기상청 생활기상지수(3.0) 조회서비스



Open API 활용 코드 상세 설명서

목 차

[1) [자외선지수조회]상세기능명세](#_Toc35612262)

[2) [대기정체지수조회]상세기능명세](#_Toc35612263)

[3) [체감온도(여름철)조회]상세기능명세](#_Toc35612264)

**1. 서비스 명세**

**1.1 생활기상지수 조회서비스**

가. 상세기능 목록

|  |  |  |  |
| --- | --- | --- | --- |
| **번호** | **API명(국문)** | **상세기능명(영문)** | **상세기능명(국문)** |
| 1 | 생활기상지수 조회서비스 | getUVIdxV4 | 자외선지수조회 |
| 2 | getAirDiffusionIdxV4 | 대기정체지수조회 |
| 3 | getSenTaIdxV4 | 체감온도(여름철)조회 |

나. 상세기능내역

1) [자외선지수조회] 상세기능명세

a) 상세기능정보

|  |  |  |  |
| --- | --- | --- | --- |
| **상세기능 번호** | 2 | **상세기능 유형** | 조회 (목록) |
| **상세기능명(국문)** | 자외선지수조회 | | |
| **상세기능 설명** | 지점코드, 발표시간 부터 3시간 단위로 최대75시간 후 까지의 예측값을 조회하는 기능  생산주기: 일 8회(3시간 간격/ 00시, 03시, 06시…….)  제공기간: 연중  각 단계 및 범위는 상단의 <표> “자외선지수 단계 및 범위” 참고 / 지수자료가 6일 경우 높음 값 표출 | | |
| **Call Back URL** | http://apis.data.go.kr/1360000/LivingWthrIdxServiceV4/getUVIdxV4 | | |
| **최대 메시지 사이즈** | [3.7] Kbyte | | |
| **평균 응답 시간** | [312] ms | **초당 최대 트랙잭션** | [45] tps |

# 자료제공 기간 : 연중 (최근 발표자료로 제한)

#자외선지수 단계 및 범위

|  |  |
| --- | --- |
| **단계** | **지수범위** |
| 위험 | 11 이상 |
| 매우높음 | 8~10 |
| 높음 | 6~7 |
| 보통 | 3~5 |
| 낮음 | 0~2 |

# 2023년 6월 14일 06시에 2023년6월14일 06시 자료 호출시 첫번째 데이터는 06시~09데이터 /

# 데이터는 3시간 구간값(06시~09시 /09시~12시/12시~15시/15시~18시), 각 단계 및 범위는 상단의 <표> “자외선지수 단계 및 범위” 참고 / 지수자료가 9일 경우 매우높음 8~10 사이값으로 매우높음 값 표출

e) 그 외 활용 예제코드

|  |
| --- |
| **Java Code** |
| /\* Java 샘플 코드 \*/  import java.io.InputStreamReader;  import java.net.HttpURLConnection;  import java.net.URL;  import java.net.URLEncoder;  import java.io.BufferedReader;  import java.io.IOException;  public class ApiExplorer  {  public static void main(String[] args) throws IOException  {  StringBuilder urlBuilder = new StringBuilder("http://apis.data.go.kr/1360000/LivingWthrIdxServiceV4/getUVIdxV4"); /\*URL\*/  urlBuilder.append("?" + URLEncoder.encode("ServiceKey","UTF-8") + "=서비스키"); /\*Service Key\*/  urlBuilder.append("&" + URLEncoder.encode("serviceKey","UTF-8") + "=" + URLEncoder.encode("-", "UTF-8")); /\*공공데이터포털에서 받은 인증키\*/  urlBuilder.append("&" + URLEncoder.encode("areaNo","UTF-8") + "=" + URLEncoder.encode("1100000000", "UTF-8")); /\*서울지점\*/  urlBuilder.append("&" + URLEncoder.encode("time","UTF-8") + "=" + URLEncoder.encode("2022071118", "UTF-8")); /\*2022년7월11일18시\*/  urlBuilder.append("&" + URLEncoder.encode("dataType","UTF-8") + "=" + URLEncoder.encode("xml", "UTF-8")); /\*xml, json 선택(미입력시 xml)\*/  URL url = new URL(urlBuilder.toString());  HttpURLConnection conn = (HttpURLConnection) url.openConnection();  conn.setRequestMethod("GET");  conn.setRequestProperty("Content-type", "application/json");  System.out.println("Response code: " + conn.getResponseCode());  BufferedReader rd;  if(conn.getResponseCode() >= 200 && conn.getResponseCode() <= 300)  {  rd = new BufferedReader(new InputStreamReader(conn.getInputStream()));  } else  {  rd = new BufferedReader(new InputStreamReader(conn.getErrorStream()));  }  StringBuilder sb = new StringBuilder();  String line;  while ((line = rd.readLine()) != null)  {  sb.append(line);  }  rd.close();  conn.disconnect();  System.out.println(sb.toString());  }  } |
| **Javascript Code** |
| /\* Javascript 샘플 코드 \*/  var xhr = new XMLHttpRequest();  var url = ‘http://apis.data.go.kr/1360000/LivingWthrIdxServiceV4/getUVIdxV4’; /\*URL\*/  var queryParams = '?' + encodeURIComponent('ServiceKey') + '='+'서비스키'; /\*Service Key\*/  queryParams += '&' + encodeURIComponent('areaNo') + '=' + encodeURIComponent('1100000000'); /\*서울지점\*/  queryParams += '&' + encodeURIComponent('time') + '=' + encodeURIComponent('2022071118'); /\*2022년7월11일18시\*/  queryParams += '&' + encodeURIComponent('dataType') + '=' + encodeURIComponent('xml'); /\*xml, json 선택(미입력시 xml)\*/  xhr.open('GET', url + queryParams);  xhr.send('');  xhr.onreadystatechange = function ()  {  if (this.readyState == 4)  {  alert('Status: '+this.status+' Headers: '+JSON.stringify(this.getAllResponseHeaders())+' Body: '+this.responseText);  }  }; |
| **PHP Code** |
| /\* PHP 샘플 코드 \*/  $ch = curl\_init();  $url = 'http://apis.data.go.kr/1360000/LivingWthrIdxServiceV4/getUVIdxV4'; /\*URL\*/  $queryParams = '?' . urlencode('ServiceKey') . '=서비스키'; /\*Service Key\*/  $queryParams .= '&' . urlencode('serviceKey') . '=' . urlencode('-'); /\*공공데이터포털에서 받은 인증키\*/  $queryParams .= '&' . urlencode('areaNo') . '=' . urlencode('1100000000'); /\*서울지점\*/  $queryParams .= '&' . urlencode('time') . '=' . urlencode('2022071118'); /\*2022년7월11일18시\*/  $queryParams .= '&' . urlencode('dataType') . '=' . urlencode('xml'); /\*xml, json 선택(미입력시 xml)\*/  curl\_setopt($ch, CURLOPT\_URL, $url . $queryParams);  curl\_setopt($ch, CURLOPT\_RETURNTRANSFER, TRUE);  curl\_setopt($ch, CURLOPT\_HEADER, FALSE);  curl\_setopt($ch, CURLOPT\_CUSTOMREQUEST, 'GET');  $response = curl\_exec($ch);  curl\_close($ch);  var\_dump($response); |
| **C++ Code** |
| /\* C++ 샘플 코드 \*/  using System;  using System.Net;  using System.Net.Http;  using System.IO;  namespace ConsoleApp1  {  class Program  {  static HttpClient client = new HttpClient();  static void Main(string[] args)  {  string url = "http://apis.data.go.kr/1360000/LivingWthrIdxServiceV4/getUVIdxV4"; // URL  url += "?ServiceKey=" + "서비스키"; // Service Key  url += "&serviceKey=-";  url += "&areaNo=1100000000";  url += "&time=2022071118";  url += "&dataType=xml";  var request = (HttpWebRequest)WebRequest.Create(url);  request.Method = "GET";  string results = string.Empty;  HttpWebResponse response;  using (response = request.GetResponse() as HttpWebResponse)  {  StreamReader reader = new StreamReader(response.GetResponseStream());  results = reader.ReadToEnd();  }  Console.WriteLine(results);  }  }  } |
| **Curl Code** |
| /\* cURL 샘플 코드 \*/  curl --include --request GET \  ‘http://apis.data.go.kr/1360000/LivingWthrIdxServiceV4/getUVIdxV4?serviceKey=인증키(URL Encode)&numOfRows=10&pageNo=1&areaNo=1100000000&time=2022071118&dataType=xml' |

