



미세미세 어플 캡처화면



서울시내 실시간
대기오염정보 확인
어플리케이션

1. 프로젝트 설명

- '미세미세' 라는 어플리케이션을 참고하여 만들었습니다.
- 측정소명을 검색하여 대기오염정보를 확인합니다.
- XAML코드로 디자인되었습니다.
- 원폼처럼 드래그앤 드랍방식이 아닙니다.
- 그리드 레이아웃을 이용하여 좌표별로 컨트롤을 올렸습니다.
- C#코드로 논리를 설계하고 이벤트 처리 또한 구현하였습니다.
- 라벨 클릭 이벤트가 없어 인터넷을 참고하여 직접 구현하였습니다.
- 프로퍼티 사용하여 두 클래스간(두 페이지간) 데이터(검색한 측정소명)를 공유하고 연결되도록 직접 구현하였습니다.
- 자마린을 독학하여 개발하였습니다. android와 ios를 모두 지원합니다.
- Api 1개 사용: (대기오염정보조회- 시도별실시간 측정정보 조회(서울시))
<http://openapi.airkorea.or.kr/openapi/services/rest/ArpltnInforInquireSvc/getCtprvnRltmMeasureDnsty?serviceKey=XhTa978BeUEMjroqJWgb%2FH9pBWX5QMxmE6MUSw15i7hs6epmOucqjI%2BXnn6ruZRIKsZ%2FTFluLxMd42F3vIvb1A%3D%3D&numOfRows=40&pageNo=1&sidName=%EC%84%9C%EC%9A%B8&ver=1.3>

2. 구동화면

- 이상은 구동화면과 각 화면에 대한 설명입니다.

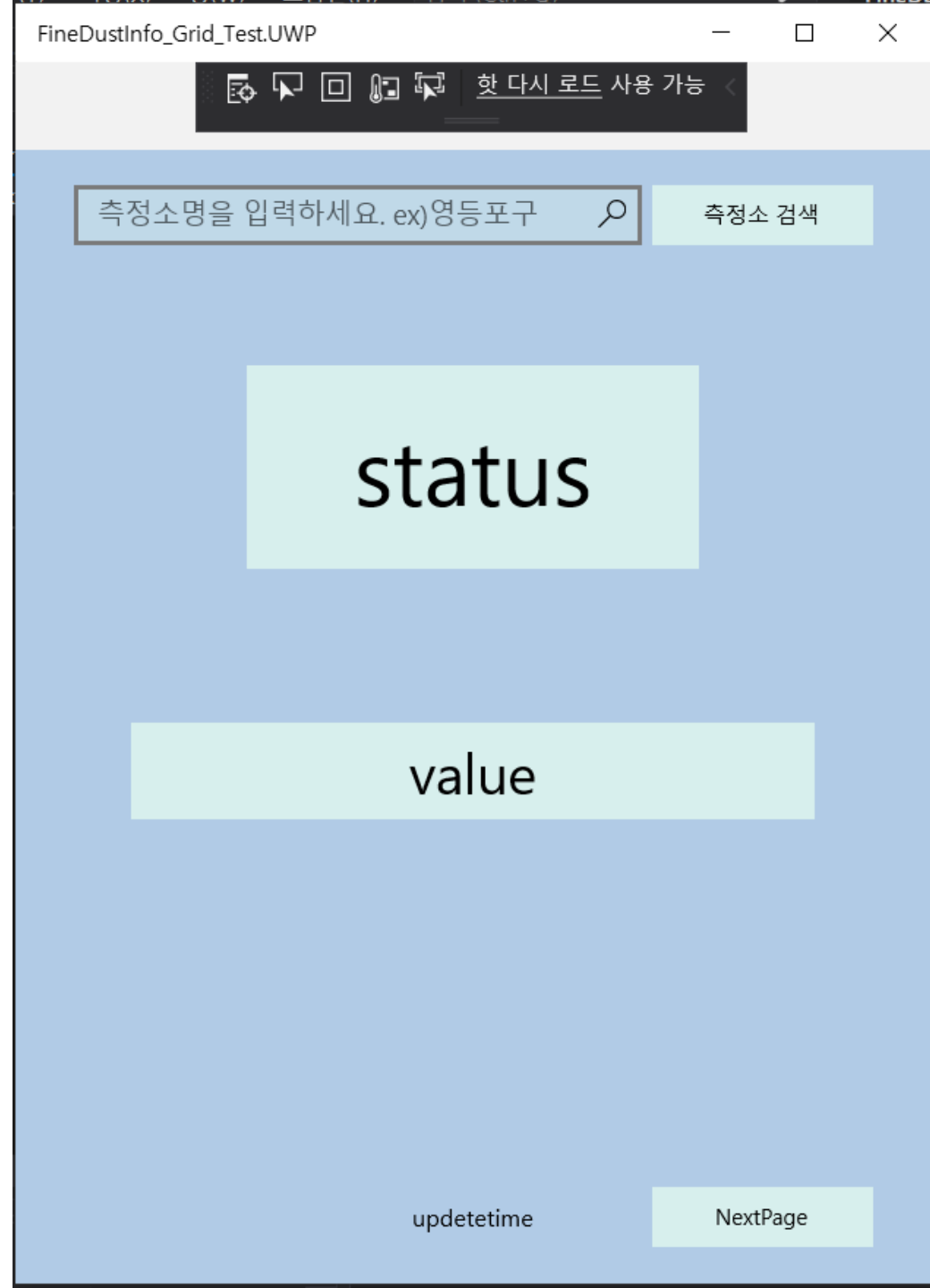
서울시내 25개 구를 포함한 40여 곳
측정소의 대기오염수치 정보를
제공하는 공공api를 사용하였습니다.

