Alibaba Cloud DataV

Best Practices

Issue: 20190710

Legal disclaimer

Alibaba Cloud reminds you to carefully read and fully understand the terms and conditions of this legal disclaimer before you read or use this document. If you have read or used this document, it shall be deemed as your total acceptance of this legal disclaimer.

1. **You shall download and obtain this document from the Alibaba Cloud website or other Alibaba Cloud-authorized channels, and use this document for your own legal business activities only. The content of this document is considered confidential information of Alibaba Cloud. You shall strictly abide by the confidentiality obligations. No part of this document shall be disclosed or provided to any third party for use without the prior written consent of Alibaba Cloud.**
2. **No part of this document shall be excerpted, translated, reproduced, transmitted, or disseminated by any organization, company, or individual in any form or by any means without the prior written consent of Alibaba Cloud.**
3. **The content of this document may be changed due to product version upgrades , adjustments, or other reasons. Alibaba Cloud reserves the right to modify the content of this document without notice and the updated versions of this document will be occasionally released through Alibaba Cloud-authorized channels. You shall pay attention to the version changes of this document as they occur and download and obtain the most up-to-date version of this document from Alibaba Cloud-authorized channels.**
4. **This document serves only as a reference guide for your use of Alibaba Cloud products and services. Alibaba Cloud provides the document in the context that Alibaba Cloud products and services are provided on an "as is", "with all faults " and "as available" basis. Alibaba Cloud makes every effort to provide relevant operational guidance based on existing technologies. However, Alibaba Cloud hereby makes a clear statement that it in no way guarantees the accuracy, integrity , applicability, and reliability of the content of this document, either explicitly or implicitly. Alibaba Cloud shall not bear any liability for any errors or financial losses incurred by any organizations, companies, or individuals arising from their download, use, or trust in this document. Alibaba Cloud shall not, under any circumstances, bear responsibility for any indirect, consequential, exemplary, incidental, special, or punitive damages, including lost profits arising from the use**

or trust in this document, even if Alibaba Cloud has been notified of the possibility of such a loss.

5. By law, all the content of the Alibaba Cloud website, including but not limited to works, products, images, archives, information, materials, website architecture, website graphic layout, and webpage design, are intellectual property of Alibaba Cloud and/or its affiliates. This intellectual property includes, but is not limited to, trademark rights, patent rights, copyrights, and trade secrets. No part of the Alibaba Cloud website, product programs, or content shall be used, modified , reproduced, publicly transmitted, changed, disseminated, distributed, or published without the prior written consent of Alibaba Cloud and/or its affiliates . The names owned by Alibaba Cloud shall not be used, published, or reproduced for marketing, advertising, promotion, or other purposes without the prior written consent of Alibaba Cloud. The names owned by Alibaba Cloud include, but are not limited to, "Alibaba Cloud", "Aliyun", "HiChina", and other brands of Alibaba Cloud and/or its affiliates, which appear separately or in combination, as well as the auxiliary signs and patterns of the preceding brands, or anything similar to the company names, trade names, trademarks, product or service names, domain names, patterns, logos, marks, signs, or special descriptions that third parties identify as Alibaba Cloud and/or its affiliates).
6. Please contact Alibaba Cloud directly if you discover any errors in this document.

Generic conventions

Table -1: Style conventions

Style	Description	Example
	This warning information indicates a situation that will cause major system changes, faults, physical injuries, and other adverse results.	 Danger: Resetting will result in the loss of user configuration data.
	This warning information indicates a situation that may cause major system changes, faults, physical injuries, and other adverse results.	 Warning: Restarting will cause business interruption. About 10 minutes are required to restore business.
	This indicates warning information, supplementary instructions, and other content that the user must understand.	 Notice: Take the necessary precautions to save exported data containing sensitive information.
	This indicates supplemental instructions, best practices, tips, and other content that is good to know for the user.	 Note: You can use Ctrl + A to select all files.
>	Multi-level menu cascade.	Settings > Network > Set network type
Bold	It is used for buttons, menus , page names, and other UI elements.	Click OK.
`Courier font`	It is used for commands.	Run the `cd / d C :/ windows` command to enter the Windows system folder.
Italics	It is used for parameters and variables.	`bae log list --instanceid` *Instance_ID*
[] or [a|b]	It indicates that it is a optional value, and only one item can be selected.	`ipconfig` *[-all|-t]*

Style	Description	Example
{} or {a|b}	It indicates that it is a required value, and only one item can be selected.	`swich {stand | slave}`

Contents

1 Render Log Service data in DataV

The following sections describe how to configure DataV to render data from Log Service.

The following sections describe how to:

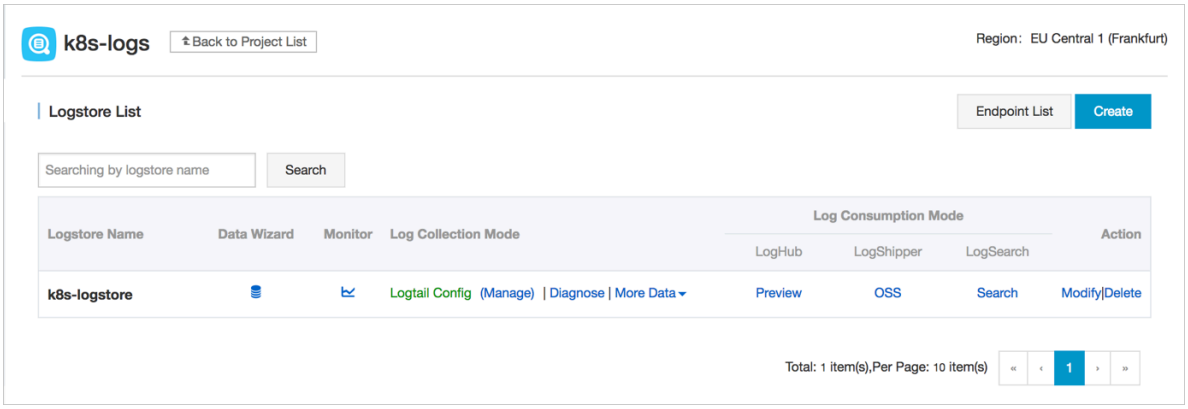
- Create and configure Log Service to work with DataV (set indexes).
- Create a sample dashboard.
- Share the dashboard publicly.

Prerequisites

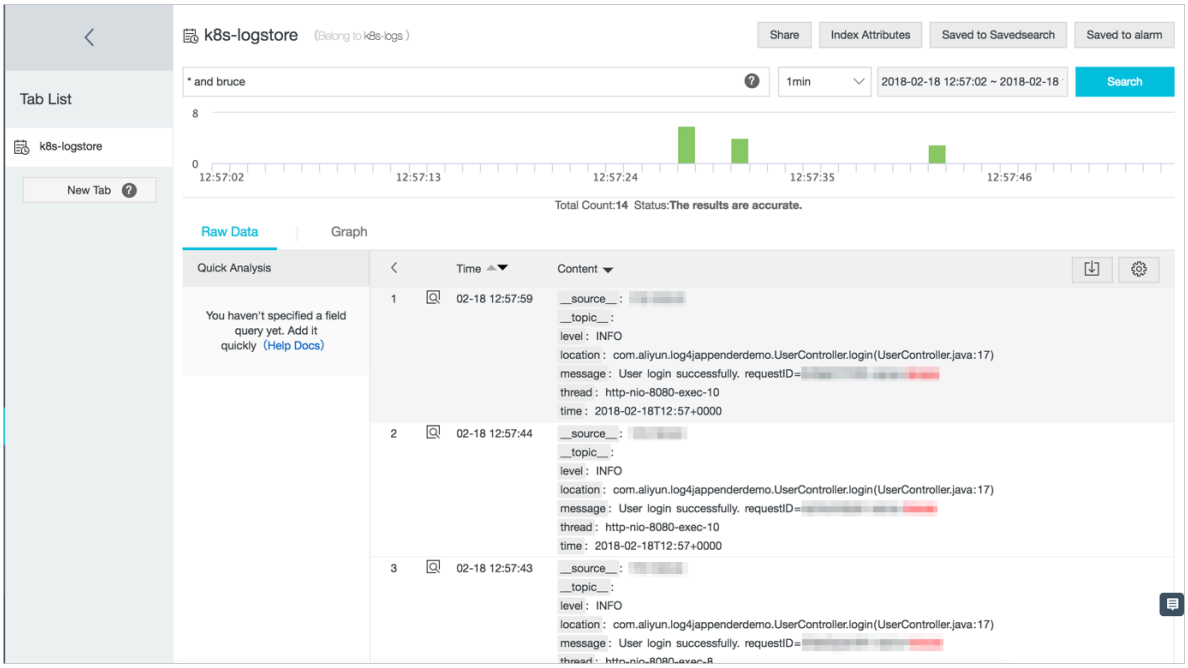
- You must have completed Configure Log4JAppender with Kubernetes and Log Serviceand the service is currently running.
- You must have purchased DataV Enterprise Edition.

Configure Log Service

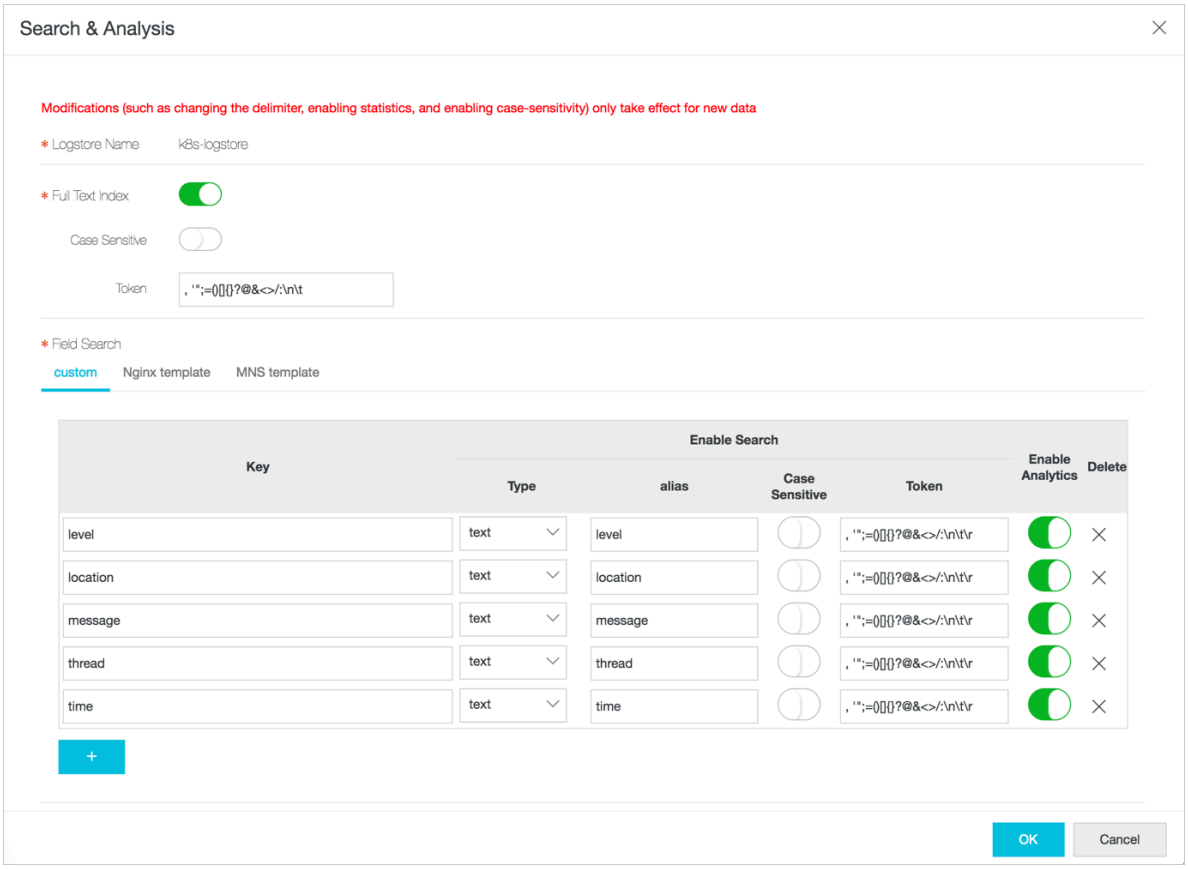
1. Visit the Logstore List page within your project.