|  |
| --- |
| **Object C Code** |
| /\* Objective-C 샘플 코드 \*/  #import <Foundation/Foundation.h>  #import <CoreFoundation/CoreFoundation.h>  @implementation NSString (URLEncoding)  //function used to encode query and matrix parameters  -(NSString \*)urlEncodeUsingEncoding:(NSStringEncoding)encoding  {  return (\_\_bridge NSString \*)CFURLCreateStringByAddingPercentEscapes(NULL,  (CFStringRef)self,  NULL,  (CFStringRef)@"!\*'\\"();:@&=+$,/?%#[]% ",  CFStringConvertNSStringEncodingToEncoding(encoding));  }  @end  int main (int argc, const char \* argv[])  {  @autoreleasepool  {  NSMutableURLRequest \*request = [[NSMutableURLRequest alloc] init];  NSString \*endpoint = [NSString stringWithFormat:@ "http://apis.data.go.kr/1360000/LivingWthrIdxServiceV4/getUVIdxV4"]; /\*URL\*/  NSString \*queryParams = [NSString stringWithFormat:@"?%@&%@&%@&%@&%@",  [NSString stringWithFormat:@"%@=%@", [@"ServiceKey" urlEncodeUsingEncoding:NSUTF8StringEncoding], [@"서비스키" urlEncodeUsingEncoding:NSUTF8StringEncoding]]  , [NSString stringWithFormat:@"%@=%@", [@"serviceKey" urlEncodeUsingEncoding:NSUTF8StringEncoding], [@"-" urlEncodeUsingEncoding:NSUTF8StringEncoding]]  , [NSString stringWithFormat:@"%@=%@", [@"areaNo" urlEncodeUsingEncoding:NSUTF8StringEncoding], [@"1100000000" urlEncodeUsingEncoding:NSUTF8StringEncoding]]  , [NSString stringWithFormat:@"%@=%@", [@"time" urlEncodeUsingEncoding:NSUTF8StringEncoding], [@"2022071118" urlEncodeUsingEncoding:NSUTF8StringEncoding]]  , [NSString stringWithFormat:@"%@=%@", [@"dataType" urlEncodeUsingEncoding:NSUTF8StringEncoding], [@"xml" urlEncodeUsingEncoding:NSUTF8StringEncoding]]];  [request setURL:[NSURL URLWithString:[NSString stringWithFormat:@"%@%@",endpoint,queryParams]]];  [request setHTTPMethod:@"GET"];  NSHTTPURLResponse \*urlResponse = nil;  NSError \*error = [[NSError alloc] init];  NSData \*responseData = [NSURLConnection sendSynchronousRequest:request returningResponse:&urlResponse error:&error];  NSString \*result = [[NSString alloc] initWithData:responseData encoding:NSUTF8StringEncoding];  NSLog(@"Response Code: %d", [urlResponse statusCode]);  if ([urlResponse statusCode] >= 200 && [urlResponse statusCode] < 300)  {  NSLog(@"Response: %@", result);  }  return 0;  }  } |
| **Ruby Code** |
| /\* Ruby 샘플 코드 \*/  require 'rubygems'  require 'rest\_client'  require 'cgi'  url = 'http://apis.data.go.kr/1360000/LivingWthrIdxServiceV4/getUVIdxV4'  headers = { :params => { CGI::escape('ServiceKey') => '서비스키',CGI::escape('serviceKey') => '-',CGI::escape('areaNo') => '1100000000',CGI::escape('time') => '2022071118',CGI::escape('dataType') => 'xml' } }  response = RestClient::Request.execute :method => 'GET', :url => url , :headers => headers  puts response |
| **Python Code** |
| /\* Python 샘플 코드 \*/  from urllib2 import Request, urlopen  from urllib import urlencode, quote\_plus  url = 'http://apis.data.go.kr/1360000/LivingWthrIdxServiceV4/getUVIdxV4'  queryParams = '?' + urlencode({ quote\_plus('ServiceKey') : '서비스키', quote\_plus('serviceKey') : '-', quote\_plus('areaNo') : '1100000000', quote\_plus('time') : '2022071118', quote\_plus('dataType') : 'xml' })  request = Request(url + queryParams)  request.get\_method = lambda: 'GET'  response\_body = urlopen(request).read()  print response\_body |
| **Node.js Code** |
| /\* NodeJs 샘플 코드 \*/  var request = require('request');  var url = 'http://apis.data.go.kr/1360000/LivingWthrIdxServiceV4/getUVIdxV4';  var queryParams = '?' + encodeURIComponent('ServiceKey') + '=서비스키'; /\* Service Key\*/  queryParams += '&' + encodeURIComponent('serviceKey') + '=' + encodeURIComponent('-'); /\* 공공데이터포털에서 받은 인증키 \*/  queryParams += '&' + encodeURIComponent('areaNo') + '=' + encodeURIComponent('1100000000'); /\* 서울지점 \*/  queryParams += '&' + encodeURIComponent('time') + '=' + encodeURIComponent('2022071118'); /\* 2022년7월11일18시 \*/  queryParams += '&' + encodeURIComponent('dataType') + '=' + encodeURIComponent('xml'); /\* xml, json 선택(미입력시 xml) \*/  request(  {  url: url + queryParams,  method: 'GET'  }, function (error, response, body)  {  //console.log('Status', response.statusCode);  //console.log('Headers', JSON.stringify(response.headers));  //console.log('Reponse received', body);  }); |
| **Kotlin Code** |
| /\* Kotlin 예제 \*/  import java.io.BufferedReader  import java.io.InputStreamReader  import java.net.HttpURLConnection  import java.net.URL  import java.net.URLEncoder  object OpenAPI  {  fun main(args: Array<String>)  {  val urlBuilder = StringBuilder("http://apis.data.go.kr/1360000/LivingWthrIdxServiceV4/getUVIdxV4") /\* URL \*/  urlBuilder.append("?" + URLEncoder.encode("ServiceKey", "UTF-8") + "=서비스키") /\* Service Key \*/  urlBuilder.append("&" + URLEncoder.encode("ServiceKey", "UTF-8") + "=" + URLEncoder.encode("-", "UTF-8")) /\* 공공데이터포털에서 받은 인증키 \*/  urlBuilder.append("&" + URLEncoder.encode("areaNo", "UTF-8") + "=" + URLEncoder.encode("1100000000", "UTF-8")) /\* 지점코드 \*/  urlBuilder.append("&" + URLEncoder.encode("time", "UTF-8") + "=" + URLEncoder.encode("2022071118", "UTF-8")) /\* 2022년7월11일18시 \*/  urlBuilder.append("&" + URLEncoder.encode("dataType", "UTF-8") + "=" + URLEncoder.encode("xml", "UTF-8")) /\* xml, json (미입력시 xml) \*/  val url = URL(urlBuilder.toString())  val conn = url.openConnection() as HttpURLConnection  conn.requestMethod = "GET"  conn.setRequestProperty("Content-type", "application/json")  System.out.println("Response code: " + conn.responseCode)  val br: BufferedReader  br = if (conn.responseCode >= 200 && conn.responseCode <= 300)  {  BufferedReader(InputStreamReader(conn.inputStream))  } else  {  BufferedReader(InputStreamReader(conn.errorStream))  }  val sb = StringBuilder()  var line: String?  Do  {  line = br.readLine()  sb.append(line)  } while (line != null)  br.close()  conn.disconnect()  System.out.println(sb.toString())  }  } |