BIBLE LMS x 공공데이터포털 > 마이페이지 > x openapi.airkorea.or.kr

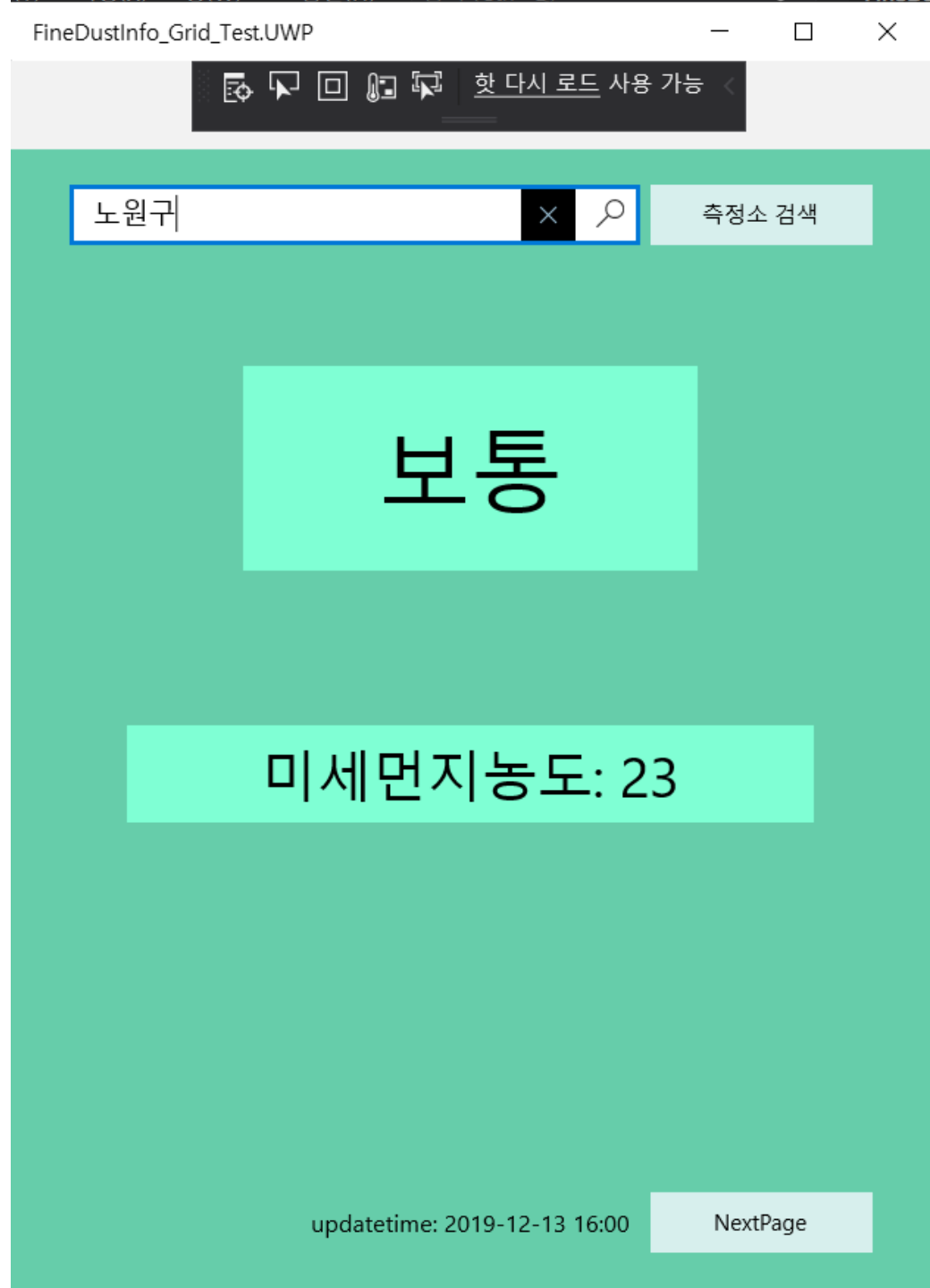
← → ↺ 주의 요함 | openapi.airkorea.or.kr/openapi/services/rest/ArpltnInforInquireSv

```
<?xml version="1.0" encoding="UTF-8"?>
<response>
  <body>
    <items>
      <item>
        <stationName>노원구</stationName>
        <mangName>도시대기</mangName>
        <dateTime>2019-12-13 16:00</dateTime>
        <so2Value>0.005</so2Value>
        <coValue>0.5</coValue>
        <o3Value>0.007</o3Value>
        <no2Value>0.042</no2Value>
        <pm10Value>25</pm10Value>
        <pm10Value24>29</pm10Value24>
        <pm25Value>23</pm25Value>
        <pm25Value24>24</pm25Value24>
        <khaiValue>72</khaiValue>
        <khaiGrade>2</khaiGrade>
        <so2Grade>1</so2Grade>
        <coGrade>1</coGrade>
        <o3Grade>1</o3Grade>
        <no2Grade>2</no2Grade>
        <pm10Grade>1</pm10Grade>
        <pm25Grade>2</pm25Grade>
        <pm10Grade1h>1</pm10Grade1h>
        <pm25Grade1h>2</pm25Grade1h>
      </item>
      <item>
        <stationName>화랑로</stationName>
        <mangName>도로변대기</mangName>
        <dateTime>2019-12-13 16:00</dateTime>
        <so2Value>0.004</so2Value>
        <coValue>0.3</coValue>
        <o3Value>0.008</o3Value>
        <no2Value>0.034</no2Value>
        <pm10Value>-</pm10Value>
        <pm10Value24>-</pm10Value24>
        <pm25Value>-</pm25Value>
        <pm25Value24>-</pm25Value24>
        <khaiValue>-</khaiValue>
        <khaiGrade/>
        <so2Grade>1</so2Grade>
        <coGrade>1</coGrade>
        <o3Grade>1</o3Grade>
        <no2Grade>2</no2Grade>
        <pm10Grade/>
        <pm25Grade/>
        <pm10Grade1h/>
        <pm25Grade1h/>
      </item>
    </items>
    <numOfRows>40</numOfRows>
    <pageNo>1</pageNo>
    <totalCount>40</totalCount>
  </body>
</response>
```

초기화면입니다. 측정소를 검색할 수 있고 대기상태와 미세먼지 농도를 표시합니다. 업데이트 날짜와 시간을 확인할 수 있고 다음 페이지로 넘어갈 수 있습니다.



검색했을 때 해당 측정소 근방의
대기상태와 미세먼지 농도가
표시됩니다. 보통일 때는 배경색이
초록색이 됩니다.



대기오염 상태가 매우 나쁨일 때는
배경색이 빨간색이 됩니다.
좋음>보통>나쁨>매우나쁨 순입니다.