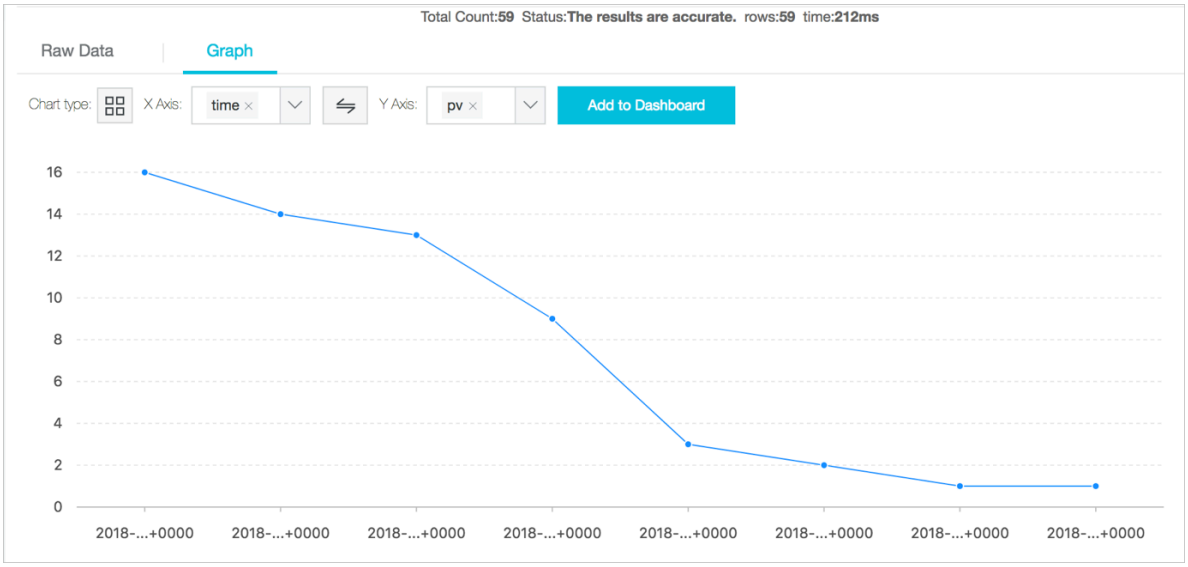
2. Click Search next to the name of your project. The following page is displayed:



3. Create indices for all required fields. The following example creates an index for each item. Click Index Attributes from the upper menu of the page and click Modify.
4. Verify the data from the Search & Analysis page:

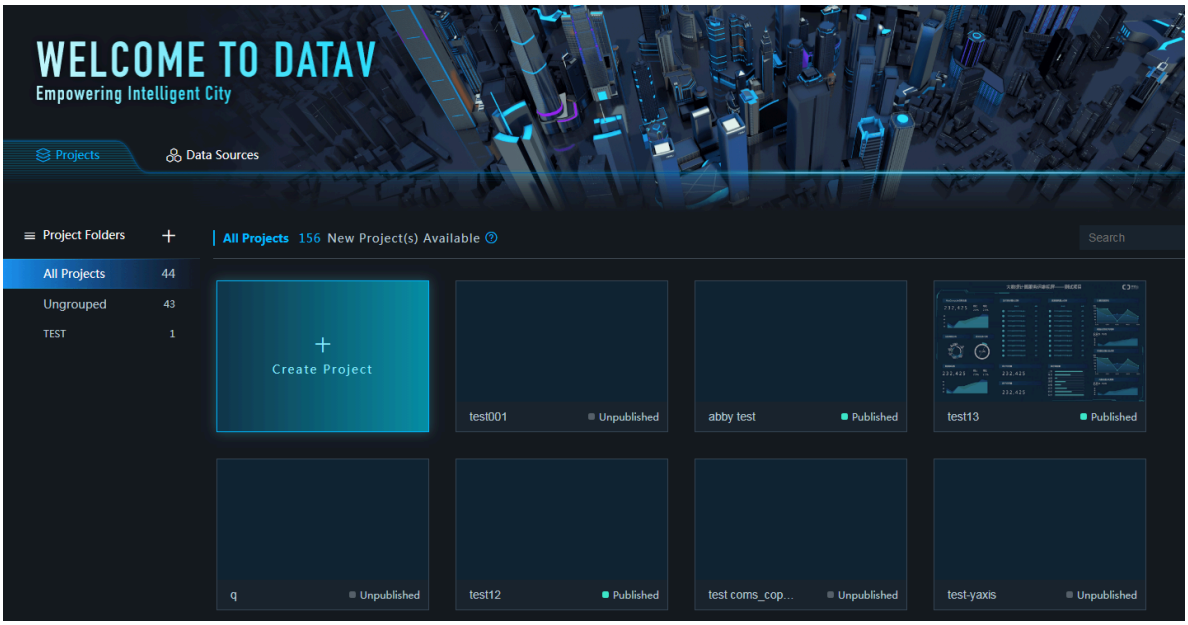


5. Once the data has been imported properly, switch to Graph view (in the following graph, the axis is ‘time’):



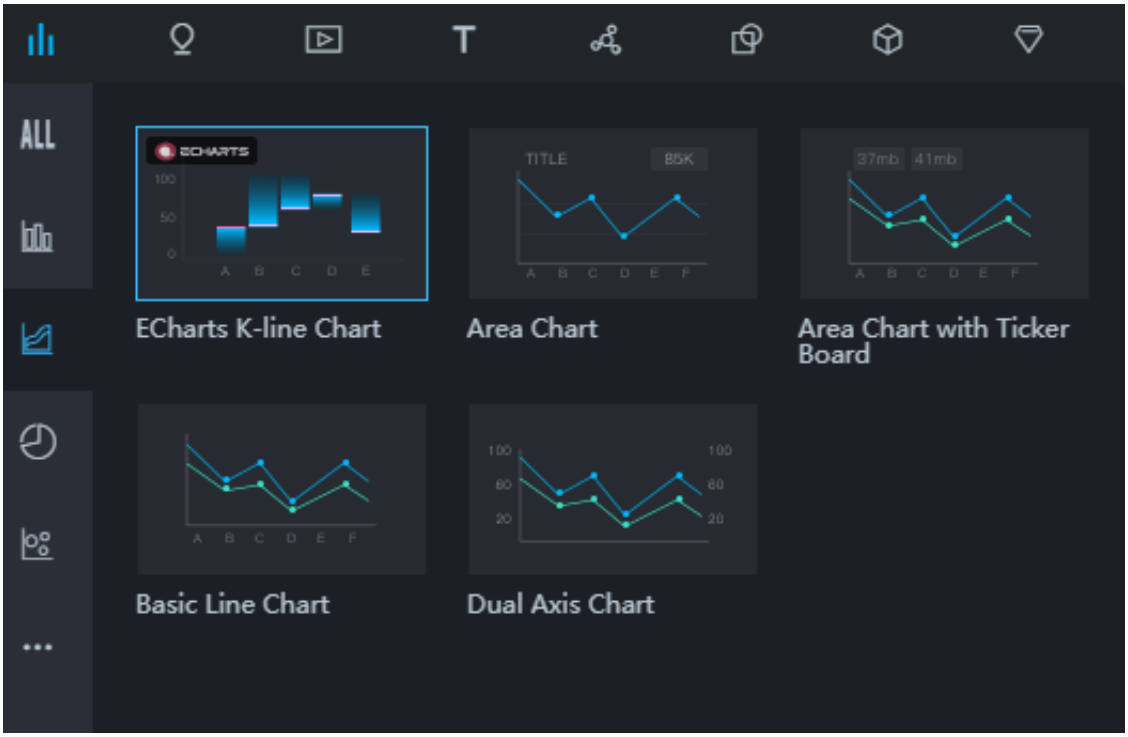
Configure DataV

1. Visit the DataV product page to create your first project.

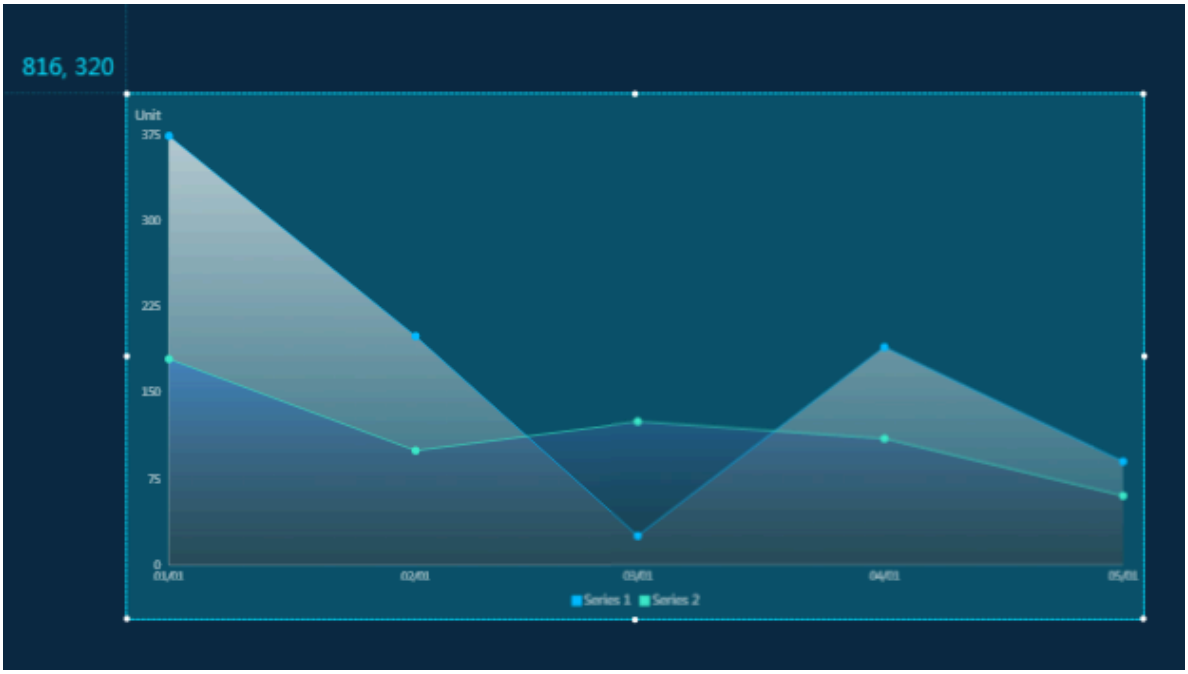


2. Click Create Project, select a blank template, and click Create.

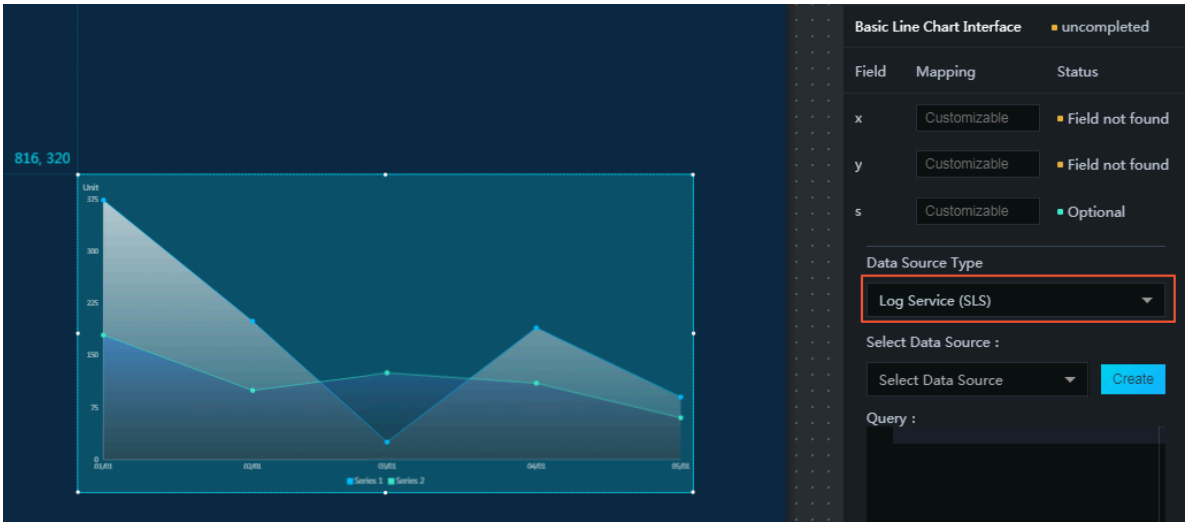
3. Add a widget to the dashboard.



The widget displays some sample static dataset.



4. Click the widget and, select the `Log Service ( SLS )` from `Data Source Type` from the Data tab on the right side.



5. Click Create in the `Select Data Source` . The New data dialog is displayed, fill in the relevant information, and click OK.

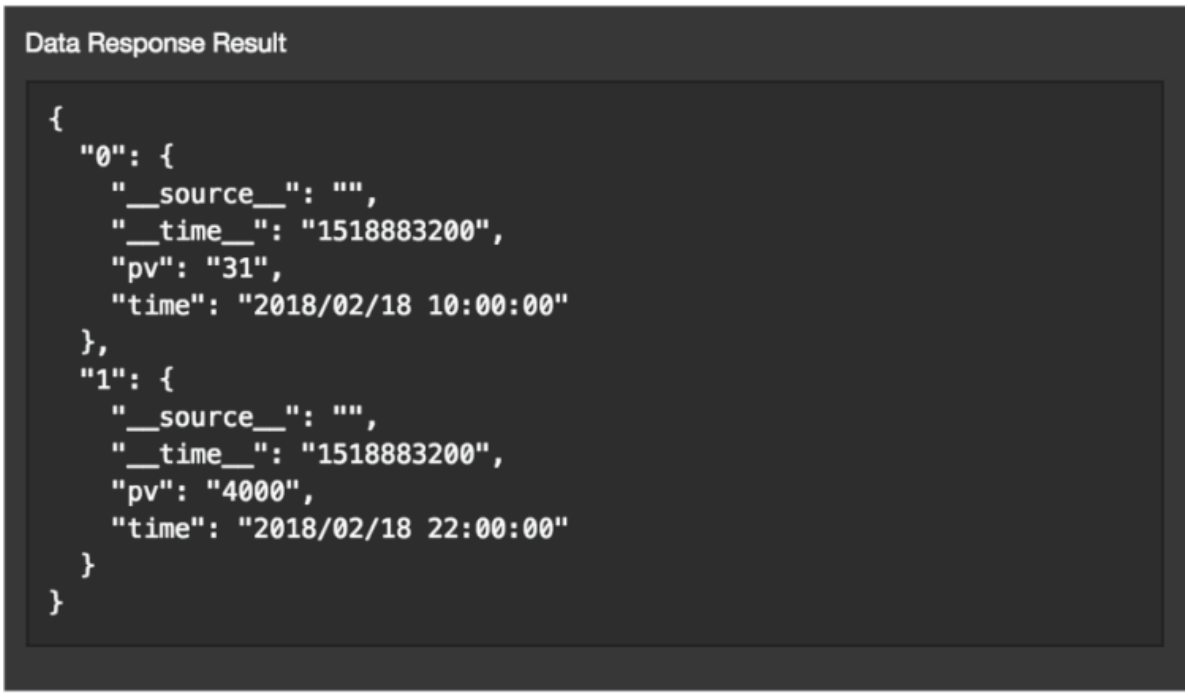
> Note:
> Make sure you add `http://` or `https://` in the `Endpoint` field.

6. Once completed, select the newly created Source. The following example uses a simple example query:

```
{
" projectNam  e ": " k8s - logs ",
" logStoreNa  me ": " k8s - logstore ",
" topic ": "",
" from ": " 1518883200 ",
" to ": " 1518969600 ",
" query ": "* |  select   count ( 1 )  as   pv ,  date_forma  t (
 from_unixt  ime ( __time__  -  __time__ % 3600 ) ,'% Y /% m /% d  %
 H :% i :% s ')  as   time   group   by   time   order   by   time
 limit   1000 " ,
" line ":  100 ,
" offset ":  0
}
```

> Note:
> *from* and *to* are the timestamps you can use to examine raw data in the `Search` console.

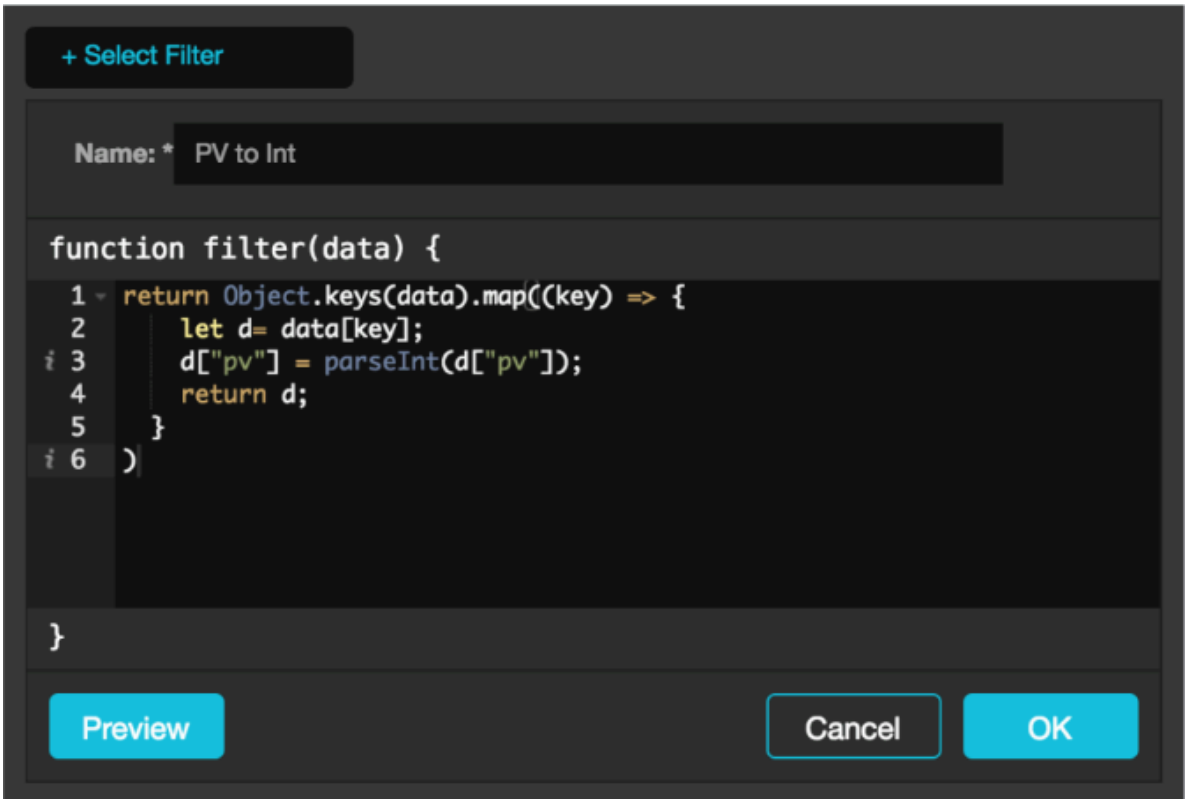
7. Preview the data by clicking View Data Response button at the lower-side of the window. The following response result window is displayed:



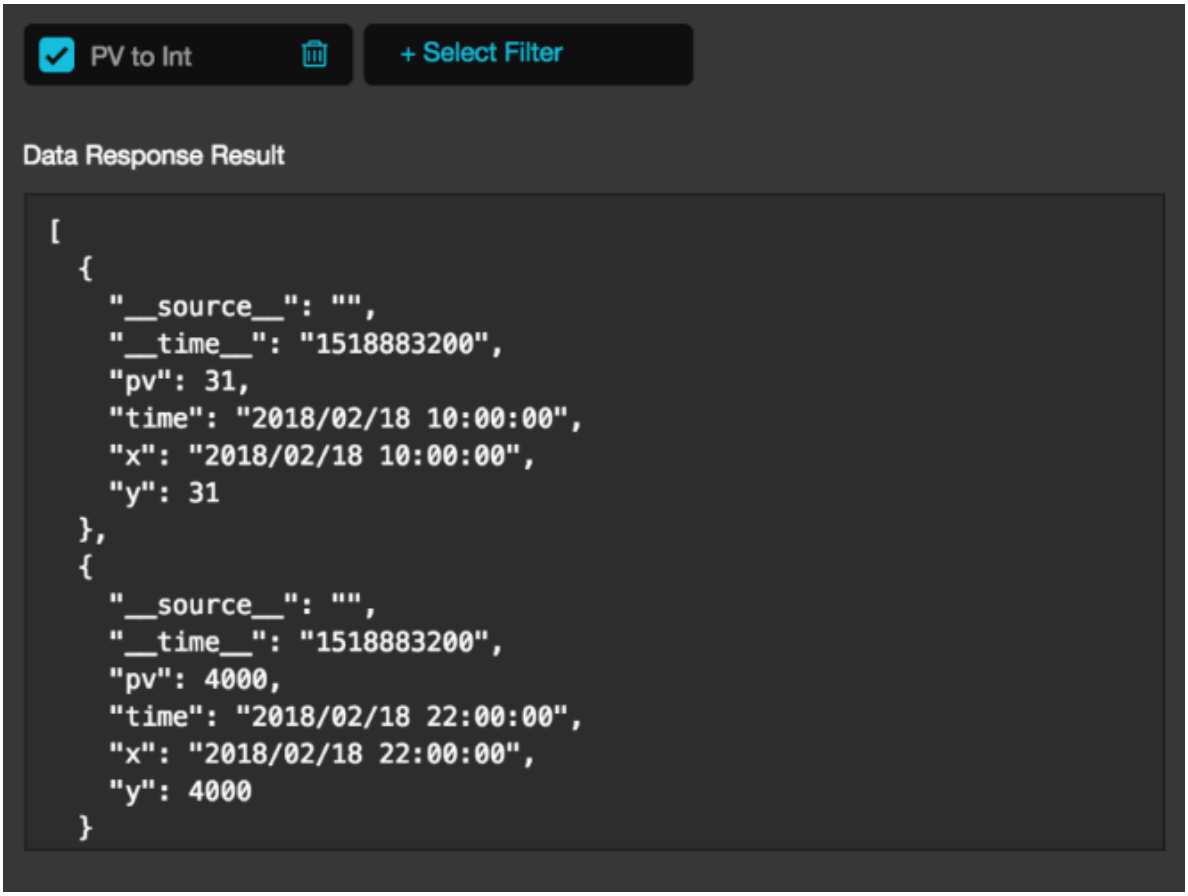
8. Click Select Filter and apply the following filter to make sure the `pv` is an integer, and click OK.

```
 return   Object . keys ( data ). map (( key ) => {
  let   d =  data [ key ];
  d [" pv "] =  parseInt ( d [" pv "]);
  return   d ;
}
```

```
)
```

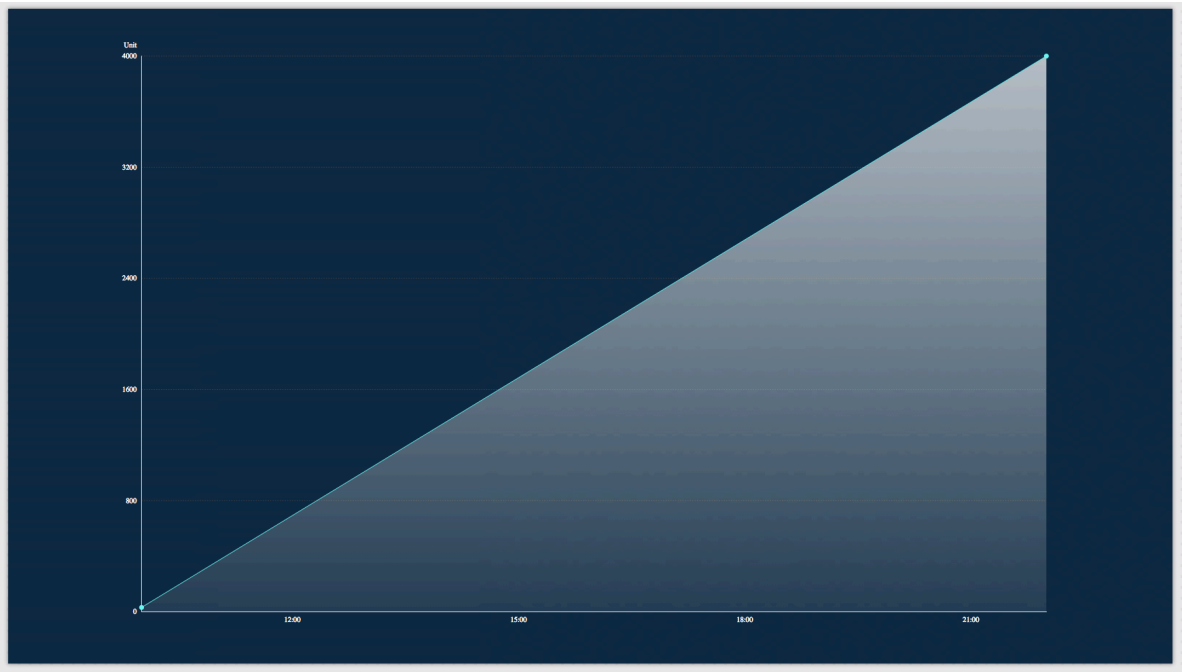


9. Set the axes and verify the settings are set correctly.



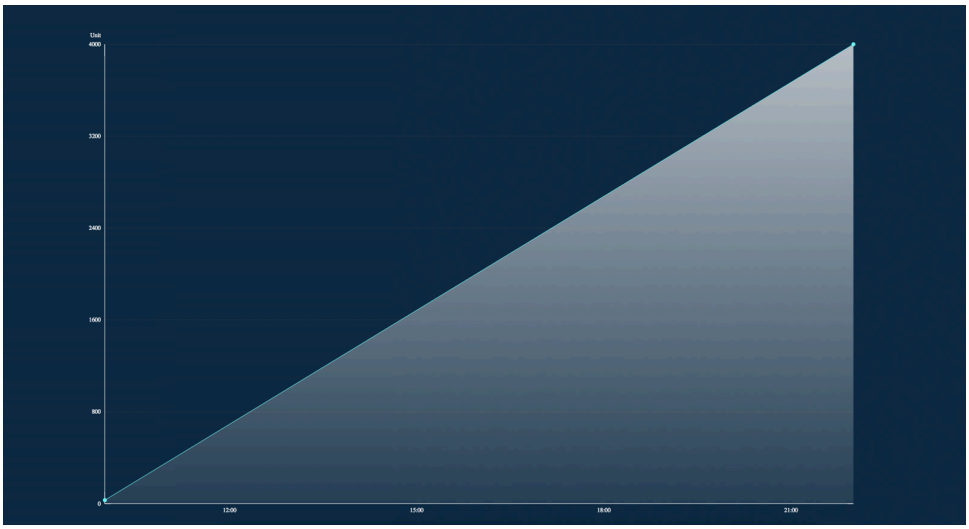
10.Click Preview.

You can see that `x` and `y` use the correct data type, and `pv` is an integer.



11. To share this dashboard publicly, click Publish in the upper-right corner of the page.

    An example of a completed and published DataV dashboard, using a dataset from a Log Service data, is as follows:



Conclusion

You have successfully configured DataV and Log Service together on Alibaba Cloud and used Log Service to perform real-time monitoring by means of a custom dashboard.

References

For more information on Log Service and containers, see

- Log Service
- Container Service

2 Use DataV to view air quality changes before and after the Spring Festival

2.1 Overview

This topic describes how to create a project in DataV to view air quality changes before and after the Spring Festival.

Procedure

1. Make preparations.
   a. Obtain data
   b. Process data
   c. Process the APIs
2. Create a project.
   a. Create a project
   b. Add widgets
   c. Add data
3. Publish the project.

   For more information, see Publish a project.

Description

You need to use the following feature or widgets when creating a project:

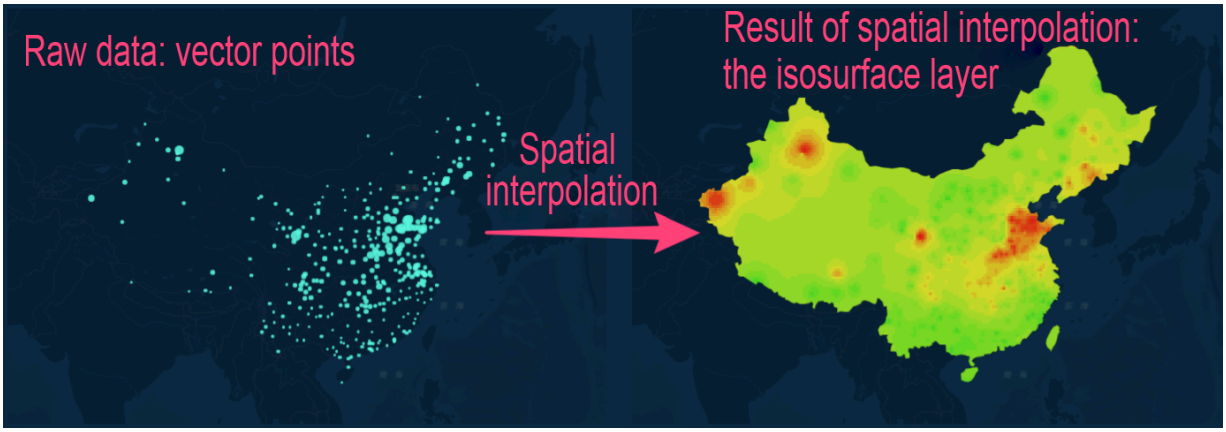
- Spatial interpolation
- Isosurface layer
- Timeline

Spatial interpolation

Spatial interpolation is generally used to convert scattered data into consecutive data on a curve to compare the data with data in other distribution modes.

That is, the data obtained from existing monitoring sites can be used to estimate data of other locations. Then, colors are mapped according to the value range and an isothermal map is generated.

By creating an isothermal map using DataV, you will start the process of spatial interpolation, in which scattered data obtained from monitoring sites is used to create consecutive data on a curve.