2) [대기정체지수조회] 상세기능명세

a) 상세기능정보

|  |  |  |  |
| --- | --- | --- | --- |
| **상세기능 번호** | 3 | **상세기능 유형** | 조회 (목록) |
| **상세기능명(국문)** | 대기정체지수조회 | | |
| **상세기능 설명** | 지점코드, 발표시간부터 3시간 단위로 78시간 후 까지의 예측값을 조회하는 기능  생산주기: 일 8회(3시간 간격/ 00시, 03시, 06시…….)  제공기간: 연중  각 단계 및 범위는 상단의 <표> “대기정채지수 단계 및 범위” 참고 / 지수자료가 75일 경우 보통 값 표출 | | |
| **Call Back URL** | http://apis.data.go.kr/1360000/LivingWthrIdxServiceV4/getAirDiffusionIdxV4 | | |
| **최대 메시지 사이즈** | [4.7] Kbyte | | |
| **평균 응답 시간** | [305] ms | **초당 최대 트랙잭션** | [30] tps |

# 자료제공 기간 : 연중 (최근 발표자료로 제한)

# 대기정체지수 단계 및 범위

|  |  |
| --- | --- |
| **지수단계** | **데이터 조건** |
| 낮음 | 25 |
| 보통 | 50 |
| 높음 | 75 |
| 매우높음 | 100 |