FineDustInfo_Grid_Test.UWP

하트 다시 로드 사용 가능 <

종로 × 🔍 측정소 검색

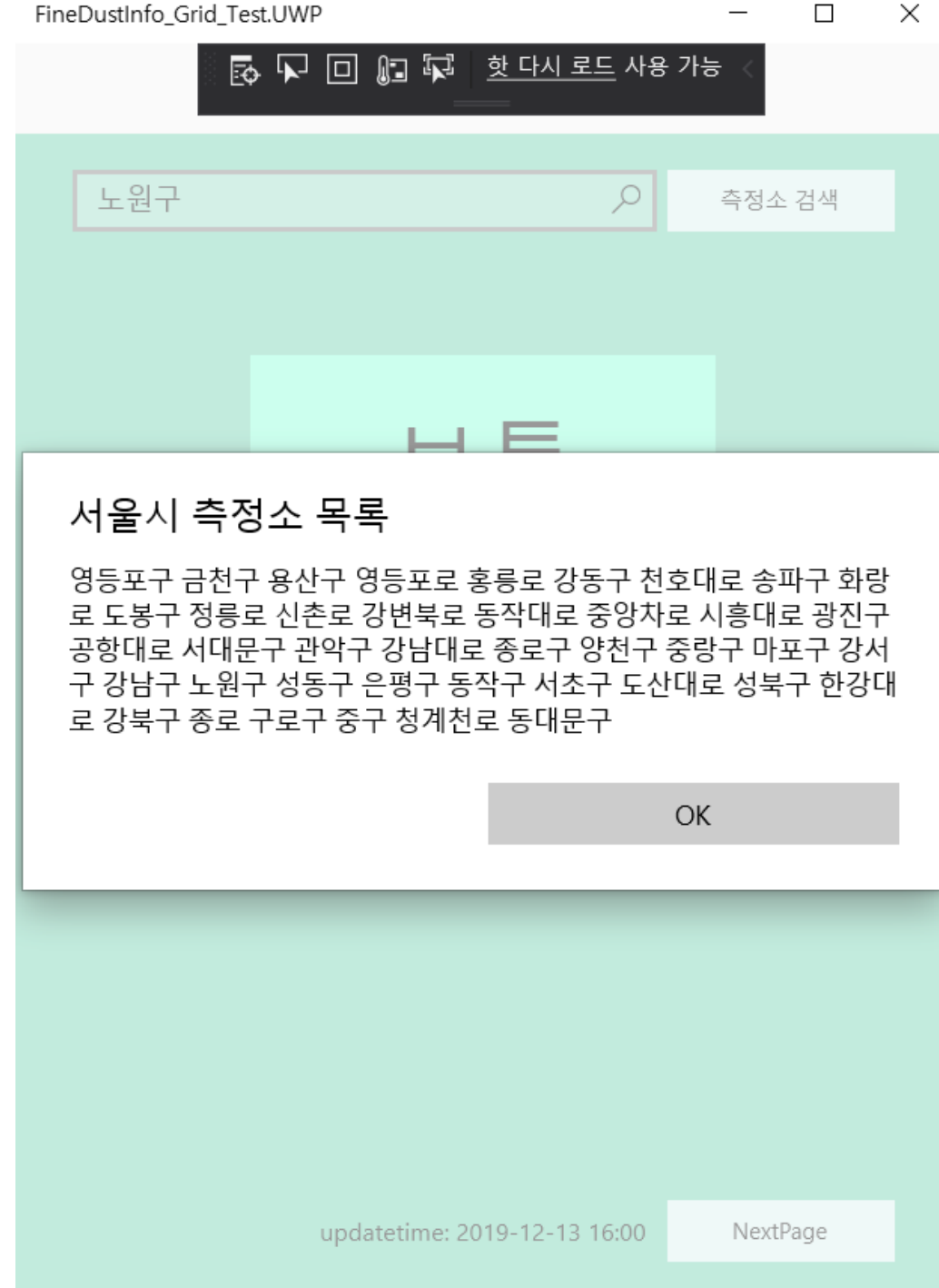
매우 나쁨

미세먼지농도: -

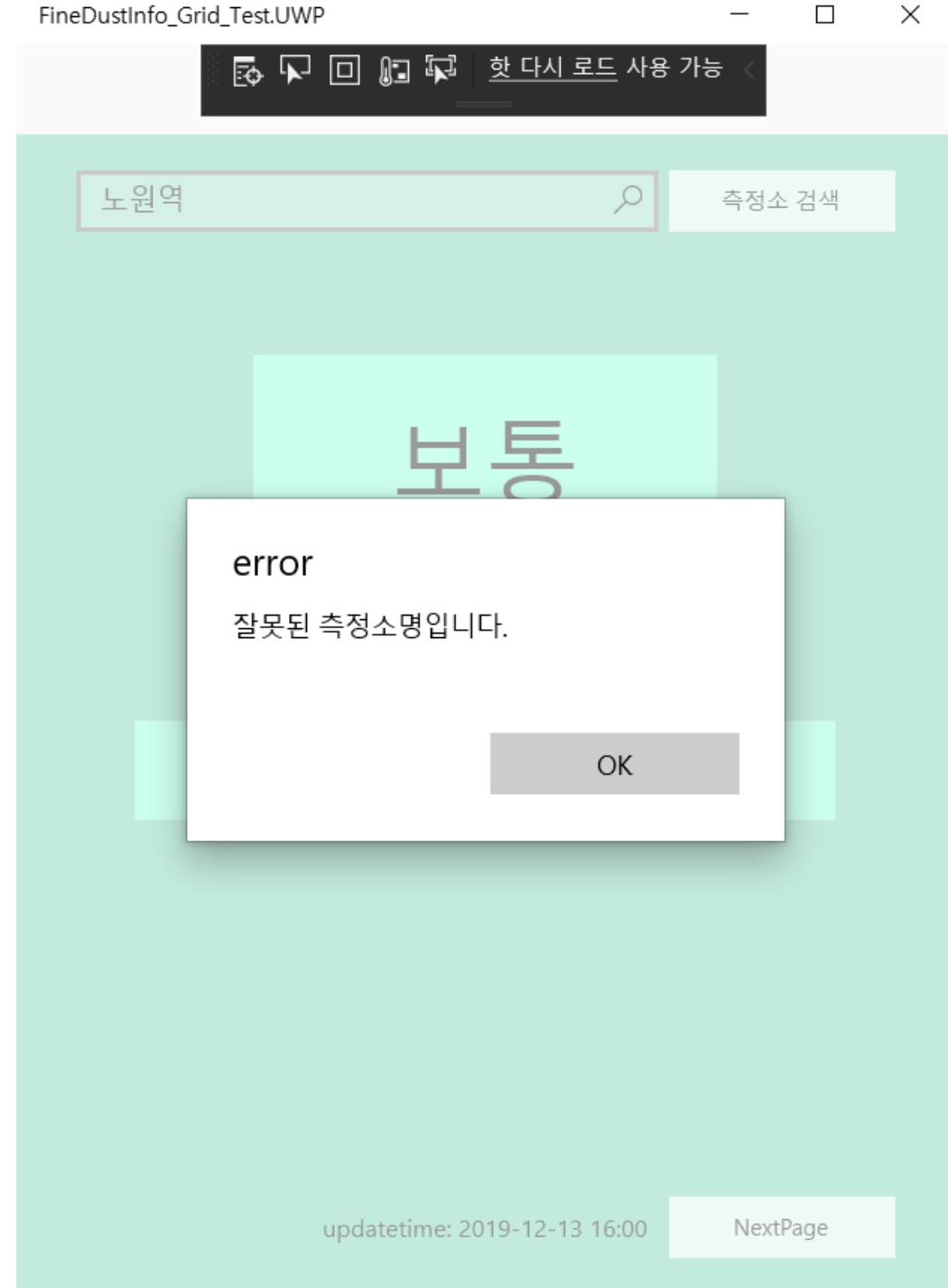
updatetime: 2019-12-13 16:00

NextPage

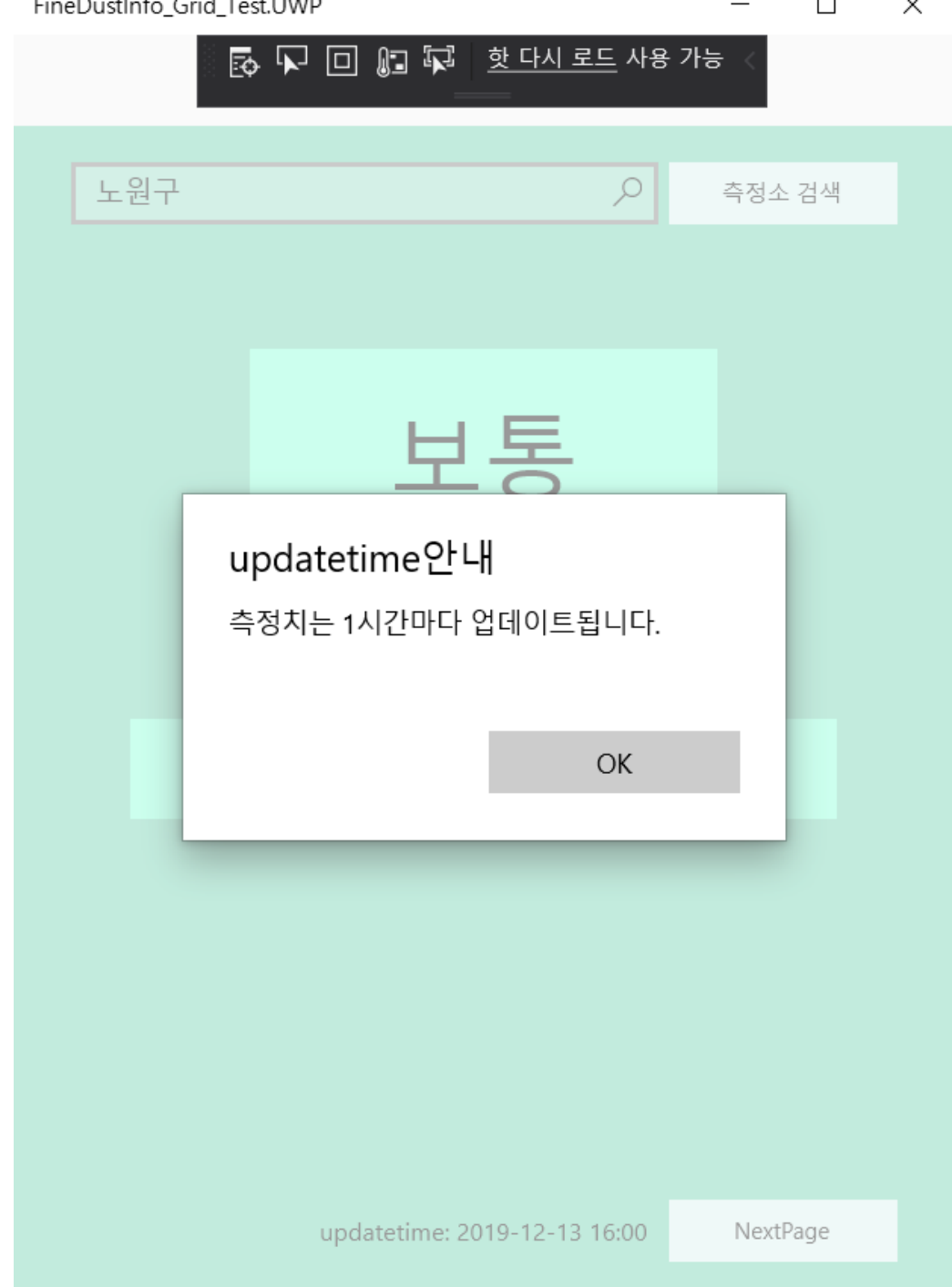
서울시내 측정소 목록을 확인하고
원하는 측정소의 정보를 검색할 수
있습니다.