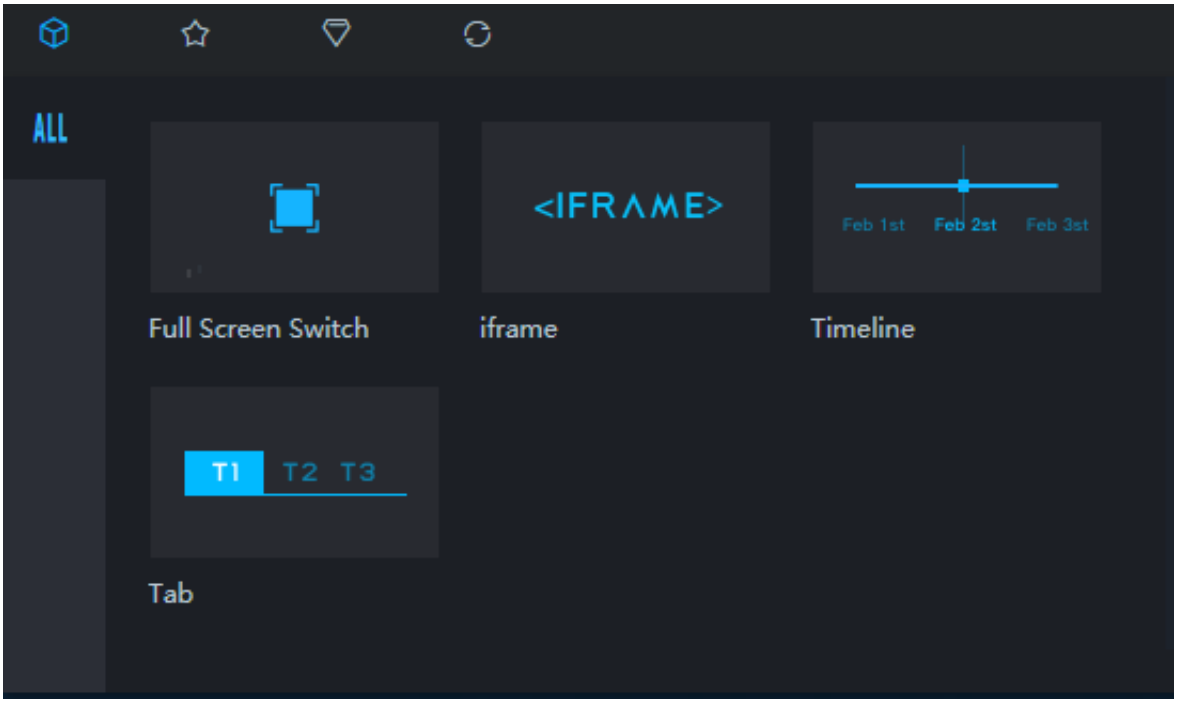


Isosurface layer

DataV provides an isosurface layer map widget featuring lightweight analysis, which can help you create a grid area map using data of known vector points. You can use this widget to create a real-time air quality map, as shown in the following figure.

Timeline

The timeline widget is necessary to display air quality changes during a period of time.



This widget supports callback IDs, which can be used to connect this widget to other widgets. Data of connected widgets will be automatically updated when the time on the timeline changes.

If a correct callback ID is set, the system will trigger a data request when time changes and automatically adds the callback ID and the value of the callback ID to the parameter list of the corresponding APIs of other widgets.

- IP address of the initial API: http://127.0.0.1:8888/aqi
- IP address of the API after callback is triggered: http://127.0.0.1:8888/aqi?date=2017012722

The callback ID is `date , 2017012722` .

The callback ID supports SQL statements. To use the callback ID, you need to use a colon (:) and the callback ID name in your SQL statements.

- Initial SQL statement: `select : date as value ;`
- SQL statement after callback is triggered: `select ' 2017022722 ' as value ;`

2.2 Preparations

2.2.1 Obtain data

This topic describes how to obtain air quality data before and after the Spring Festival. You need to obtain the data first before you can process the data.

You can download data from historical air quality data.

Note:

We recommend that you download the files in CSV format.

In this example, the data is obtained from 1,497 monitoring sites from the dates of January 1, 2017 to February 2, 2017.

Open the downloaded files and check whether the data needs to be supplemented or filtered.

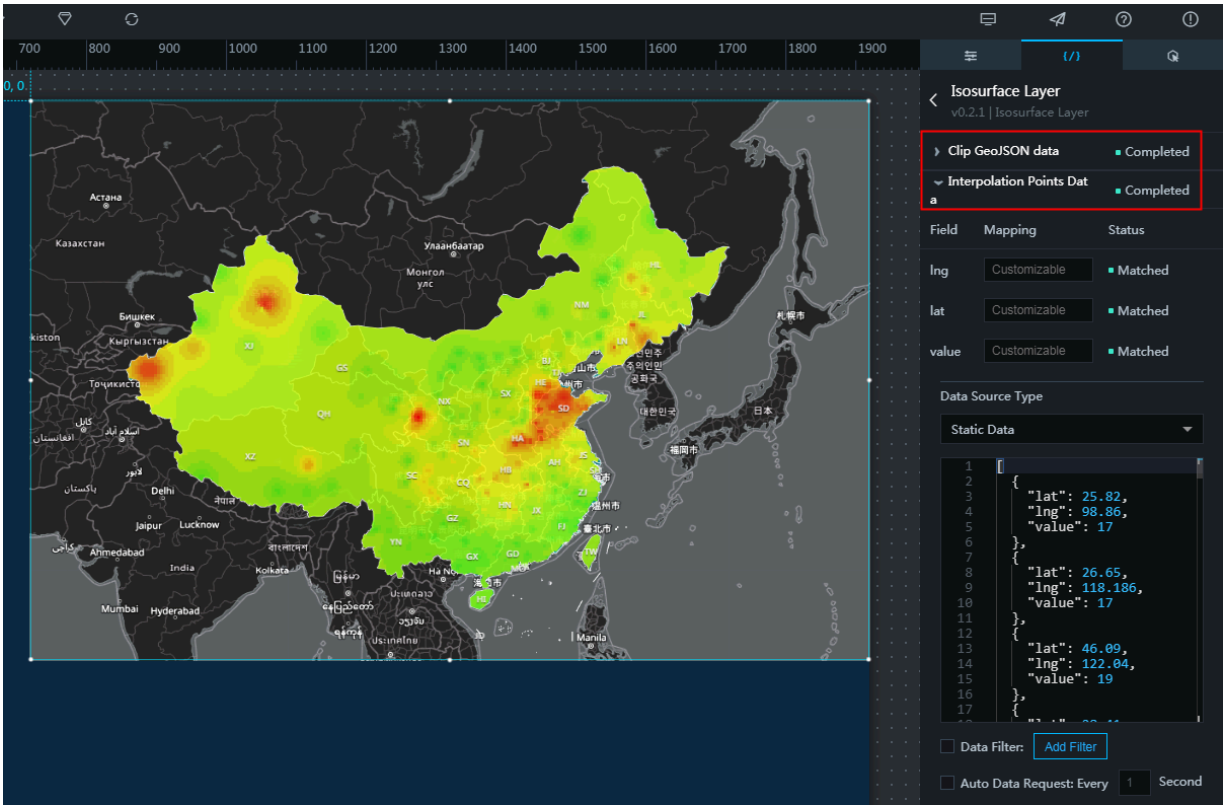
Monitoring site code	Monitoring site	City	Latitude	Longitude
1001A	Wanshouxigong	Beijing	116.366	39.8673
1002A	Dingling	Beijing	116.17	40.2866
1003A	Dongsi	Beijing	116.434	39.9522
1004A	Temple of Heaven	Beijing	116.434	39.8745
1005A	Beijing National Agriculture Exhibition Center	Beijing	116.473	39.9716
1006A	Guanyuan	Beijing	116.361	39.9425
1007A	Haidian Wanliu	Beijing	116.315	39.9934
1008A	Shunyi District	Beijing	116.72	40.1438
1009A	Huairou District	Beijing	116.644	40.3937
1010A	Changping District	Beijing	116.23	40.1952
1011A	Olympic Sports Centre	Beijing	116.407	40.0031
1012A	Gucheng	Beijing	116.225	39.9279
1013A	Municipal Environmental Monitoring Center	Tianjin	117.151	39.097
1014A	Nankou Road	Tianjin	117.193	39.173
1015A	Qinjian Road	Tianjin	117.145	39.1654
1016A	Nanjing Road	Tianjin	117.184	39.1205
1017A	Dazhigu No.8 Road	Tianjin	117.237	39.1082
1018A	Qianjin Road	Tianjin	117.202	39.0927
1019A	Beichen Technology Park	Tianjin	117.1837	39.2133
1020A	Tianshan Road	Tianjin	117.269	39.1337
1021A	Yuejin Road	Tianjin	117.307	39.0877
1023A	Forth Avenue	Tianjin	117.707	39.0343
1024A	Yongming Road	Tianjin	117.457	38.8394
1025A	Hangtian Road	Tianjin	117.401	39.124
1026A	Hanbei Road	Tianjin	117.764	39.1587
1027A	Tuanbowa	Tianjin	117.157	38.9149
1028A	School of Chemical Engineering	Shijiazhuang		
1029A	Hospital for Workers and Staff	Shijiazhuang	114.4548	38.0513
1030A	Gaoxin District	Shijiazhuang	114.6046	38.0398
1031A	Northwest Water Source Base	Shijiazhuang	114.5019	38.1398
1032A	High Education area in Southwest Shijiazhuang	Shijiazhuang	114.4586	38.00583
1033A	Centennial Park	Shijiazhuang	114.5331	38.01778
1034A	Great Hall of the People	Shijiazhuang	114.5214	38.0524
1035A	Fenglong Mountain	Shijiazhuang	114.3541	37.9097
1036A	Supply and Marketing Cooperative	Tangshan	118.1662	39.6308
1037A	Radar Station	Tangshan	118.144	39.643

You need to supplement or filter the data that contains no longitude and latitude information.

2.2.2 Process data

In this example, the CSV files are converted into JSON files.

The following is an example of the data format required by the isosurface layer widget. You need to process the data to better meet the requirements.



- Clip GeoJSON data: boundary data of the research area. Here the research area covers the whole of China, and the data is in GeoJSON format.

  GeoJSON is an open standard format designed for representing simple geographical features. For more information, see GeoJSON standards.
- Interpolation Points Data: an array that includes the longitude, latitude, and value of a monitoring site.