e) 그 외 활용 예제코드

|  |
| --- |
| **Java Code** |
| /\* Java 샘플 코드 \*/  import java.io.InputStreamReader;  import java.net.HttpURLConnection;  import java.net.URL;  import java.net.URLEncoder;  import java.io.BufferedReader;  import java.io.IOException;  public class ApiExplorer {  public static void main(String[] args) throws IOException {  StringBuilder urlBuilder = new StringBuilder("http://apis.data.go.kr/1360000/LivingWthrIdxServiceV4/getAirDiffusionIdxV4"); /\*URL\*/  urlBuilder.append("?" + URLEncoder.encode("ServiceKey","UTF-8") + "=서비스키"); /\*Service Key\*/  urlBuilder.append("&" + URLEncoder.encode("serviceKey","UTF-8") + "=" + URLEncoder.encode("-", "UTF-8")); /\*공공데이터포털에서 받은 인증키\*/  urlBuilder.append("&" + URLEncoder.encode("areaNo","UTF-8") + "=" + URLEncoder.encode("1100000000", "UTF-8")); /\*서울지점\*/  urlBuilder.append("&" + URLEncoder.encode("time","UTF-8") + "=" + URLEncoder.encode("2021070618", "UTF-8")); /\*2021년7월6일18시\*/  urlBuilder.append("&" + URLEncoder.encode("dataType","UTF-8") + "=" + URLEncoder.encode("xml", "UTF-8")); /\*xml, json 선택(미입력시 xml)\*/  URL url = new URL(urlBuilder.toString());  HttpURLConnection conn = (HttpURLConnection) url.openConnection();  conn.setRequestMethod("GET");  conn.setRequestProperty("Content-type", "application/json");  System.out.println("Response code: " + conn.getResponseCode());  BufferedReader rd;  if(conn.getResponseCode() >= 200 && conn.getResponseCode() <= 300) {  rd = new BufferedReader(new InputStreamReader(conn.getInputStream()));  } else {  rd = new BufferedReader(new InputStreamReader(conn.getErrorStream()));  }  StringBuilder sb = new StringBuilder();  String line;  while ((line = rd.readLine()) != null) {  sb.append(line);  }  rd.close();  conn.disconnect();  System.out.println(sb.toString());  }  } |
| **Javascript Code** |
| /\* Javascript 샘플 코드 \*/  var xhr = new XMLHttpRequest();  var url = ‘http://apis.data.go.kr/1360000/LivingWthrIdxServiceV4/getAirDiffusionIdxV4’; /\*URL\*/  var queryParams = '?' + encodeURIComponent('ServiceKey') + '='+'서비스키'; /\*Service Key\*/  queryParams += '&' + encodeURIComponent('areaNo') + '=' + encodeURIComponent('1100000000'); /\*서울지점\*/  queryParams += '&' + encodeURIComponent('time') + '=' + encodeURIComponent('2021070618'); /\*2021년7월6일18시\*/  queryParams += '&' + encodeURIComponent('dataType') + '=' + encodeURIComponent('xml'); /\*xml, json 선택(미입력시 xml)\*/  xhr.open('GET', url + queryParams);  xhr.send('');  xhr.onreadystatechange = function () {  if (this.readyState == 4) {  alert('Status: '+this.status+' Headers: '+JSON.stringify(this.getAllResponseHeaders())+' Body: '+this.responseText);  }  }; |
| **PHP Code** |
| /\* PHP 샘플 코드 \*/  $ch = curl\_init();  $url = ‘http://apis.data.go.kr/1360000/LivingWthrIdxServiceV4/getAirDiffusionIdxV4’; /\*URL\*/  $queryParams = '?' . urlencode('ServiceKey') . '=서비스키'; /\*Service Key\*/  $queryParams .= '&' . urlencode('serviceKey') . '=' . urlencode('-'); /\*공공데이터포털에서 받은 인증키\*/  $queryParams .= '&' . urlencode('areaNo') . '=' . urlencode('1100000000'); /\*서울지점\*/  $queryParams .= '&' . urlencode('time') . '=' . urlencode('2021070618'); /\*2021년7월6일18시\*/  $queryParams .= '&' . urlencode('dataType') . '=' . urlencode('xml'); /\*xml, json 선택(미입력시 xml)\*/  curl\_setopt($ch, CURLOPT\_URL, $url . $queryParams);  curl\_setopt($ch, CURLOPT\_RETURNTRANSFER, TRUE);  curl\_setopt($ch, CURLOPT\_HEADER, FALSE);  curl\_setopt($ch, CURLOPT\_CUSTOMREQUEST, 'GET');  $response = curl\_exec($ch);  curl\_close($ch);  var\_dump($response); |
| **C++ Code** |
| /\* C++ 샘플 코드 \*/  using System;  using System.Net;  using System.Net.Http;  using System.IO;  namespace ConsoleApp1  {  class Program  {  static HttpClient client = new HttpClient();  static void Main(string[] args)  {  string url = "http://apis.data.go.kr/1360000/LivingWthrIdxServiceV4/getAirDiffusionIdxV4";  // URL  url += "?ServiceKey=" + "서비스키"; // Service Key  url += "&serviceKey=-";  url += "&areaNo=1100000000";  url += "&time=2021070618";  url += "&dataType=xml";  var request = (HttpWebRequest)WebRequest.Create(url);  request.Method = "GET";  string results = string.Empty;  HttpWebResponse response;  using (response = request.GetResponse() as HttpWebResponse)  {  StreamReader reader = new StreamReader(response.GetResponseStream());  results = reader.ReadToEnd();  }  Console.WriteLine(results);  }  }  } |
| **Curl Code** |
| /\* cURL 샘플 코드 \*/  curl --include --request GET \  ‘http://apis.data.go.kr/1360000/LivingWthrIdxServiceV4/getAirDiffusionIdxV4?serviceKey=인증키(URL Encode)&numOfRows=10&pageNo=1&areaNo=1100000000&time=2021070618&dataType=xml' |
| **Object C Code** |