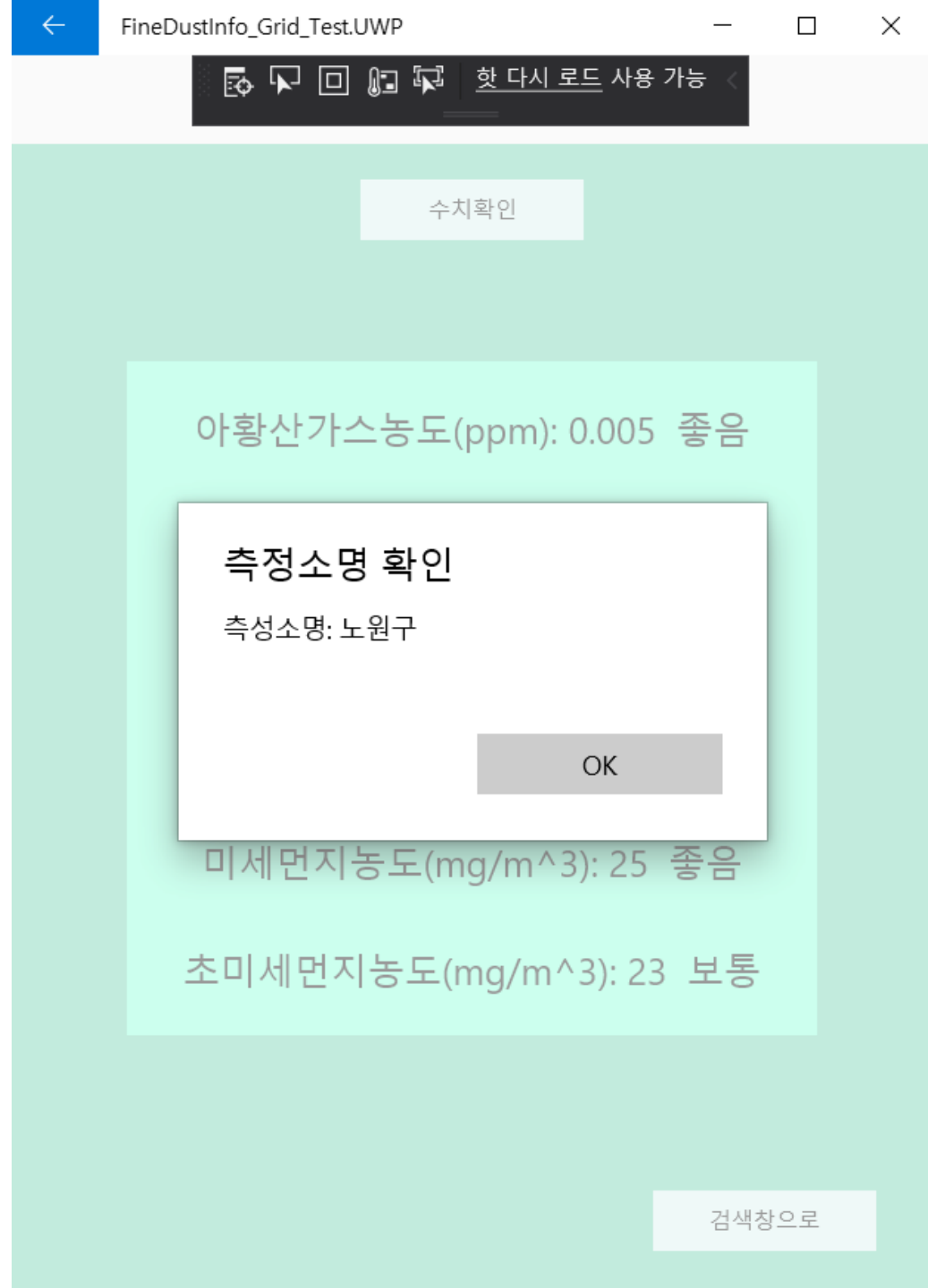
목록에 없는 측정소를 검색하면
에러메시지가 나타납니다.



update라벨을 클릭하면
updatetime안내 메시지가 나타납니다.



NextPage버튼을 클릭한 후의
화면입니다. 측정소 명이 전달된 것을
확인할 수 있습니다.



전달된 측정소명을 사용하여
대기오염정보 자료에 대한 파싱을
수행합니다. 검색한 측정소의 구체적인
대기 오염 농도 값을 확인 할 수
있습니다.