To create an isosurface map for a period of time in a day, for example, an air quality index (AQI) map at 12:00 on January 20, 2017, you need to obtain the position data (longitude and latitude) and the corresponding AQI of each monitoring site. To process data, follow these steps:

1. Use the following node scripts to process the CSV files:

```
var csv = require("fast-csv");
var fs = require('fs');
var map = {};
csv
.fromPath("./Site list (including the longitudes and latitudes)-new-1497.csv", {headers: true, objectMode: true})
.on("data", function(data){
  map[data['code']] = data;
})
.on("end", function(){
  fs.writeFile('./Longitude and latitude mapping in the site list.json', JSON.stringify(map));
```

```
  console . log (" done ");
});
```

A dictionary is obtained. In the dictionary, the monitoring site No. is the key and the site information is the value.

```
{
 " 1001A ": {
     " code ": " 1001A ",
     " name ": " Wanshouxig  ong ",
     " city ": " Beijing ",
     " lng ": " 116 . 366 ",
     " lat ": " 39 . 8673 "
 },
 " 1002A ": {
     " code ": " 1002A ",
     " name ": " Dingling ",
     " city ": " Beijing ",
     " lng ": " 116 . 17 ",
     " lat ": " 40 . 2865 "
 },
 " 1003A ": {
     " code ": " 1003A ",
     " name ": " Dongsi ",
     " city ": " Beijing ",
     " lng ": " 116 . 434 ",
     " lat ": " 39 . 9522 "
 },
 ...
}
```

2. **Process data obtained from 1,497 monitoring sites on January 20, 2017.**

   Use the following scripts to process the AQI data obtained within 24 hours from each monitoring site. Extract the data and add the longitudes and latitudes to the sites according to the longitude and latitude mapping list.

```
 var   fs  =  require (' fs ');
 var   csv  =  require (" fast - csv ");
 var   mapdata  =  require ('. / Longitude   and   latitude
 mapping   in   the   site   list . json ');
 var   file  = '. / Site_20170  101 - 20170202 / china_site
 s_20170120 . csv ';
 var   filename  =  file . replace (/^. *[\\\/]/, ''). split ('.')[
 0 ] . split (' _ ')[ 2 ];
 var   datas  = {};
 csv
. fromPath ( file , {  headers :  true ,  objectMode :  true  })
. on (" data ",  function  ( data ) {
  if  ( data . type  === ' AQI ') {
    datas [ data . hour ] = [];
    for  ( var   key   in   data ) {
      if  ( mapdata [ key ]) {
        datas [ data . hour ]. push ({
          name :  mapdata [ key ]. name ,
          value : + data [ key ],
          code :  mapdata [ key ]. code ,
          city :  mapdata [ key ]. city ,
          lng : + mapdata [ key ]. lng ,
```

```
            lat : + mapdata [ key ]. lat
          })
        }
      }
    }
  }
})
. on (" end ",  function  () {
  fs . writeFile ('./ data /' +  filename  + '. json ',  JSON .
 stringify ( datas ));
  console . log (" done ");
});
```

Use the period of time for each day as the key, and the array as the value. The array contains the AQI information and position of each monitoring site of the corresponding periods. Then the data of each period of time for each day can be used in the isosurface layer widget.

```
{
" 0 ": [{ " name ": " Wanshouxig  ong ", " value ":  18 , " code ":
 " 1001A ", " city ": " Beijing ", " lng ":  116 . 366 , " lat ":
 39 . 8673  }, { " name ": " Dingling ", " value ":  25 , " code ":
 " 1002A ", " city ": " Beijing ", " lng ":  116 . 17 , " lat ":  40
 . 2865  }, ...],
" 1 ": [{ " name ": " Wanshouxig  ong ", " value ":  28 , " code ":
 " 1001A ", " city ": " Beijing ", " lng ":  116 . 366 , " lat ":
 39 . 8673  }, { " name ": " Dingling ", " value ":  65 , " code ":
 " 1002A ", " city ": " Beijing ", " lng ":  116 . 17 , " lat ":  40
 . 2865  }, ...],
" 2 ": [{ " name ": " Wanshouxig  ong ", " value ":  88 , " code ":
 " 1001A ", " city ": " Beijing ", " lng ":  116 . 366 , " lat ":
 39 . 8673  }, { " name ": " Dingling ", " value ":  95 , " code ":
 " 1002A ", " city ": " Beijing ", " lng ":  116 . 17 , " lat ":  40
 . 2865  }, ...]
 ...
}
```

2.2.3 Process the APIs

This topic describes how to change data on the isosurface layer with the timeline. An API or a database is needed to obtain data from different monitoring sites during different periods of time.

We recommend that you write an API.

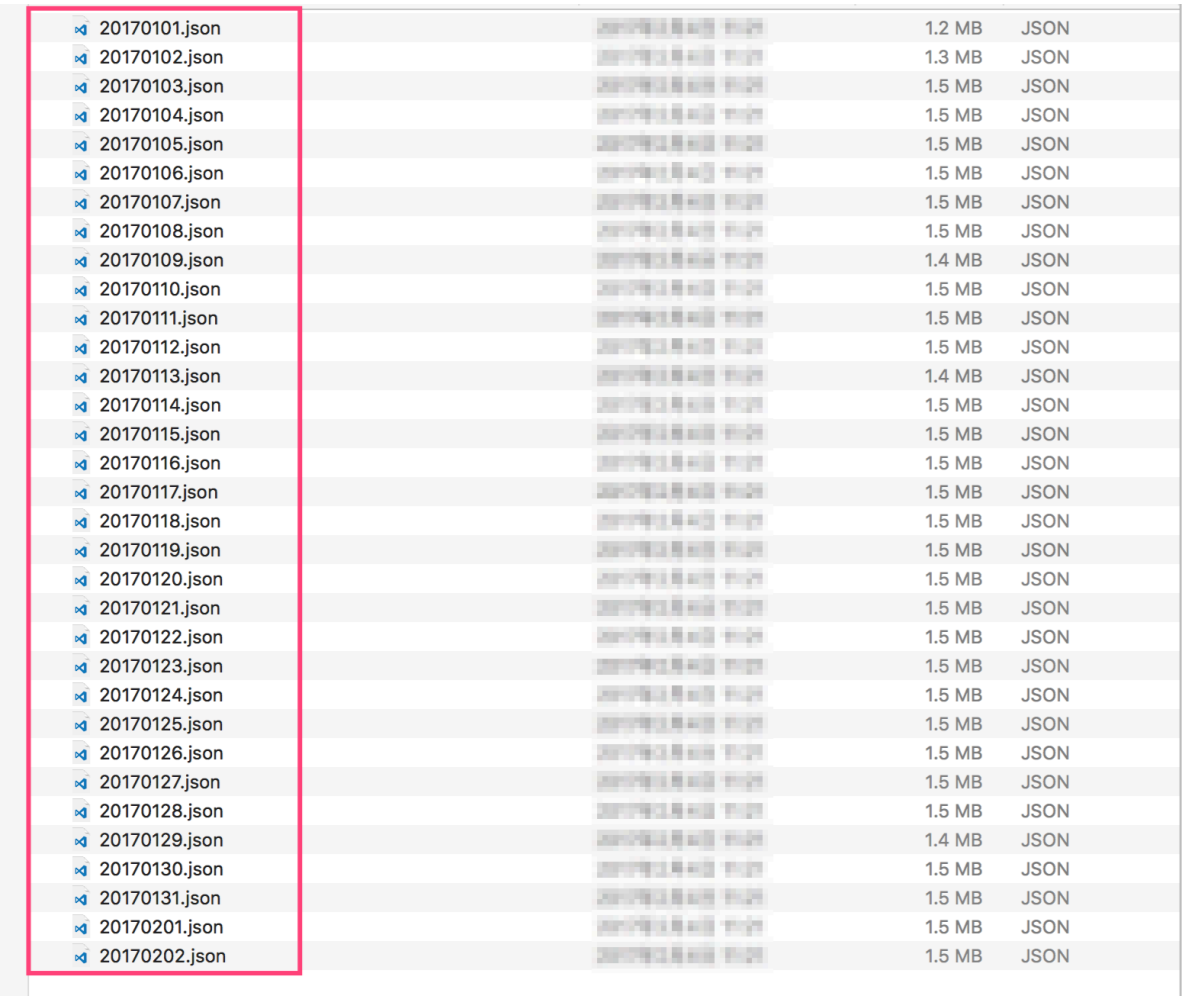
- Request address: /aqi
- Request method: GET
- Request parameter:
  - Parameter: `date`
  - Parameter type: string, for example, 2017012722. The format is YYYYmmDDHH.

1. **Process all the downloaded data. Node.js provides a glob module to process all data in the directory in batches.**

```
var   fs  =  require (' fs ');
var   csv  =  require (" fast - csv ");
var   glob  =  require (' glob ');
var   mapdata  =  require ('. / Longitude   and   latitude
mapping   in   the   site   list . json ');
glob (". / Site_20170  101 - 20170202 /*. csv ",  function  ( err
,  files ) {
files . forEach ( function  ( file ) {
 var   filename  =  file . replace (/^. *[\\\/]/, ''). split ('.')[
0 ] . split (' _ ')[ 2 ];
 var   datas  = {};
 csv
  . fromPath ( file , {  headers :  true ,  objectMode :  true  })
  . on (" data ",  function  ( data ) {
     if  ( data . type  === ' AQI ') {
       datas [ data . hour ] = [];
       for  ( var   key   in   data ) {
         if  ( mapdata [ key ]) {
           datas [ data . hour ]. push ({
             name :  mapdata [ key ]. name ,
             value : + data [ key ],
             code :  mapdata [ key ]. code ,
             city :  mapdata [ key ]. city ,
             lng : + mapdata [ key ]. lng ,
             lat : + mapdata [ key ]. lat
          })
        }
      }
    }
  })
  . on (" end ",  function  () {
     fs . writeFile ('./ data /' +  filename  + '. json ',  JSON .
stringify ( datas ));
     console . log (" done ");
  });
});
```

```
});
```

The result is as follows.