| /\* Objective-C 샘플 코드 \*/  #import <Foundation/Foundation.h>  #import <CoreFoundation/CoreFoundation.h>  @implementation NSString (URLEncoding)  //function used to encode query and matrix parameters  -(NSString \*)urlEncodeUsingEncoding:(NSStringEncoding)encoding  {  return (\_\_bridge NSString \*)CFURLCreateStringByAddingPercentEscapes(NULL,  (CFStringRef)self,  NULL,  (CFStringRef)@"!\*'\\"();:@&=+$,/?%#[]% ",  CFStringConvertNSStringEncodingToEncoding(encoding));  }  @end  int main (int argc, const char \* argv[])  {  @autoreleasepool  {  NSMutableURLRequest \*request = [[NSMutableURLRequest alloc] init];  NSString \*endpoint = [NSString stringWithFormat:@" http://apis.data.go.kr/1360000/LivingWthrIdxServiceV4/getAirDiffusionIdxV4"]; /\*URL\*/  NSString \*queryParams = [NSString stringWithFormat:@"?%@&%@&%@&%@&%@",  [NSString stringWithFormat:@"%@=%@", [@"ServiceKey" urlEncodeUsingEncoding:NSUTF8StringEncoding], [@"서비스키" urlEncodeUsingEncoding:NSUTF8StringEncoding]]  , [NSString stringWithFormat:@"%@=%@", [@"serviceKey" urlEncodeUsingEncoding:NSUTF8StringEncoding], [@"-" urlEncodeUsingEncoding:NSUTF8StringEncoding]]  , [NSString stringWithFormat:@"%@=%@", [@"areaNo" urlEncodeUsingEncoding:NSUTF8StringEncoding], [@"1100000000" urlEncodeUsingEncoding:NSUTF8StringEncoding]]  , [NSString stringWithFormat:@"%@=%@", [@"time" urlEncodeUsingEncoding:NSUTF8StringEncoding], [@"2021070618" urlEncodeUsingEncoding:NSUTF8StringEncoding]]  , [NSString stringWithFormat:@"%@=%@", [@"dataType" urlEncodeUsingEncoding:NSUTF8StringEncoding], [@"xml" urlEncodeUsingEncoding:NSUTF8StringEncoding]]];  [request setURL:[NSURL URLWithString:[NSString stringWithFormat:@"%@%@",endpoint,queryParams]]];  [request setHTTPMethod:@"GET"];  NSHTTPURLResponse \*urlResponse = nil;  NSError \*error = [[NSError alloc] init];  NSData \*responseData = [NSURLConnection sendSynchronousRequest:request returningResponse:&urlResponse error:&error];  NSString \*result = [[NSString alloc] initWithData:responseData encoding:NSUTF8StringEncoding];  NSLog(@"Response Code: %d", [urlResponse statusCode]);  if ([urlResponse statusCode] >= 200 && [urlResponse statusCode] < 300)  {  NSLog(@"Response: %@", result);  }  return 0;  }  } |
| **Ruby Code** |
| /\* Ruby 샘플 코드 \*/  require 'rubygems'  require 'rest\_client'  require 'cgi'  url = ‘http://apis.data.go.kr/1360000/LivingWthrIdxServiceV4/getAirDiffusionIdxV4’  headers = { :params => { CGI::escape('ServiceKey') => '서비스키',CGI::escape('serviceKey') => '-',CGI::escape('areaNo') => '1100000000',CGI::escape('time') => '2021070618',CGI::escape('dataType') => 'xml' } }  response = RestClient::Request.execute :method => 'GET', :url => url , :headers => headers  puts response |
| **Python Code** |
| /\* Python 샘플 코드 \*/  From urllib2 import Request, urlopen  from urllib import urlencode, quote\_plus  url = ‘http://apis.data.go.kr/1360000/LivingWthrIdxService01/getAirDiffusionIdxV4’  queryParams = '?' + urlencode({ quote\_plus('ServiceKey') : '서비스키', quote\_plus('serviceKey') : '-', quote\_plus('areaNo') : '1100000000', quote\_plus('time') : '2021070618', quote\_plus('dataType') : 'xml' })  request = Request(url + queryParams)  request.get\_method = lambda: 'GET'  response\_body = urlopen(request).read()  print response\_body |
| **Node.js Code** |
| /\* NodeJs 샘플 코드 \*/  var request = require('request');  var url =’ http://apis.data.go.kr/1360000/LivingWthrIdxServiceV4/getAirDiffusionIdxV4’;  var queryParams = '?' + encodeURIComponent('ServiceKey') + '=서비스키'; /\* Service Key\*/  queryParams += '&' + encodeURIComponent('serviceKey') + '=' + encodeURIComponent('-'); /\* 공공데이터포털에서 받은 인증키 \*/  queryParams += '&' + encodeURIComponent('areaNo') + '=' + encodeURIComponent('1100000000'); /\* 서울지점 \*/  queryParams += '&' + encodeURIComponent('time') + '=' + encodeURIComponent('2021070618'); /\* 2021년7월6일18시 \*/  queryParams += '&' + encodeURIComponent('dataType') + '=' + encodeURIComponent('xml'); /\* xml, json 선택(미입력시 xml) \*/  request( {  url: url + queryParams,  method: 'GET'  }, function (error, response, body)  {  //console.log('Status', response.statusCode);  //console.log('Headers', JSON.stringify(response.headers));  //console.log('Reponse received', body);  }); |
| **Kotlin Code** |
| /\* Kotlin 예제 \*/  import java.io.BufferedReader  import java.io.InputStreamReader  import java.net.HttpURLConnection  import java.net.URL  import java.net.URLEncoder  object OpenAPI  {  fun main(args: Array<String>) {  val urlBuilder = StringBuilder("http://apis.data.go.kr/1360000/LivingWthrIdxServiceV4/getAirDiffusionIdxV4") /\* URL \*/  urlBuilder.append("?" + URLEncoder.encode("ServiceKey", "UTF-8") + "=서비스키") /\* Service Key \*/  urlBuilder.append("&" + URLEncoder.encode("ServiceKey", "UTF-8") + "=" + URLEncoder.encode("-", "UTF-8")) /\* 공공데이터포털에서 받은 인증키 \*/  urlBuilder.append("&" + URLEncoder.encode("areaNo", "UTF-8") + "=" + URLEncoder.encode("110000000", "UTF-8")) /\* 지점코드 \*/  urlBuilder.append("&" + URLEncoder.encode("time", "UTF-8") + "=" + URLEncoder.encode("2021070618", "UTF-8")) /\* 2021년7월6일18시 \*/  urlBuilder.append("&" + URLEncoder.encode("dataType", "UTF-8") + "=" + URLEncoder.encode("xml", "UTF-8")) /\* xml, json (미입력시 xml) \*/  val url = URL(urlBuilder.toString())  val conn = url.openConnection() as HttpURLConnection  conn.requestMethod = "GET"  conn.setRequestProperty("Content-type", "application/json")  System.out.println("Response code: " + conn.responseCode)  val br: BufferedReader  br = if (conn.responseCode >= 200 && conn.responseCode <= 300)  {  BufferedReader(InputStreamReader(conn.inputStream))  } else  {  BufferedReader(InputStreamReader(conn.errorStream))  }  val sb = StringBuilder()  var line: String?  do  {  line = br.readLine()  sb.append(line)  } while (line != null)  br.close()  conn.disconnect()  System.out.println(sb.toString())  }  } |