←

FineDustInfo_Grid_Test.UWP

— □ ×

🔍 ↶ ↷ 📏 📱 ↶ ↷

↶

↶ 다시 로드 사용 가능

수치확인

아황산가스농도(ppm): 0.005 좋음

일산화탄소농도(ppm): 0.5 좋음

오존농도(ppm): 0.007 좋음

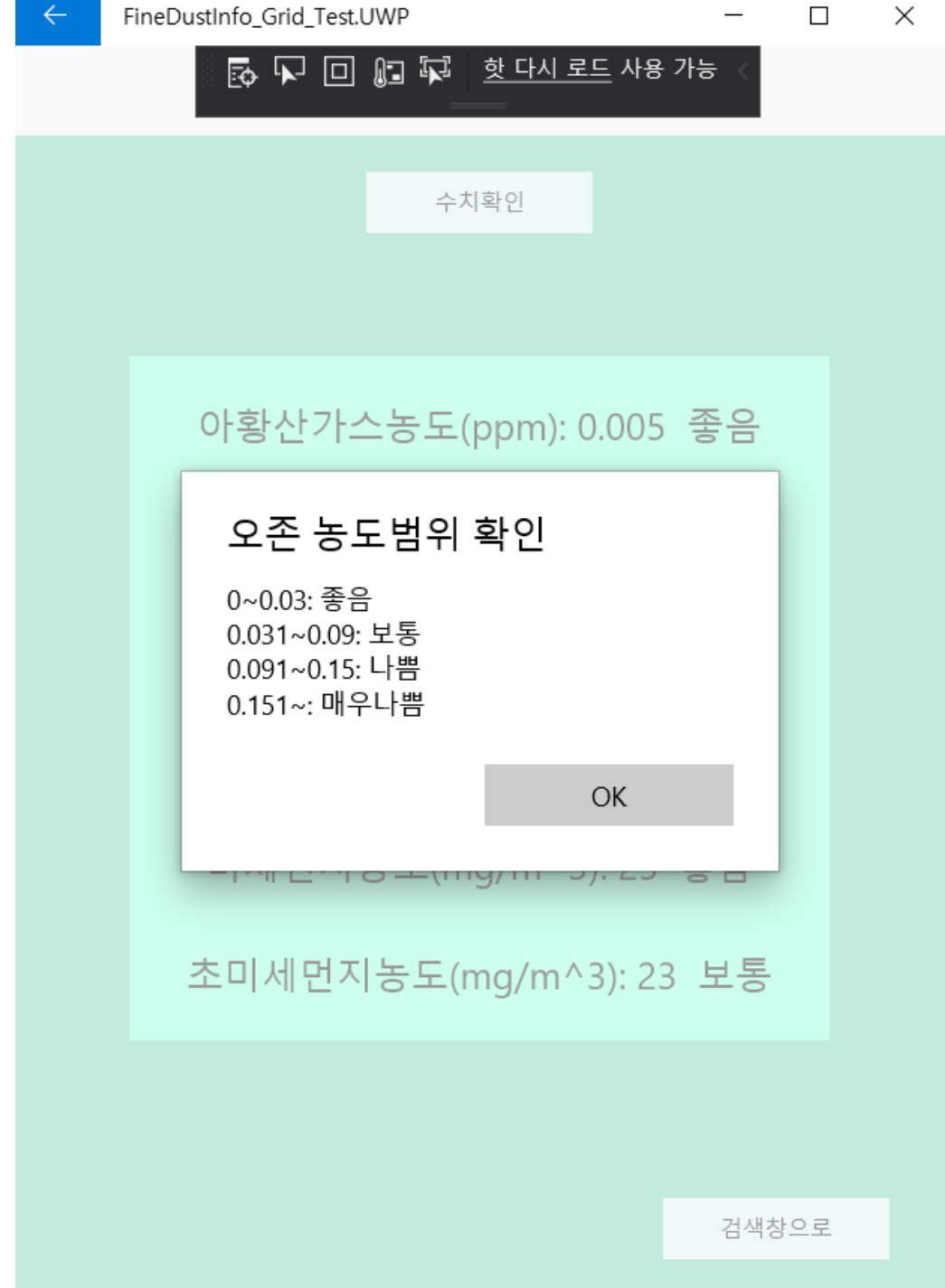
이산화질소농도(ppm): 0.042 보통

미세먼지농도(mg/m³): 25 좋음

초미세먼지농도(mg/m³): 23 보통

검색창으로

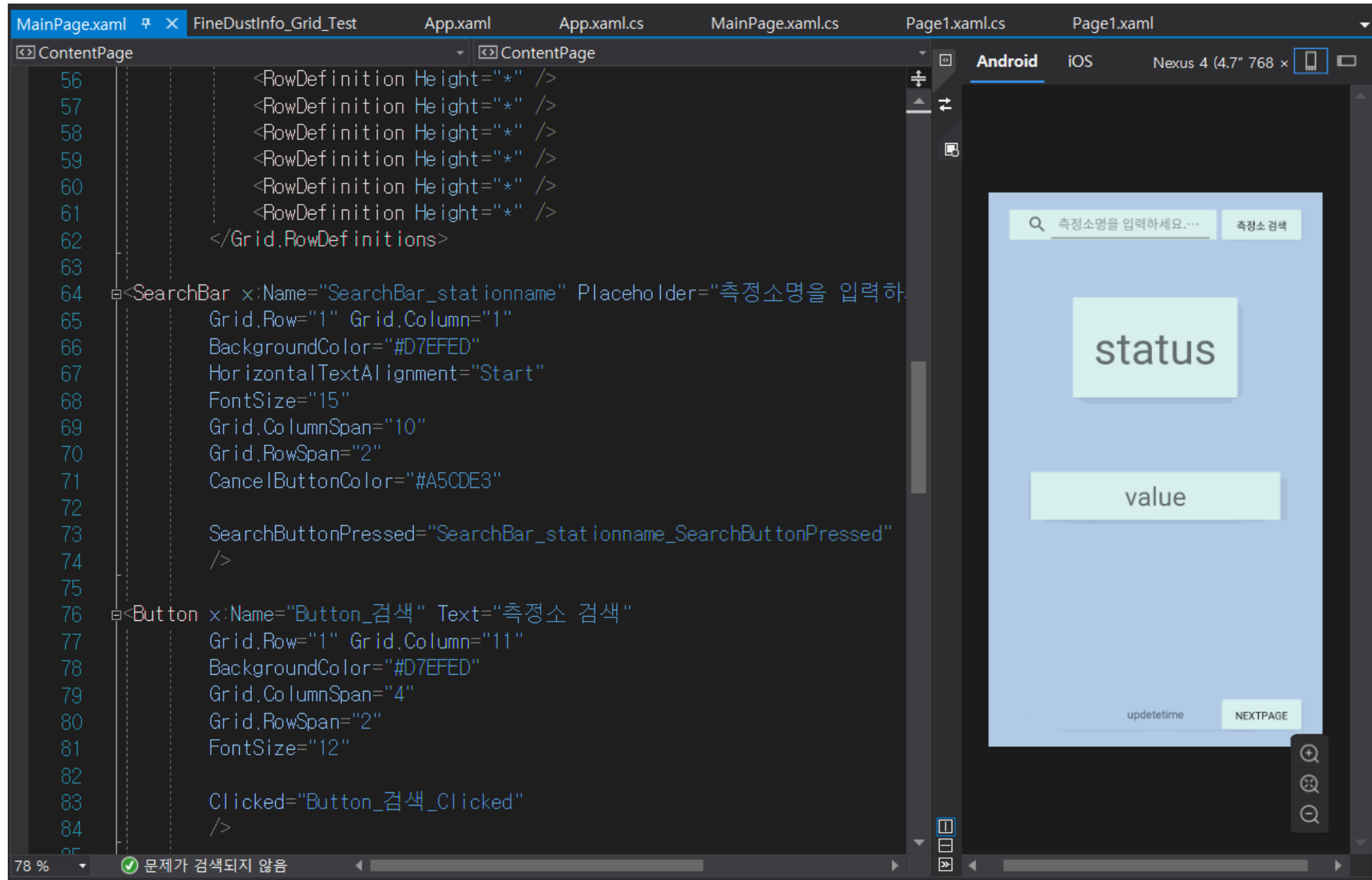
각각의 구체적인 대기오염 농도 값에
대한 상태를 구분하는
경계값을 확인할 수 있습니다.



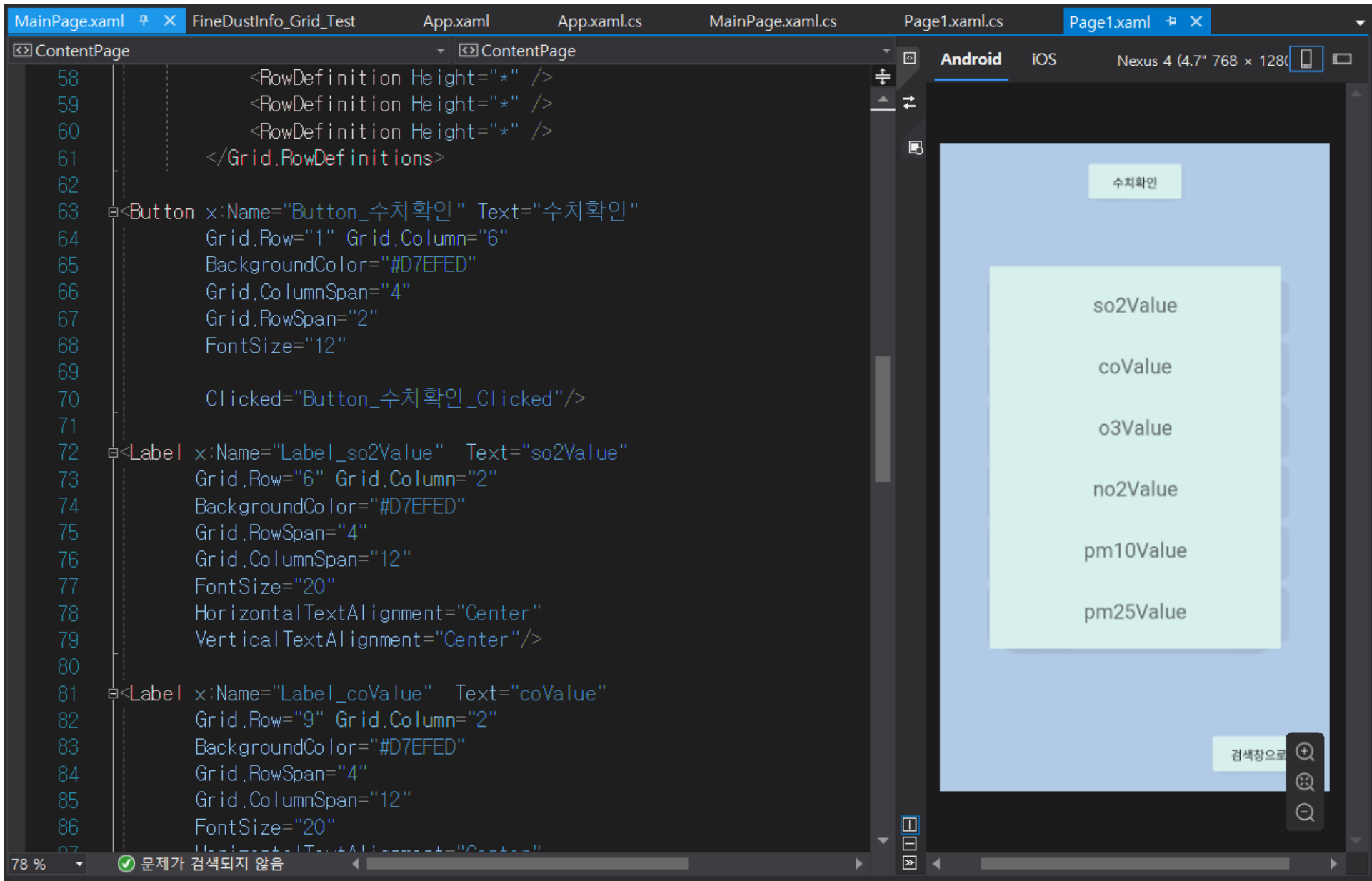
3. 코드설명

- 이상은 코드와 코드에 대한 설명입니다.