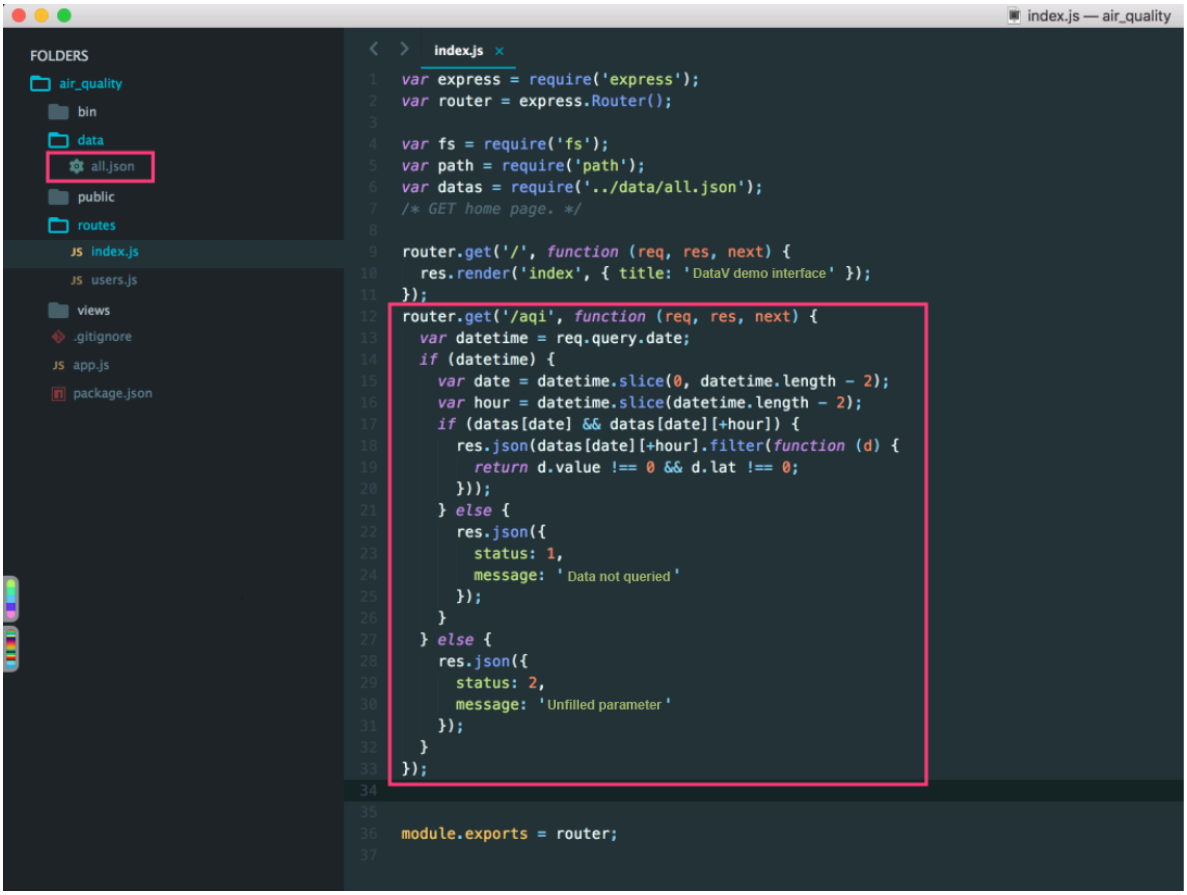


2. Use the glob module to integrate the data. Use the file name (which is a date) as the key, and the corresponding content as the value. Then, you will obtain an integration file named *all . json* .

```
// The   following   method   is   not   suitable   to   process
 data   in   large   scale .
 var   fs   =  require (' fs ');
 var   csv   =  require (" fast - csv ");
 var   glob   =  require (' glob ');
 glob ("./ data /*. json ",  function   ( err ,  files ) {
 var   datas   = {};
 files . forEach ( function   ( file ) {
  var   filename   =  file . replace (/^. *[\\\/]/, ''). split ('.')[
 0 ];
  datas [ filename ] =  require ( file );
});
 fs . writeFile ('./ data / all . json ',  JSON . stringify ( datas
 ));
 console . log (' done ');
```

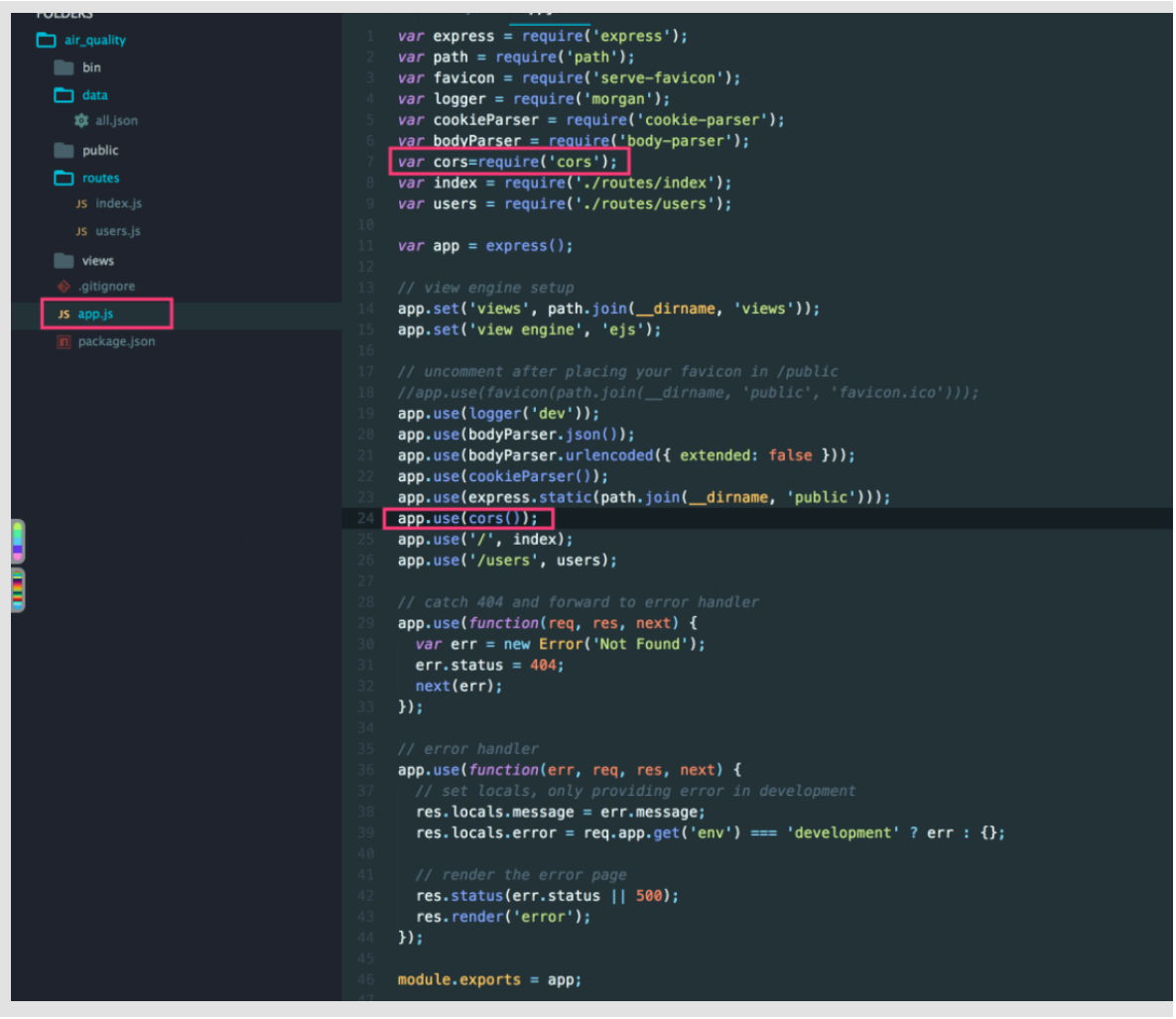
```
});
```

3. Use the express framework of Node.js to initialize an express project, and add an API according to the preceding API requirements.

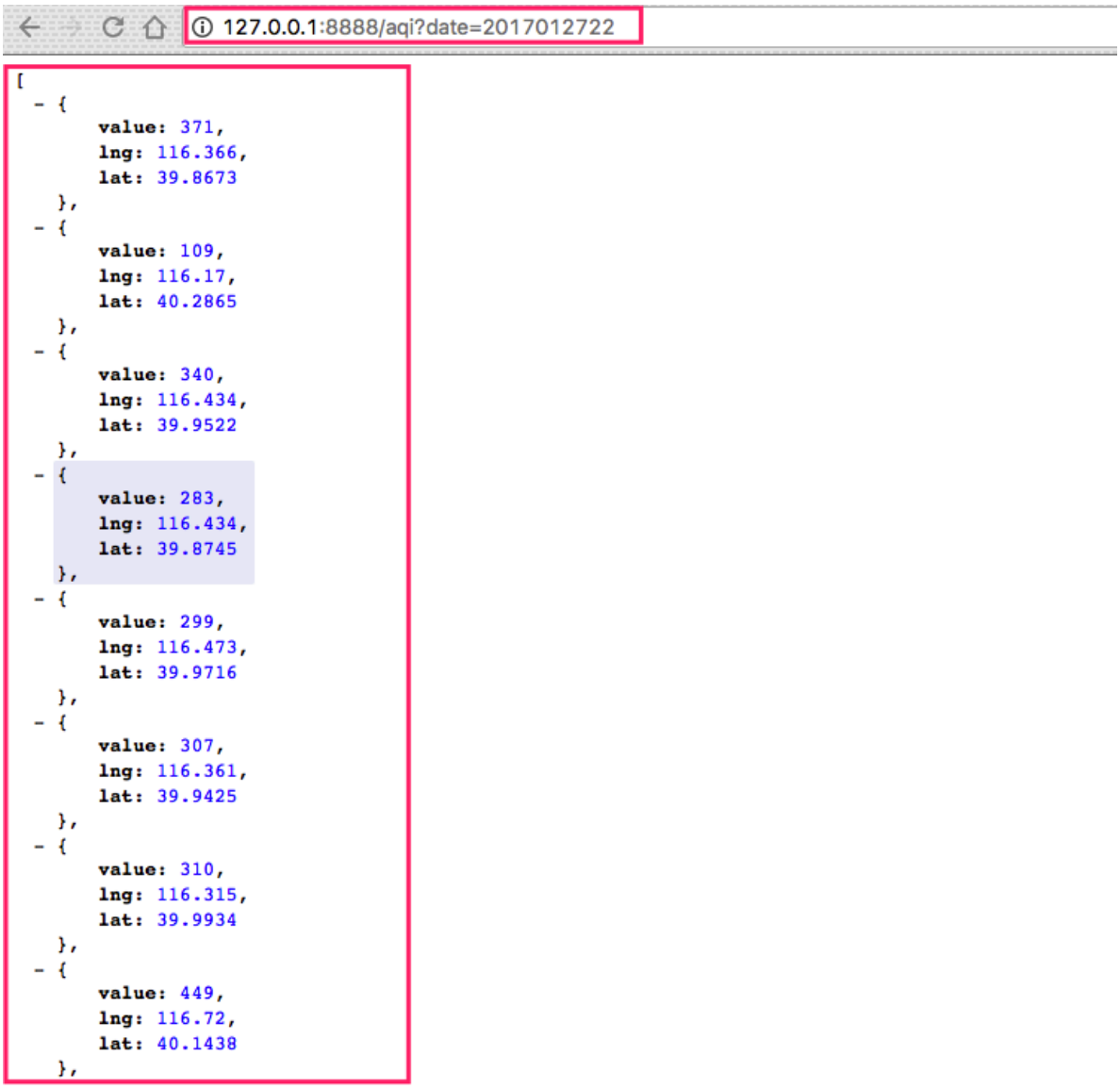


Note:

To avoid cross-domain requests, you can add a cors module to the `app.js` file.


