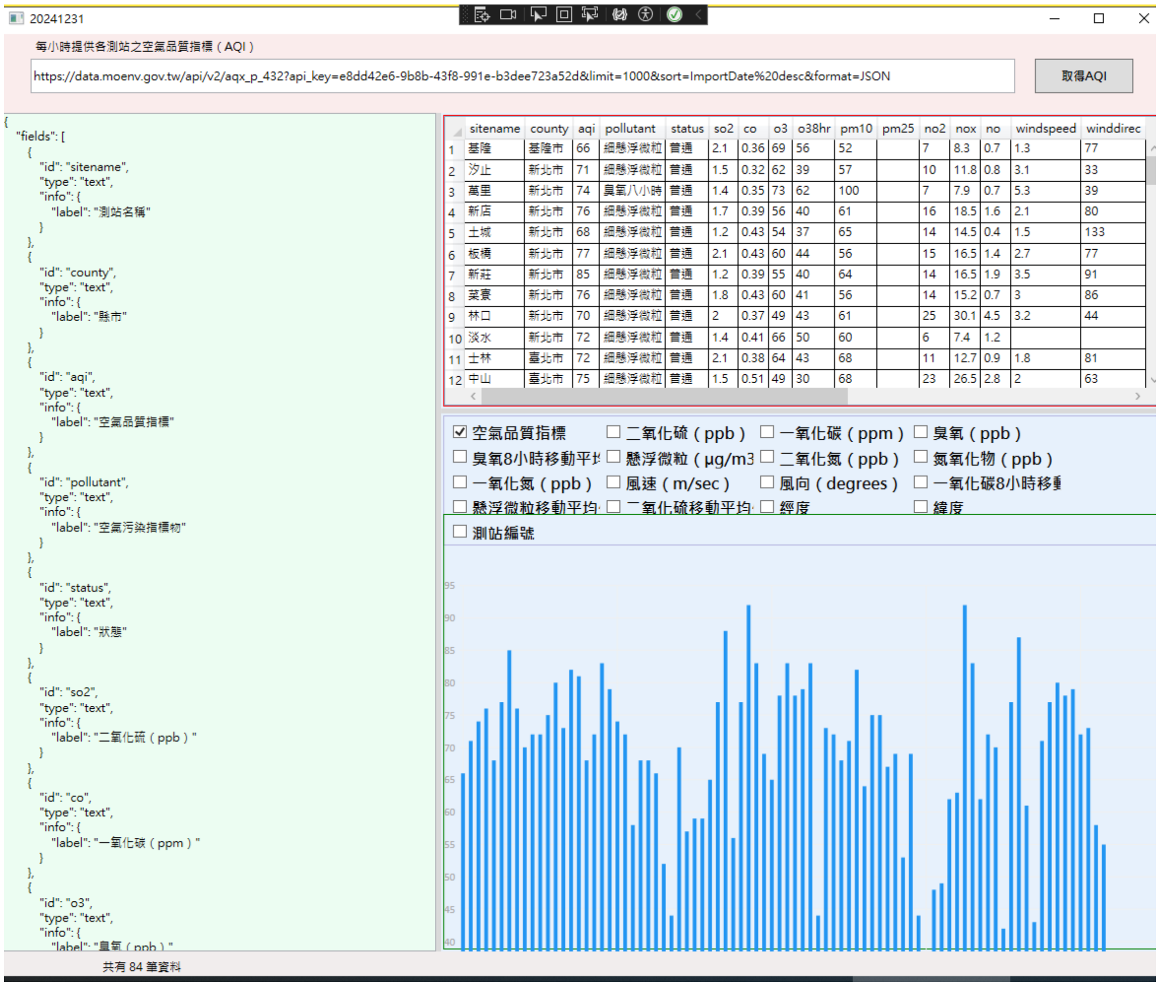

每小時提供各測站之空氣品質指標(AQI)