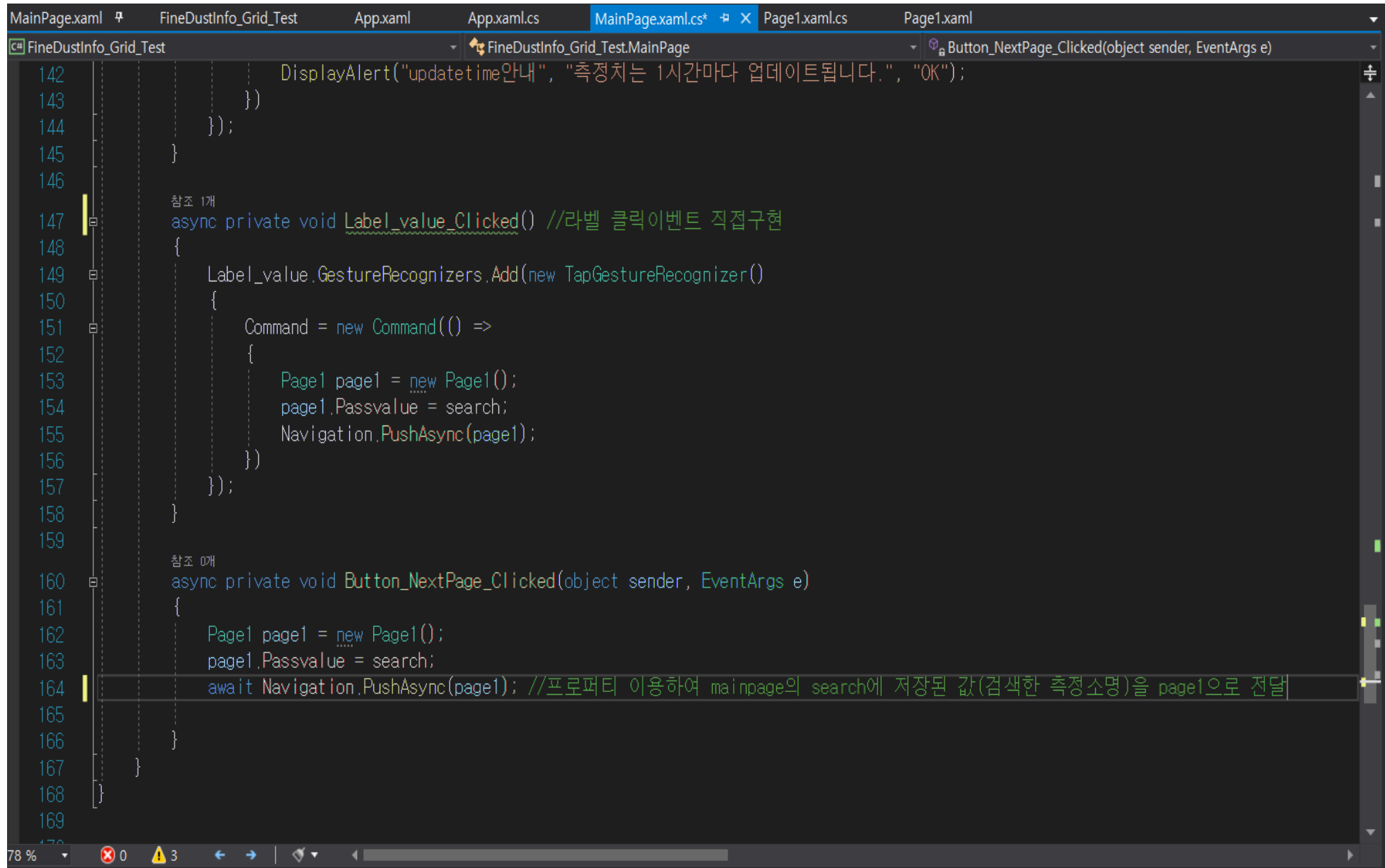
메인페이지의 디자인입니다. 열, 행을 직접 지정하여 그렸습니다.



서브페이지 디자인입니다. 세부적인 대기오염농도 정보를 제공합니다.

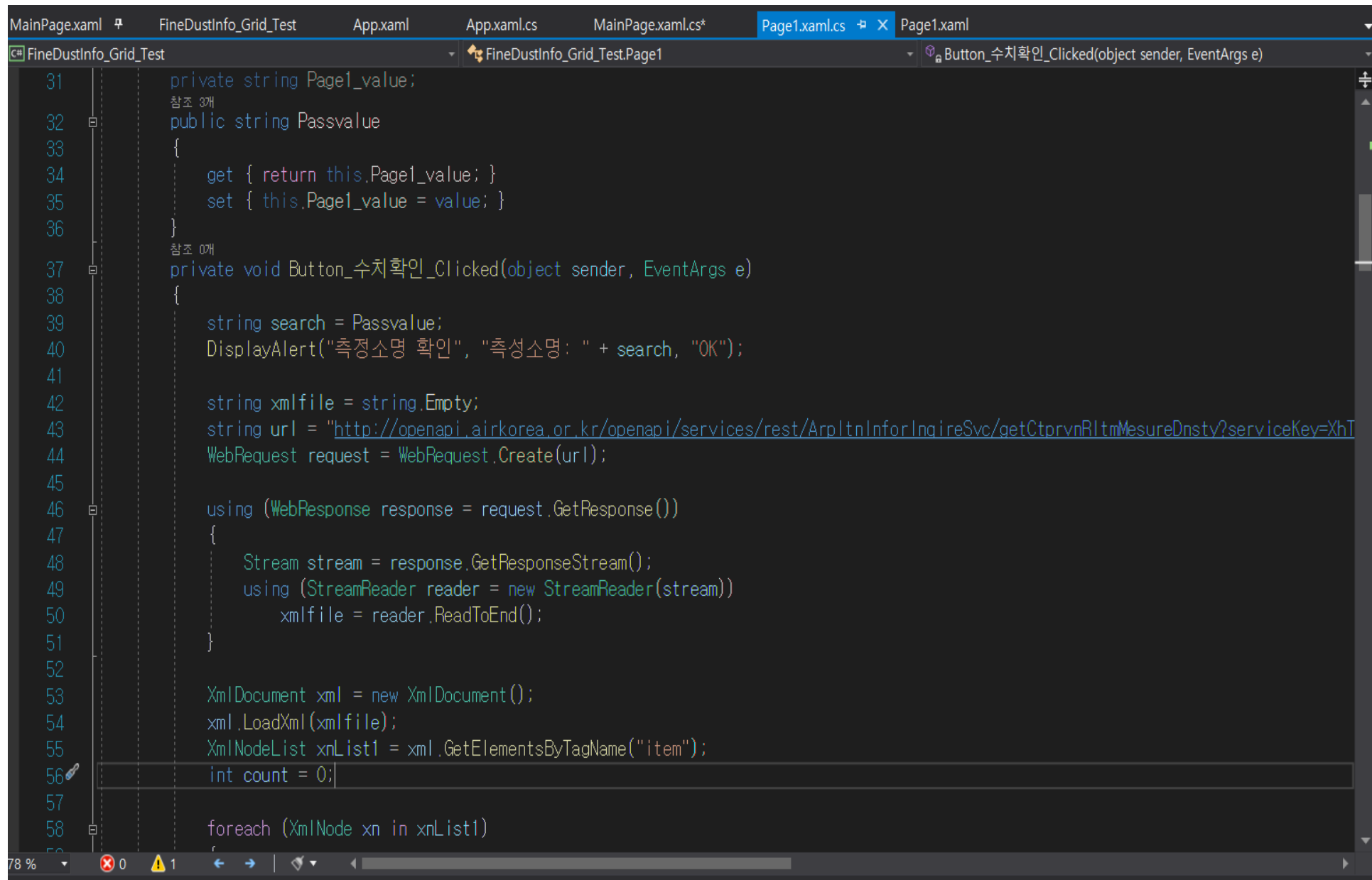


윈도우 폼의 라벨 클릭이벤트 직접 구현하였고 페이지간 변수값을 공유하는 기능을 직접 구현하였습니다.



```
MainPage.xaml  FineDustInfo_Grid_Test  App.xaml  App.xaml.cs  MainPage.xaml.cs*  Page1.xaml.cs  Page1.xaml
FineDustInfo_Grid_Test
FineDustInfo_Grid_Test.MainPage
Button_NextPage_Clicked(object sender, EventArgs e)
142      DisplayAlert("updatetime안내", "측정치는 1시간마다 업데이트됩니다.", "OK");
143  })
144  });
145  }
146
147  참조 1개
148  async private void Label_value_Clicked() //라벨 클릭이벤트 직접구현
149  {
150      Label_value.GestureRecognizers.Add(new TapGestureRecognizer()
151      {
152          Command = new Command(() =>
153          {
154              Page1 page1 = new Page1();
155              page1.Passvalue = search;
156              Navigation.PushAsync(page1);
157          })
158      });
159
160  참조 0개
161  async private void Button_NextPage_Clicked(object sender, EventArgs e)
162  {
163      Page1 page1 = new Page1();
164      page1.Passvalue = search;
165      await Navigation.PushAsync(page1); //프로퍼티 이용하여 mainpage의 search에 저장된 값(검색한 측정소명)을 page1으로 전달
166  }
167  }
168  }
169
170
78 % 0 3
```