3) [체감온도(여름철)조회] 상세기능명세

a) 상세기능정보

|  |  |  |  |
| --- | --- | --- | --- |
| **상세기능 번호** | 4 | **상세기능 유형** | 조회 (상세) |
| **상세기능명(국문)** | 체감온도(여름철)조회 | | |
| **상세기능 설명** | 지점코드, 발표시간부터 1시간 단위로 최대 78시간 후 까지의 예측값을 제공한다  생산주기: 일 8회(3시간 간격/ 00시, 03시, 06시…….)  제공기간: 5월~9월  각 단계 및 범위는 상단의 <표> “체감온도(여름철) 단계 및 범위” 참고 / 지수자료(도로)가 34일 경우 경고 값 표출 | | |
| **Call Back URL** | http://apis.data.go.kr/1360000/LivingWthrIdxServiceV4/getSenTaIdxV4 | | |
| **최대 메시지 사이즈** | [11.2] Kbyte | | |
| **평균 응답 시간** | [500] ms | **초당 최대 트랙잭션** | [30] tps |

# 자료제공 기간 : 5월 1일 ~ 9월 30일

# 체감온도(여름철) 단계 및 범위

|  |  |
| --- | --- |
| 지수단계 | 데이터 조건 |
| 노인 | - : 29미만  관심 : 29이상 31 미만  주의 : 31이상 34미만  경고 : 34이상 37미만  위험 : 37이상 |
| 어린이 |
| 취약거주환경 |
| 농촌 | - : 29미만  관심 : 29이상 33 미만  주의 : 33이상 35미만  경고 : 35이상 38미만  위험 : 38이상 |
| 비닐하우스 |
| 도로 | - : 31미만  관심 : 31이상 33 미만  주의 : 33이상 35미만  경고 : 35이상 38미만  위험 : 38이상 |
| 건설현장 |
| 조선소 |

# 서비스 대상에 대한 정의

|  |  |
| --- | --- |
| **서비스대상** | **정의** |
| 노인  (A41) | 일반 환경에서 노년의 사람이 서 있는 상태를 가정 |
| 어린이  (A42) | 어린이의 야외활동이 많은 운동장(모래 바닥)의 지상 1~1.5m 높이의 환경을 가정 |
| 취약거주환경 (A46) | 창문이 없거나 작아 환기가 안되고 바람이 거의 없는 쪽방, 강한 햇볕 아래 직접적 영향을 받는 옥탑방 환경을 가정 |
| 농촌  (A44) | 농작업을 하는 야외 노지로 그늘이 없고 바람이 드는 환경을 가정 |
| 비닐하우스 (A45) | 농작물 수확을 위해 지어진 백색의 비닐하우스로 통풍 및 환기가 되지 않는 환경을 가정 |
| 도로(A47) | 햇볕이 내리쬐는 포장된 도로 위에서 작업하는 환경을 가정 |
| 건설현장(A48) | 작업자 주변에 철근이나 구조물 등이 설치된 건설현장에서 작업하는 환경을 가정 |
| 조선소(A49) | 햇볕이 내리쬐는 철판 바닥 위에서 작업하는 환경을 가정 |
| - 기상청 홈페이지 표출 (www.weather.go.kr > 테마날씨 > 생활기상지수 | |