https://data.moenv.gov.tw/api/v2/aqx_p_432?api_key=e8dd42e6-9b8b-43f8-991e-b3dee723a52d&limit=1000&sort=ImportDate%20desc&format=JSON

```
{
  "fields": [
    {
      "id": "sitename",
      "type": "text",
      "info": {
        "label": "測站名稱"
      }
    },
    {
      "id": "county",
      "type": "text",
      "info": {
        "label": "縣市"
      }
    },
    {
      "id": "aqi",
      "type": "text",
      "info": {
        "label": "空氣品質指標"
      }
    },
    {
      "id": "pollutant",
      "type": "text",
      "info": {
        "label": "空氣污染指標物"
      }
    },
    {
      "id": "status",
      "type": "text",
      "info": {
        "label": "狀態"
      }
    },
    {
      "id": "so2",
      "type": "text",
      "info": {
        "label": "二氧化硫(ppb)"
      }
    },
    {
      "id": "co",
      "type": "text",
      "info": {
        "label": "一氧化碳(ppm)"
      }
    },
    {
      "id": "o3",
      "type": "text",
      "info": {
        "label": "臭氧(ppb)"
```

	sitename	county	aqi	pollutant	status	so2	co	o3	o38hr	pm10	pm25	no2	nox	no	windspeed	winddirec
1	基隆	基隆市	66	細懸浮微粒	普通	2.1	0.36	69	56	52		7	8.3	0.7	1.3	77
2	汐止	新北市	71	細懸浮微粒	普通	1.5	0.32	62	39	57		10	11.8	0.8	3.1	33
3	萬里	新北市	74	臭氧八小時	普通	1.4	0.35	73	62	100		7	7.9	0.7	5.3	39
4	新店	新北市	76	細懸浮微粒	普通	1.7	0.39	56	40	61		16	18.5	1.6	2.1	80
5	土城	新北市	68	細懸浮微粒	普通	1.2	0.43	54	37	65		14	14.5	0.4	1.5	133
6	板橋	新北市	77	細懸浮微粒	普通	2.1	0.43	60	44	56		15	16.5	1.4	2.7	77
7	新莊	新北市	85	細懸浮微粒	普通	1.2	0.39	55	40	64		14	16.5	1.9	3.5	91
8	菜寮	新北市	76	細懸浮微粒	普通	1.8	0.43	60	41	56		14	15.2	0.7	3	86
9	林口	新北市	70	細懸浮微粒	普通	2	0.37	49	43	61		25	30.1	4.5	3.2	44
10	淡水	新北市	72	細懸浮微粒	普通	1.4	0.41	66	50	60		6	7.4	1.2		
11	士林	臺北市	72	細懸浮微粒	普通	2.1	0.38	64	43	68		11	12.7	0.9	1.8	81
12	中山	臺北市	75	細懸浮微粒	普通	1.5	0.51	49	30	68		23	26.5	2.8	2	63

☑ 空氣品質指標 ☐ 二氧化硫(ppb) ☐ 一氧化碳(ppm) ☐ 臭氧(ppb)
☐ 臭氧8小時移動平均 ☐ 懸浮微粒(μg/m3 ☐ 二氧化氮(ppb) ☐ 氮氧化物(ppb)
☐ 一氧化氮(ppb) ☐ 風速(m/sec) ☐ 風向(degrees) ☐ 一氧化碳8小時移動
☐ 懸浮微粒移動平均 ☐ 二氧化硫移動平均 ☐ 經度 ☐ 緯度
☐ 測站編號