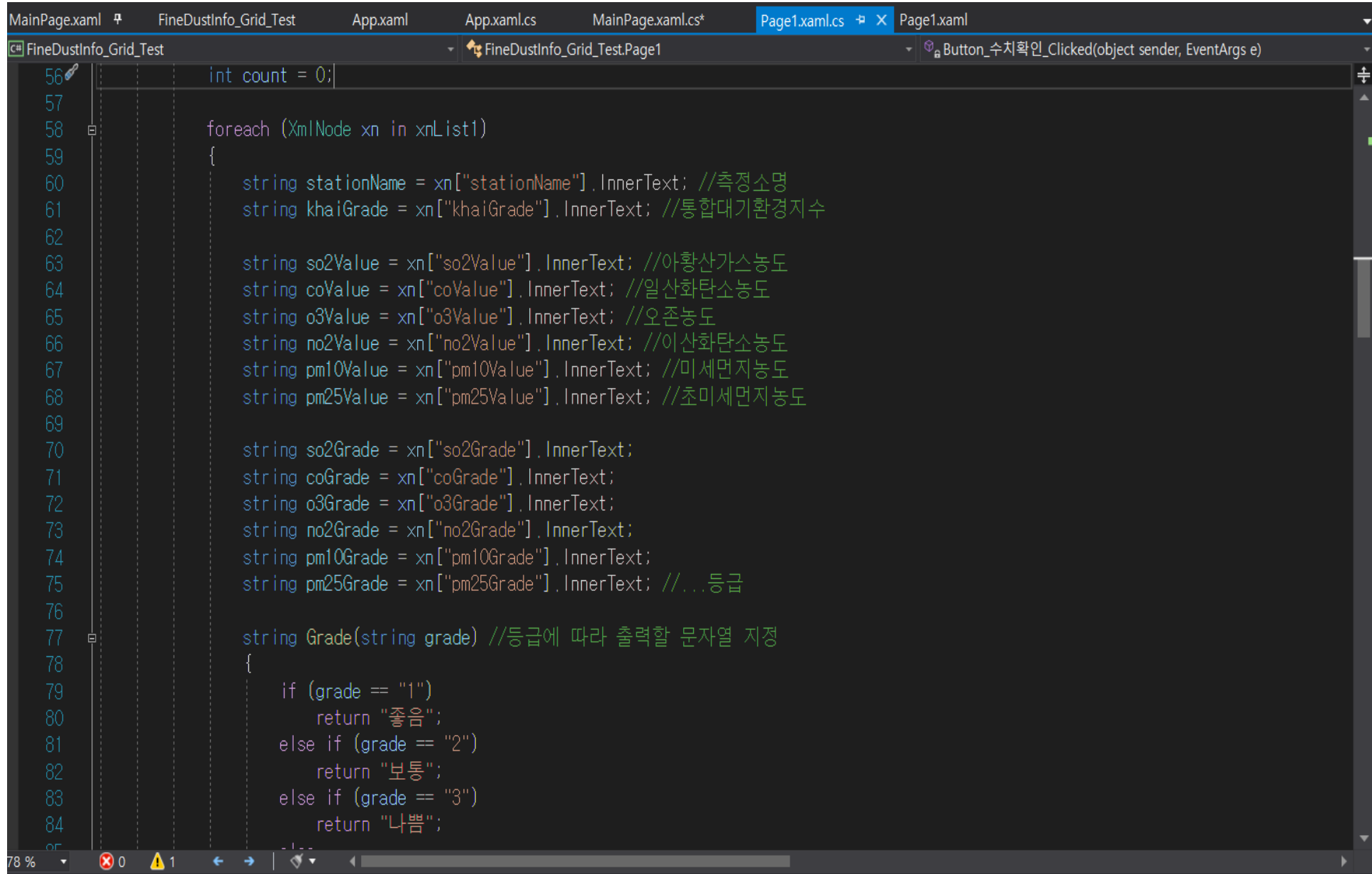
서브페이지의 api 파싱부분 코드입니다. 버튼 클릭 이벤트가 발생할 경우 동작합니다.



```
31 private string Page1_value;
32     참조 3개
33     public string Passvalue
34     {
35         get { return this.Page1_value; }
36         set { this.Page1_value = value; }
37     }
38     참조 0개
39     private void Button_수치확인_Clicked(object sender, EventArgs e)
40     {
41         string search = Passvalue;
42         DisplayAlert("측정소명 확인", "측정소명: " + search, "OK");
43
44         string xmlfile = string.Empty;
45         string url = "http://openapi.airkorea.or.kr/openapi/services/rest/ArpltnInforInquireSvc/getCtprvnRltmMesureDnsty?serviceKey=XhT";
46         WebRequest request = WebRequest.Create(url);
47
48         using (WebResponse response = request.GetResponse())
49         {
50             Stream stream = response.GetResponseStream();
51             using (StreamReader reader = new StreamReader(stream))
52             {
53                 xmlfile = reader.ReadToEnd();
54             }
55
56             XmlDocument xml = new XmlDocument();
57             xml.LoadXml(xmlfile);
58             XmlNodeList xnList1 = xml.GetElementsByTagName("item");
59             int count = 0;
60
61             foreach (XmlNode xn in xnList1)
```

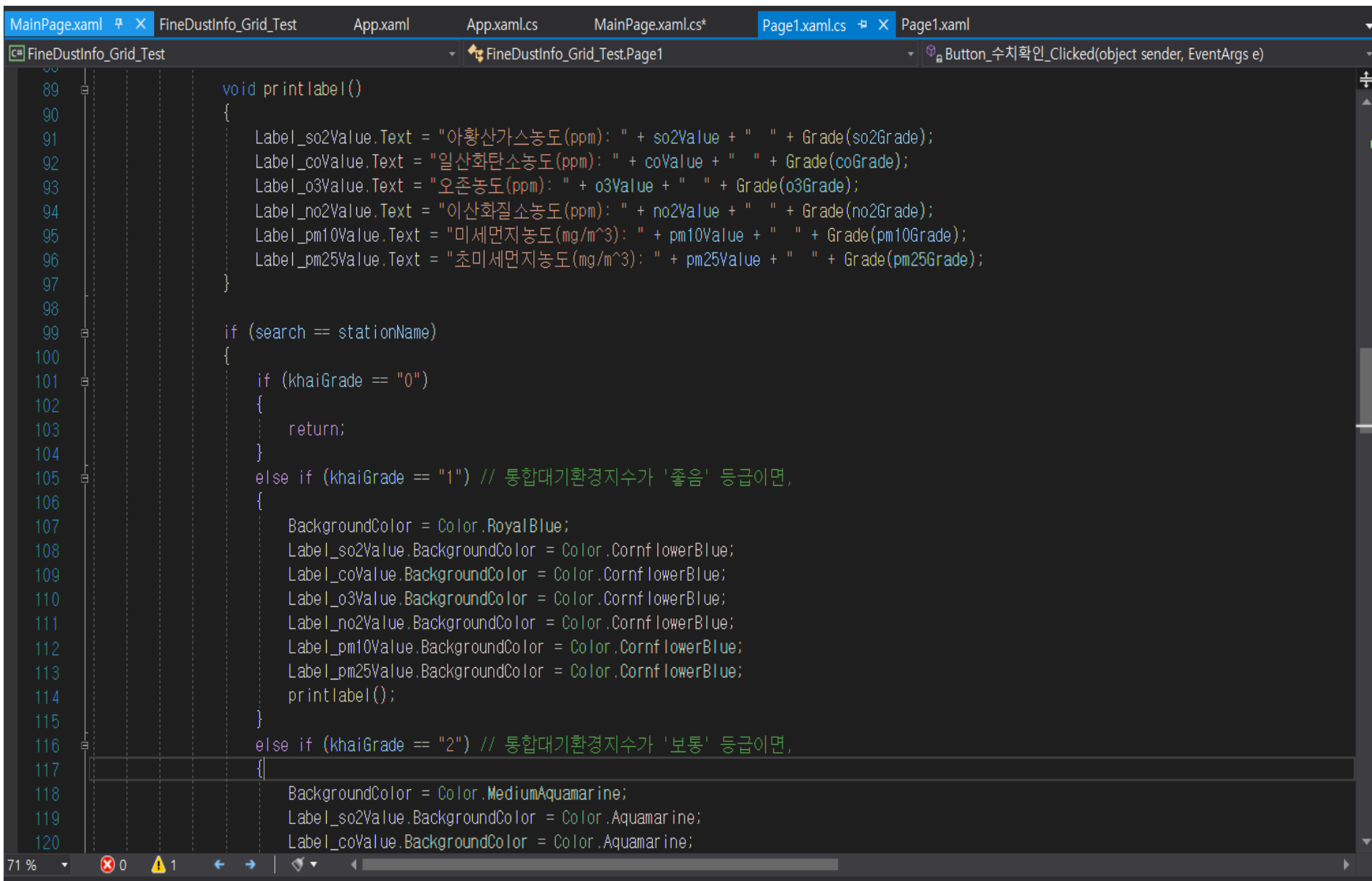
78 % 0 1

서브페이지의 api 파싱부분 코드입니다. 검색된 지역의 대기오염 농도를 가져옵니다.