e) 그 외 활용 예제코드

|  |
| --- |
| **Java Code** |
| /\* Java 샘플 코드 \*/  import java.io.InputStreamReader;  import java.net.HttpURLConnection;  import java.net.URL;  import java.net.URLEncoder;  import java.io.BufferedReader;  import java.io.IOException;  public class ApiExplorer  {  public static void main(String[] args) throws IOException  {  StringBuilder urlBuilder = new StringBuilder("http://apis.data.go.kr/1360000/LivingWthrIdxServiceV4/getSenTaIdxV4"); /\*URL\*/  urlBuilder.append("?" + URLEncoder.encode("ServiceKey","UTF-8") + "=서비스키"); /\*Service Key\*/  urlBuilder.append("&" + URLEncoder.encode("ServiceKey","UTF-8") + "=" + URLEncoder.encode("-", "UTF-8")); /\*공공데이터포털에서 받은 인증키\*/  urlBuilder.append("&" + URLEncoder.encode("pageNo","UTF-8") + "=" + URLEncoder.encode("1", "UTF-8")); /\*페이지번호\*/  urlBuilder.append("&" + URLEncoder.encode("numOfRows","UTF-8") + "=" + URLEncoder.encode("10", "UTF-8")); /\*한 페이지 결과 수\*/  urlBuilder.append("&" + URLEncoder.encode("dataType","UTF-8") + "=" + URLEncoder.encode("XML", "UTF-8")); /\*요청자료형식(XML/JSON)\*/  urlBuilder.append("&" + URLEncoder.encode("areaNo","UTF-8") + "=" + URLEncoder.encode("1100000000", "UTF-8")); /\*지점코드\*/  urlBuilder.append("&" + URLEncoder.encode("time","UTF-8") + "=" + URLEncoder.encode("2021070618", "UTF-8")); /\*2021년 7월 6일 18시 발표\*/  urlBuilder.append("&" + URLEncoder.encode("requestCode","UTF-8") + "=" + URLEncoder.encode("A41", "UTF-8")); /\*서비스대상 요청코드\*/  URL url = new URL(urlBuilder.toString());  HttpURLConnection conn = (HttpURLConnection) url.openConnection();  conn.setRequestMethod("GET");  conn.setRequestProperty("Content-type", "application/json");  System.out.println("Response code: " + conn.getResponseCode());  BufferedReader rd;  if(conn.getResponseCode() >= 200 && conn.getResponseCode() <= 300)  {  rd = new BufferedReader(new InputStreamReader(conn.getInputStream()));  } else  {  rd = new BufferedReader(new InputStreamReader(conn.getErrorStream()));  }  StringBuilder sb = new StringBuilder();  String line;  while ((line = rd.readLine()) != null)  {  sb.append(line);  }  rd.close();  conn.disconnect();  System.out.println(sb.toString());  }  } |
| **Javascript Code** |
| /\* Javascript 샘플 코드 \*/  var xhr = new XMLHttpRequest();  var url = 'http://apis.data.go.kr/1360000/LivingWthrIdxServiceV4/getSenTaIdxV4'; /\*URL\*/  var queryParams = '?' + encodeURIComponent('ServiceKey') + '='+'서비스키'; /\*Service Key\*/  queryParams += '&' + encodeURIComponent('pageNo') + '=' + encodeURIComponent('1'); /\*페이지번호\*/  queryParams += '&' + encodeURIComponent('numOfRows') + '=' + encodeURIComponent('10'); /\*한 페이지 결과 수\*/  queryParams += '&' + encodeURIComponent('dataType') + '=' + encodeURIComponent('XML'); /\*요청자료형식(XML/JSON)\*/  queryParams += '&' + encodeURIComponent('areaNo') + '=' + encodeURIComponent('1100000000'); /\*지점코드\*/  queryParams += '&' + encodeURIComponent('time') + '=' + encodeURIComponent('2021070618'); /\*2021년 7월 06일 18시 발표\*/  queryParams += '&' + encodeURIComponent('requestCode') + '=' + encodeURIComponent('A41'); /\*서비스대상 요청코드\*/  xhr.open('GET', url + queryParams);  xhr.send('');  xhr.onreadystatechange = function () {  if (this.readyState == 4) {  alert('Status: '+this.status+' Headers: '+JSON.stringify(this.getAllResponseHeaders())+' Body: '+this.responseText);  }  }; |
| **PHP Code** |
| /\* PHP 샘플 코드 \*/  $ch = curl\_init();  $url = 'http://apis.data.go.kr/1360000/LivingWthrIdxServiceV4/getSenTaIdxV4'; /\*URL\*/  $queryParams = '?' . urlencode('ServiceKey') . '=서비스키'; /\*Service Key\*/  $queryParams .= '&' . urlencode('ServiceKey') . '=' . urlencode('-'); /\*공공데이터포털에서 받은 인증키\*/  $queryParams .= '&' . urlencode('pageNo') . '=' . urlencode('1'); /\*페이지번호\*/  $queryParams .= '&' . urlencode('numOfRows') . '=' . urlencode('10'); /\*한 페이지 결과 수\*/  $queryParams .= '&' . urlencode('dataType') . '=' . urlencode('XML'); /\*요청자료형식(XML/JSON)\*/  $queryParams .= '&' . urlencode('areaNo') . '=' . urlencode('1100000000'); /\*지점코드\*/  $queryParams .= '&' . urlencode('time') . '=' . urlencode('2021070618'); /\*2021년 7월 06일 18시 발표\*/  $queryParams .= '&' . urlencode('requestCode') . '=' . urlencode('A41'); /\*서비스대상 요청코드\*/  curl\_setopt($ch, CURLOPT\_URL, $url . $queryParams);  curl\_setopt($ch, CURLOPT\_RETURNTRANSFER, TRUE);  curl\_setopt($ch, CURLOPT\_HEADER, FALSE);  curl\_setopt($ch, CURLOPT\_CUSTOMREQUEST, 'GET');  $response = curl\_exec($ch);  curl\_close($ch);  var\_dump($response); |
| **C++ Code** |
| /\* C++ 샘플 코드 \*/  using System;  using System.Net;  using System.Net.Http;  using System.IO;  namespace ConsoleApp1  {  class Program  {  static HttpClient client = new HttpClient();  static void Main(string[] args)  {  string url = "http://apis.data.go.kr/1360000/LivingWthrIdxServiceV4/getSenTaIdxV4"; // URL  url += "?ServiceKey=" + "서비스키"; // Service Key  url += "&ServiceKey=-";  url += "&pageNo=1";  url += "&numOfRows=10";  url += "&dataType=XML";  url += "&areaNo=1100000000";  url += "&time=2021070618";  url += "&requestCode=A41";    var request = (HttpWebRequest)WebRequest.Create(url);  request.Method = "GET";    string results = string.Empty;  HttpWebResponse response;  using (response = request.GetResponse() as HttpWebResponse)  {  StreamReader reader = new StreamReader(response.GetResponseStream());  results = reader.ReadToEnd();  }  Console.WriteLine(results);  }  }  } |
| **Curl Code** |
| /\* cURL 샘플 코드 \*/  curl --include --request GET \  'http://apis.data.go.kr/1360000/LivingWthrIdxServiceV4/getSenTaIdxV4?ServiceKey=서비스키&ServiceKey=-&pageNo=1&numOfRows=10&dataType=XML&areaNo=1100000000&time=2021070618&requestCode=A41' |
| **Object C Code** |