共有 84 筆資料

```csharp
using LiveCharts;
using LiveCharts.Wpf;
using System.Net.Http;
using System.Text.Json;
using System.Windows;
using System.Windows.Controls;

namespace 20241231
{
    public partial class MainWindow : Window
    {
        // 預設的 URL,用於抓取空氣品質資料
        string defaultURL = "https://data.moenv.gov.tw/api/v2/aqx_p_432?api_key=e8dd42e6-
9b8b-43f8-991e-b3dee723a52d&limit=1000&sort=ImportDate%20desc&format=JSON";

        // 用於儲存 AQI 資料的實體
        AQIData aqiData = new AQIData();
        List<Field> fields = new List<Field>();
        List<Record> records = new List<Record>();
        List<Record> selectedRecords = new List<Record>();
```

```
// 用於儲存圖表的系列資料
SeriesCollection seriesCollection = new SeriesCollection();

public MainWindow()
{
    InitializeComponent();
    UrlTextBox.Text = defaultURL;
}

// 當按下 "Get AQI" 按鈕時，進行資料抓取
private async void GetAQIButton_Click(object sender, RoutedEventArgs e)
{
    ContentTextBox.Text = "抓取資料中...";

    // 非同步抓取資料
    string data = await FetchContentAsync(defaultURL);
    ContentTextBox.Text = data;

    // 反序列化資料成 AQIData 物件
    aqiData = JsonSerializer.Deserialize<AQIData>(data);
    fields = aqiData.fields.ToList();
    records = aqiData.records.ToList();
    selectedRecords = records;

    statusTextBlock.Text = $"共有 {records.Count} 筆資料";

    // 顯示 AQI 資料
    DisplayAQIData();
}

// 顯示 AQI 資料在 DataGrid 中
private void DisplayAQIData()
{
    RecordDataGrid.ItemsSource = records;

    // 取得第一筆資料
    Record record = records[0];

    // 為每個欄位建立 CheckBox
    foreach (Field field in fields)
    {
        var propertyInfo = record.GetType().GetProperty(field.id);
        if (propertyInfo != null)
        {
            var value = propertyInfo.GetValue(record) as string;
            if (double.TryParse(value, out double v))
            {
                CheckBox cb = new CheckBox
                {
                    Content = field.info.label,
                    Tag = field.id,
                    Margin = new Thickness(3),
                        FontSize = 16,
                        FontWeight = FontWeights.Bold,
                        Width = 150
                    };
                    cb.Checked += UpdateChart;
                    cb.Unchecked += UpdateChart;
                    DataWrapPanel.Children.Add(cb);
                }
            }
        }
    }

    // 更新圖表
    private void UpdateChart(object sender, RoutedEventArgs e)
    {
        // 清除現有的系列
        seriesCollection.Clear();

        // 根據 CheckBox 狀態更新圖表資料
        foreach (CheckBox cb in DataWrapPanel.Children)
        {
            if (cb.IsChecked == true)
            {
                List<string> labels = new List<string>();
                string tag = cb.Tag as string;
                ColumnSeries columnSeries = new ColumnSeries();
                ChartValues<double> values = new ChartValues<double>();

                // 為選中的紀錄新增資料點
                foreach (Record r in selectedRecords)
                {
                    var propertyInfo = r.GetType().GetProperty(tag);
                    if (propertyInfo != null)
                    {
                        var value = propertyInfo.GetValue(r) as string;
                        if (double.TryParse(value, out double v))
                        {
                            labels.Add(r.sitename);
                            values.Add(v);
                        }
                    }
                }
                columnSeries.Values = values;
                columnSeries.Title = tag;
                columnSeries.LabelPoint = point =>
$"{labels[(int)point.X]}:{point.Y.ToString()}";
                seriesCollection.Add(columnSeries);
            }
        }
        AQIChart.Series = seriesCollection;
    }
```

```
        // 非同步抓取資料內容
        private async Task<string> FetchContentAsync(string url)
        {
            using (var client = new HttpClient())
            {
                client.Timeout = TimeSpan.FromSeconds(100);
                try
                {
                    HttpResponseMessage response = await client.GetAsync(url);
                    response.EnsureSuccessStatusCode();
                    string responseBody = await response.Content.ReadAsStringAsync();
                    return responseBody;
                }
                catch (HttpRequestException e)
                {
                    MessageBox.Show($"Request exception: {e.Message}");
                    return null;
                }
            }
        }

        // 在 DataGrid 加載行時顯示行號
        private void RecordDataGrid_LoadingRow(object sender,
System.Windows.Controls.DataGridRowEventArgs e)
        {
            e.Row.Header = (e.Row.GetIndex() + 1).ToString();
        }

        // 當 DataGrid 的選擇變更時，更新選擇的紀錄和圖表
        private void RecordDataGrid_SelectionChanged(object sender,
SelectionChangedEventArgs e)
        {
            selectedRecords = RecordDataGrid.SelectedItems.Cast<Record>().ToList();
            statusTextBlock.Text = $"共選擇 {selectedRecords.Count} 筆資料";
            UpdateChart(null, null);
        }
    }
}
```