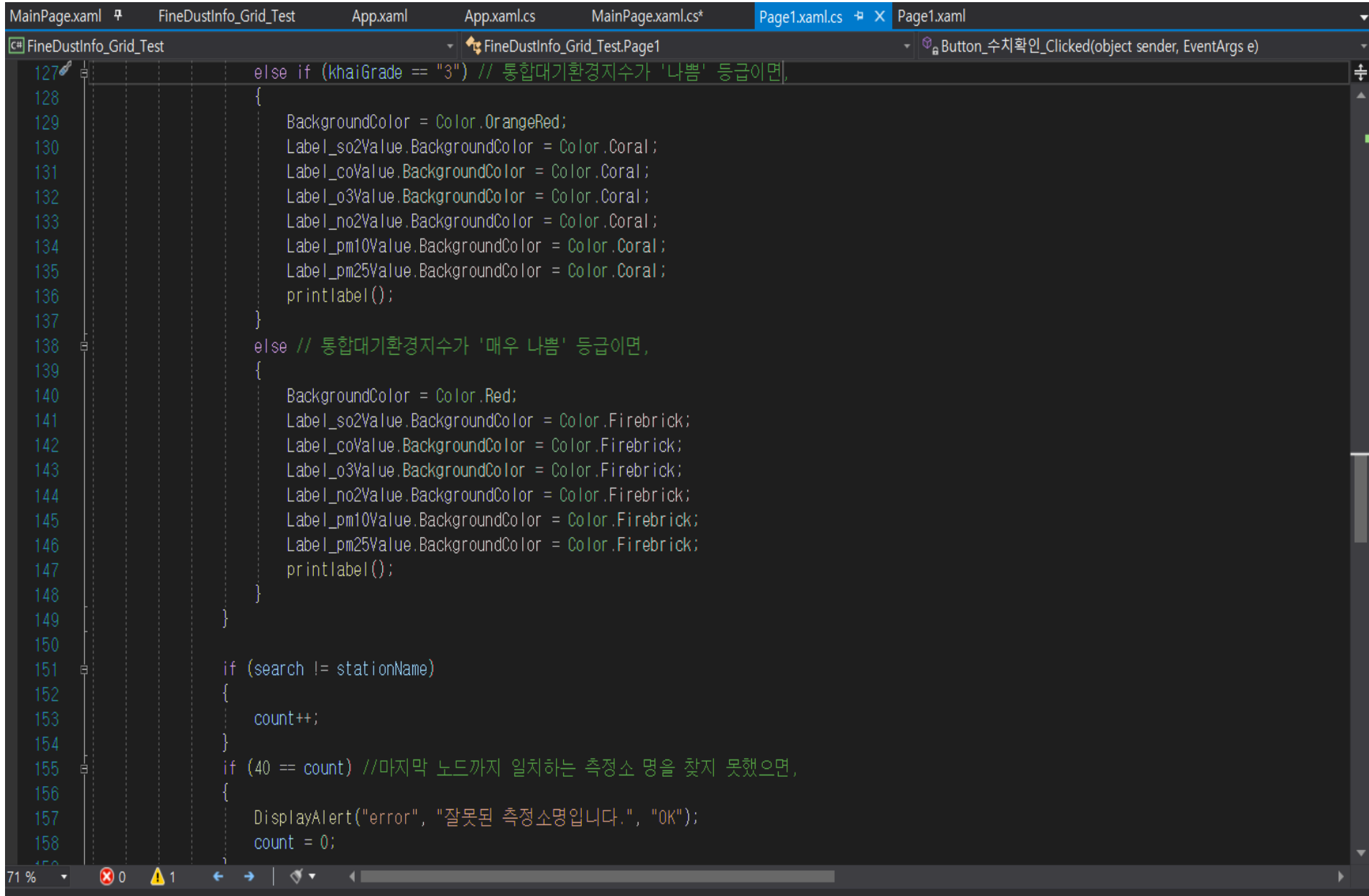
```
56 int count = 0;
57
58 foreach (XmlNode xn in xnList1)
59 {
60     string stationName = xn["stationName"].InnerText; //측정소명
61     string khaiGrade = xn["khaiGrade"].InnerText; //통합대기환경지수
62
63     string so2Value = xn["so2Value"].InnerText; //야황산가스농도
64     string coValue = xn["coValue"].InnerText; //일산화탄소농도
65     string o3Value = xn["o3Value"].InnerText; //오존농도
66     string no2Value = xn["no2Value"].InnerText; //이산화탄소농도
67     string pm10Value = xn["pm10Value"].InnerText; //미세먼지농도
68     string pm25Value = xn["pm25Value"].InnerText; //초미세먼지농도
69
70     string so2Grade = xn["so2Grade"].InnerText;
71     string coGrade = xn["coGrade"].InnerText;
72     string o3Grade = xn["o3Grade"].InnerText;
73     string no2Grade = xn["no2Grade"].InnerText;
74     string pm10Grade = xn["pm10Grade"].InnerText;
75     string pm25Grade = xn["pm25Grade"].InnerText; //...등급
76
77     string Grade(string grade) //등급에 따라 출력할 문자열 지정
78     {
79         if (grade == "1")
80             return "좋음";
81         else if (grade == "2")
82             return "보통";
83         else if (grade == "3")
84             return "나쁨";
85     }
```

메인페이지의 출력부분 코드입니다. 대기오염정도(좋음->보통->나쁨->매우나쁨)에 따라 배경색을(파랑->초록->주황->빨강) 출력합니다.



```
89 void printLabel()
90 {
91     Label_so2Value.Text = "아황산가스농도(ppm): " + so2Value + " " + Grade(so2Grade);
92     Label_coValue.Text = "일산화탄소농도(ppm): " + coValue + " " + Grade(coGrade);
93     Label_o3Value.Text = "오존농도(ppm): " + o3Value + " " + Grade(o3Grade);
94     Label_no2Value.Text = "이산화질소농도(ppm): " + no2Value + " " + Grade(no2Grade);
95     Label_pm10Value.Text = "미세먼지농도(mg/m^3): " + pm10Value + " " + Grade(pm10Grade);
96     Label_pm25Value.Text = "초미세먼지농도(mg/m^3): " + pm25Value + " " + Grade(pm25Grade);
97 }
98
99 if (search == stationName)
100 {
101     if (khaiGrade == "0")
102     {
103         return;
104     }
105     else if (khaiGrade == "1") // 통합대기환경지수가 '좋음' 등급이면,
106     {
107         BackgroundColor = Color.RoyalBlue;
108         Label_so2Value.BackgroundColor = Color.CornflowerBlue;
109         Label_coValue.BackgroundColor = Color.CornflowerBlue;
110         Label_o3Value.BackgroundColor = Color.CornflowerBlue;
111         Label_no2Value.BackgroundColor = Color.CornflowerBlue;
112         Label_pm10Value.BackgroundColor = Color.CornflowerBlue;
113         Label_pm25Value.BackgroundColor = Color.CornflowerBlue;
114         printLabel();
115     }
116     else if (khaiGrade == "2") // 통합대기환경지수가 '보통' 등급이면,
117     {
118         BackgroundColor = Color.MediumAquamarine;
119         Label_so2Value.BackgroundColor = Color.Aquamarine;
120         Label_coValue.BackgroundColor = Color.Aquamarine;
```

메인페이지의 출력부분 코드입니다. 측정소를 찾지 못하면 오류 팝업을 띄웁니다.



```
127 else if (khaiGrade == "3") // 통합대기환경지수가 '나쁨' 등급이면,
128 {
129     BackgroundColor = Color.OrangeRed;
130     Label_so2Value.BackgroundColor = Color.Coral;
131     Label_coValue.BackgroundColor = Color.Coral;
132     Label_o3Value.BackgroundColor = Color.Coral;
133     Label_no2Value.BackgroundColor = Color.Coral;
134     Label_pm10Value.BackgroundColor = Color.Coral;
135     Label_pm25Value.BackgroundColor = Color.Coral;
136     printlabel();
137 }
138 else // 통합대기환경지수가 '매우 나쁨' 등급이면,
139 {
140     BackgroundColor = Color.Red;
141     Label_so2Value.BackgroundColor = Color.Firebrick;
142     Label_coValue.BackgroundColor = Color.Firebrick;
143     Label_o3Value.BackgroundColor = Color.Firebrick;
144     Label_no2Value.BackgroundColor = Color.Firebrick;
145     Label_pm10Value.BackgroundColor = Color.Firebrick;
146     Label_pm25Value.BackgroundColor = Color.Firebrick;
147     printlabel();
148 }
149 }
150
151 if (search != stationName)
152 {
153     count++;
154 }
155 if (40 == count) //마지막 노드까지 일치하는 측정소 명을 찾지 못했으면,
156 {
157     DisplayAlert("error", "잘못된 측정소명입니다.", "OK");
158     count = 0;
159 }
```

감사합니다.