| /\* Objective-C 샘플 코드 \*/  #import <Foundation/Foundation.h>  #import <CoreFoundation/CoreFoundation.h>  @implementation NSString (URLEncoding)  //function used to encode query and matrix parameters  -(NSString \*)urlEncodeUsingEncoding:(NSStringEncoding)encoding  {  return (\_\_bridge NSString \*)CFURLCreateStringByAddingPercentEscapes(NULL,  (CFStringRef)self,  NULL,  (CFStringRef)@"!\*'\\"();:@&=+$,/?%#[]% ",  CFStringConvertNSStringEncodingToEncoding(encoding));  }  @end  int main (int argc, const char \* argv[])  {  @autoreleasepool  {  NSMutableURLRequest \*request = [[NSMutableURLRequest alloc] init];  NSString \*endpoint = [NSString stringWithFormat:@"http://apis.data.go.kr/1360000/LivingWthrIdxServiceV4/getSenTaIdxV4"]; /\*URL\*/  NSString \*queryParams = [NSString stringWithFormat:@"?%@&%@&%@&%@&%@&%@&%@&%@",  [NSString stringWithFormat:@"%@=%@", [@"ServiceKey" urlEncodeUsingEncoding:NSUTF8StringEncoding], [@"서비스키" urlEncodeUsingEncoding:NSUTF8StringEncoding]]  , [NSString stringWithFormat:@"%@=%@", [@"ServiceKey" urlEncodeUsingEncoding:NSUTF8StringEncoding], [@"-" urlEncodeUsingEncoding:NSUTF8StringEncoding]]  , [NSString stringWithFormat:@"%@=%@", [@"pageNo" urlEncodeUsingEncoding:NSUTF8StringEncoding], [@"1" urlEncodeUsingEncoding:NSUTF8StringEncoding]]  , [NSString stringWithFormat:@"%@=%@", [@"numOfRows" urlEncodeUsingEncoding:NSUTF8StringEncoding], [@"10" urlEncodeUsingEncoding:NSUTF8StringEncoding]]  , [NSString stringWithFormat:@"%@=%@", [@"dataType" urlEncodeUsingEncoding:NSUTF8StringEncoding], [@"XML" urlEncodeUsingEncoding:NSUTF8StringEncoding]]  , [NSString stringWithFormat:@"%@=%@", [@"areaNo" urlEncodeUsingEncoding:NSUTF8StringEncoding], [@"1100000000" urlEncodeUsingEncoding:NSUTF8StringEncoding]]  , [NSString stringWithFormat:@"%@=%@", [@"time" urlEncodeUsingEncoding:NSUTF8StringEncoding], [@"2021070618" urlEncodeUsingEncoding:NSUTF8StringEncoding]]  , [NSString stringWithFormat:@"%@=%@", [@"requestCode" urlEncodeUsingEncoding:NSUTF8StringEncoding], [@"A41" urlEncodeUsingEncoding:NSUTF8StringEncoding]]];  [request setURL:[NSURL URLWithString:[NSString stringWithFormat:@"%@%@",endpoint,queryParams]]];  [request setHTTPMethod:@"GET"];  NSHTTPURLResponse \*urlResponse = nil;  NSError \*error = [[NSError alloc] init];  NSData \*responseData = [NSURLConnection sendSynchronousRequest:request returningResponse:&urlResponse error:&error];  NSString \*result = [[NSString alloc] initWithData:responseData encoding:NSUTF8StringEncoding];  NSLog(@"Response Code: %d", [urlResponse statusCode]);  if ([urlResponse statusCode] >= 200 && [urlResponse statusCode] < 300)  {  NSLog(@"Response: %@", result);  }  return 0;  }  } |
| **Ruby Code** |
| /\* Ruby 샘플 코드 \*/  require 'rubygems'  require 'rest\_client'  require 'cgi'  url = 'http://apis.data.go.kr/1360000/LivingWthrIdxServiceV4/getSenTaIdxV4'  headers = { :params => { CGI::escape('ServiceKey') => '서비스키',CGI::escape('ServiceKey') => '-',CGI::escape('pageNo') => '1',CGI::escape('numOfRows') => '10',CGI::escape('dataType') => 'XML',CGI::escape('areaNo') => '1100000000',CGI::escape('time') => '2021070618',CGI::escape('requestCode') => 'A41' } }  response = RestClient::Request.execute :method => 'GET', :url => url , :headers => headers  puts response |
| **Python Code** |
| /\* Python 샘플 코드 \*/  from urllib2 import Request, urlopen  from urllib import urlencode, quote\_plus  url = 'http://apis.data.go.kr/1360000/LivingWthrIdxServiceV1/getSenTaIdxV4'  queryParams = '?' + urlencode({ quote\_plus('ServiceKey') : '서비스키', quote\_plus('ServiceKey') : '-', quote\_plus('pageNo') : '1', quote\_plus('numOfRows') : '10', quote\_plus('dataType') : 'XML', quote\_plus('areaNo') : '1100000000', quote\_plus('time') : '2021070618', quote\_plus('requestCode') : 'A41' })  request = Request(url + queryParams)  request.get\_method = lambda: 'GET'  response\_body = urlopen(request).read()  print response\_body |
| **Node.js Code** |
| /\* NodeJs 샘플 코드 \*/  var request = require('request');  var url = 'http://apis.data.go.kr/1360000/LivingWthrIdxServiceV4/getSenTaIdxV4';  var queryParams = '?' + encodeURIComponent('ServiceKey') + '=서비스키'; /\* Service Key\*/  queryParams += '&' + encodeURIComponent('ServiceKey') + '=' + encodeURIComponent('-'); /\* 공공데이터포털에서 받은 인증키 \*/  queryParams += '&' + encodeURIComponent('pageNo') + '=' + encodeURIComponent('1'); /\* 페이지번호 \*/  queryParams += '&' + encodeURIComponent('numOfRows') + '=' + encodeURIComponent('10'); /\* 한 페이지 결과 수 \*/  queryParams += '&' + encodeURIComponent('dataType') + '=' + encodeURIComponent('XML'); /\* 요청자료형식(XML/JSON) \*/  queryParams += '&' + encodeURIComponent('areaNo') + '=' + encodeURIComponent('1100000000'); /\* 지점코드 \*/  queryParams += '&' + encodeURIComponent('time') + '=' + encodeURIComponent('2021070618'); /\* 2021년 7월 6일 18시 발표 \*/  queryParams += '&' + encodeURIComponent('requestCode') + '=' + encodeURIComponent('A41'); /\* 서비스대상 요청코드 \*/  request(  {  url: url + queryParams,  method: 'GET'  }, function (error, response, body)  {  //console.log('Status', response.statusCode);  //console.log('Headers', JSON.stringify(response.headers));  //console.log('Reponse received', body);  }); |
| **Kotlin Code** |
| /\* Kotlin 예제 \*/  import java.io.BufferedReader  import java.io.InputStreamReader  import java.net.HttpURLConnection  import java.net.URL  import java.net.URLEncoder  object OpenAPI  {  fun main(args: Array<String>)  {  val urlBuilder = StringBuilder("http://apis.data.go.kr/1360000/LivingWthrIdxServiceV4/getSenTaIdxV4") /\* URL \*/  urlBuilder.append("?" + URLEncoder.encode("ServiceKey", "UTF-8") + "=서비스키") /\* Service Key \*/  urlBuilder.append("&" + URLEncoder.encode("ServiceKey", "UTF-8") + "=" + URLEncoder.encode("-", "UTF-8")) /\* 공공데이터포털에서 받은 인증키 \*/  urlBuilder.append("&" + URLEncoder.encode("areaNo", "UTF-8") + "=" + URLEncoder.encode("1168053100", "UTF-8")) /\* 지점코드 \*/  urlBuilder.append("&" + URLEncoder.encode("time", "UTF-8") + "=" + URLEncoder.encode("2021070618", "UTF-8")) /\* 2021년7월6일18시 \*/ urlBuilder.append("&" + URLEncoder.encode("requestCode","UTF-8") + "=" + URLEncoder.encode("A41", "UTF-8")); /\*서비스대상 요청코드\*/  urlBuilder.append("&" + URLEncoder.encode("\_type", "UTF-8") + "=" + URLEncoder.encode("xml", "UTF-8")) /\* xml, json (미입력시 xml) \*/  val url = URL(urlBuilder.toString())  val conn = url.openConnection() as HttpURLConnection  conn.requestMethod = "GET"  conn.setRequestProperty("Content-type", "application/json")  System.out.println("Response code: " + conn.responseCode)  val br: BufferedReader  br = if (conn.responseCode >= 200 && conn.responseCode <= 300)  {  BufferedReader(InputStreamReader(conn.inputStream))  } else  {  BufferedReader(InputStreamReader(conn.errorStream))  }  val sb = StringBuilder()  var line: String?  do  {  line = br.readLine()  sb.append(line)  } while (line != null)  br.close()  conn.disconnect()  System.out.println(sb.toString())  }  } |

# 생활기상지수 생산주기, 생산시간, 예측기간, 기상청 홈페이지 서비스 기간

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **지 수 명** | **생산주기** | **생산시간** | **예측기간** | **제공기간** |
| 자외선지수 | 일8회 | 03~24(3hr) | 오늘,내일,모레,글피  (+1h~ 최대+78h) | 연중 |
| 대기정체지수 |  | 03~24(3hr) | 오늘,내일,모레,글피  (+3h~ 최대+78h) | 연중 |
| 체감온도(여름철) | 03~24(3hr) | 오늘,내일,모레,글피  (+1h~ 최대+78h) | 5월1일~  9월30일 |