air_quality
bin
data
all.json
public
routes
index.js
users.js
views
.gitignore
app.js
package.json
var express = require('express');
var path = require('path');
var favicon = require('serve-favicon');
var logger = require('morgan');
var cookieParser = require('cookie-parser');
var bodyParser = require('body-parser');
var cors=require('cors');
var index = require('./routes/index');
var users = require('./routes/users');
var app = express();
// view engine setup
app.set('views', path.join(__dirname, 'views'));
app.set('view engine', 'ejs');
// uncomment after placing your favicon in /public
//app.use(favicon(path.join(__dirname, 'public', 'favicon.ico')));
app.use(logger('dev'));
app.use(bodyParser.json());
app.use(bodyParser.urlencoded({ extended: false }));
app.use(cookieParser());
app.use(express.static(path.join(__dirname, 'public')));
app.use(cors());
app.use('/', index);
app.use('/users', users);
// catch 404 and forward to error handler
app.use(function(req, res, next) {
var err = new Error('Not Found');
err.status = 404;
next(err);
});
// error handler
app.use(function(err, req, res, next) {
// set locals, only providing error in development
res.locals.message = err.message;
res.locals.error = req.app.get('env') === 'development' ? err : {};
// render the error page
res.status(err.status || 500);
res.render('error');
});
module.exports = app;

4. After processing the API, run the `npm start` command to test the API.



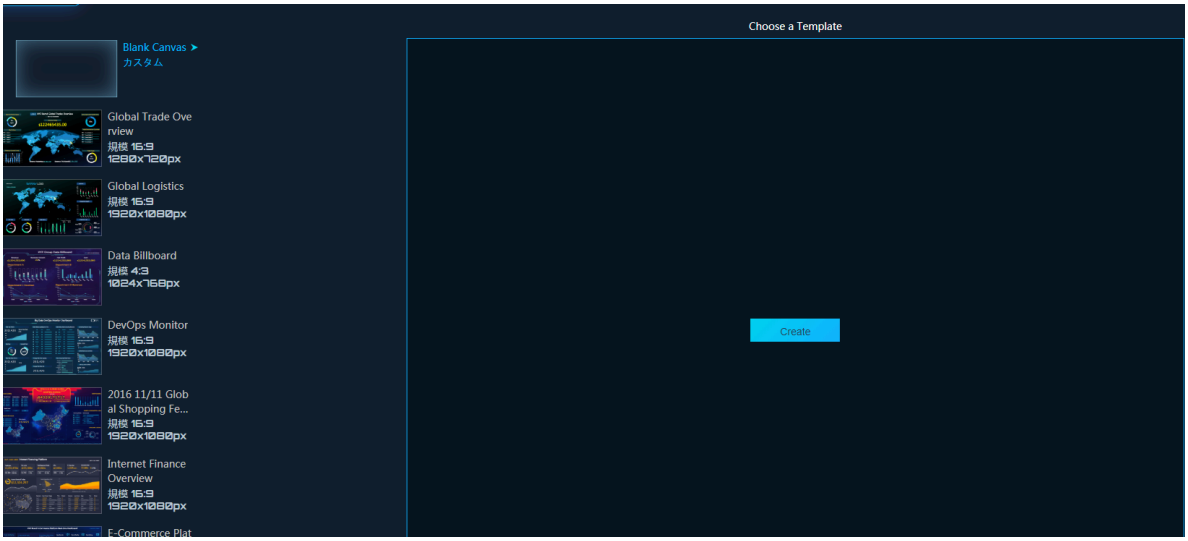
2.3 Create a project

2.3.1 Create a project

> Note:
> The data source used in this example is a local API file. Therefore, you do not need to add a data source. You can directly call the API from the widgets of the project. To use another data source, you must add the data source to DataV before you can create a project.

1. Log on to the DataV console.
2. Choose Projects > Create Project.

3. Select the blank template and click Create.



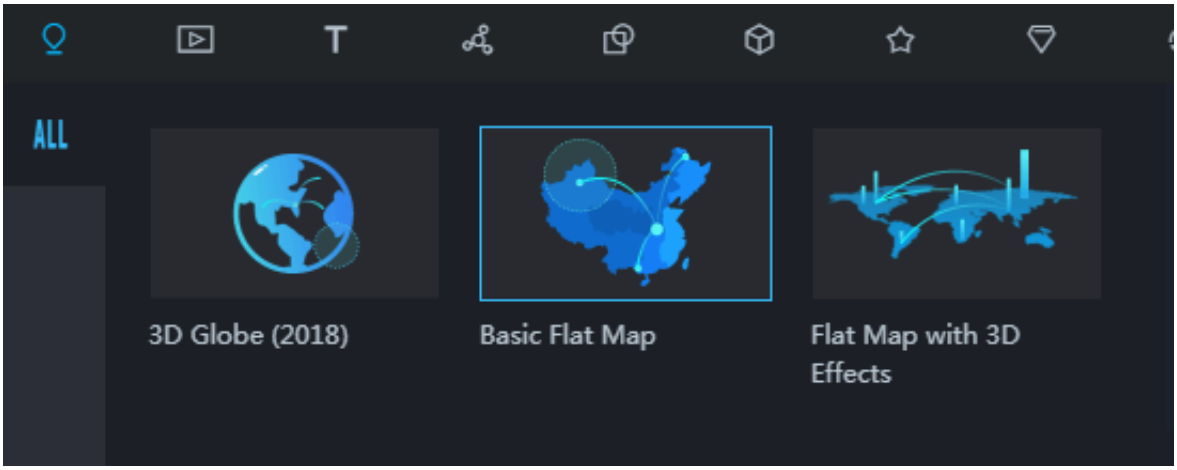
4. Enter a name for the project and click Create.

   After your project is successfully created, the project editor page is displayed.

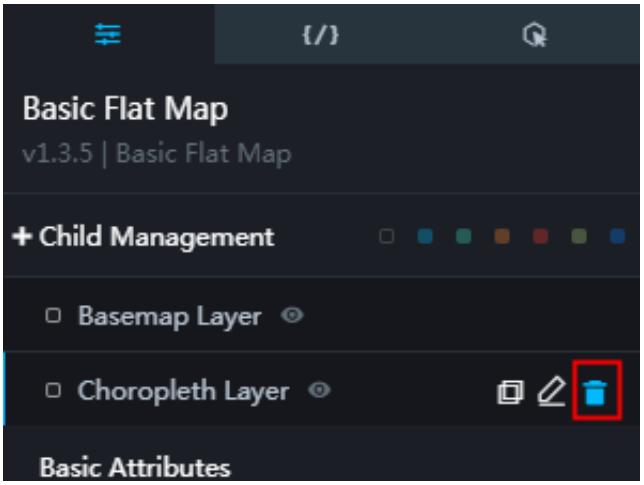
2.3.2 Add widgets

Add a map widget and child widgets

1. On the project editor page, choose Maps > Basic Flat Map.

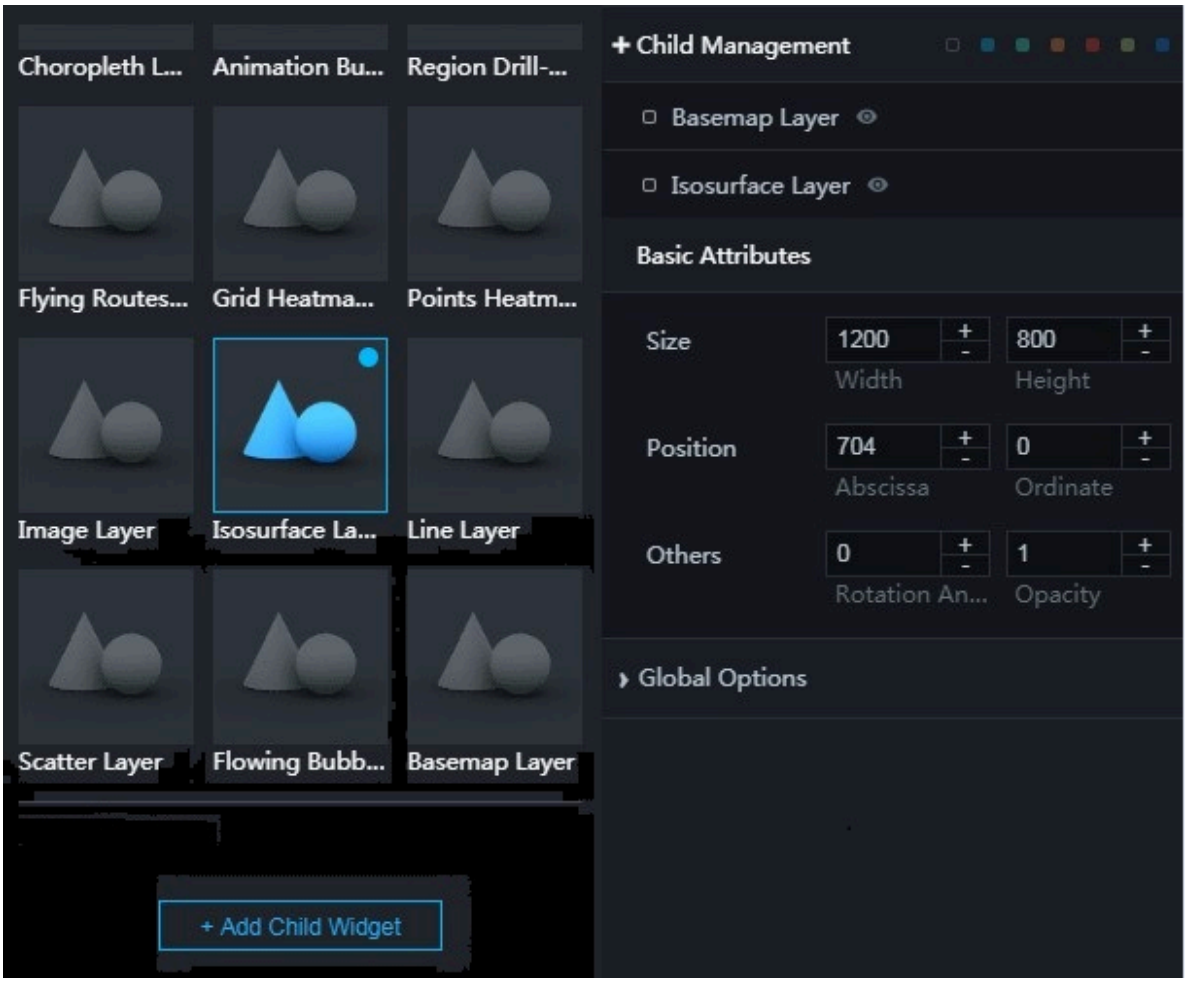


2. On the Configuration pane, delete all the child widgets except the basemap layer widget.



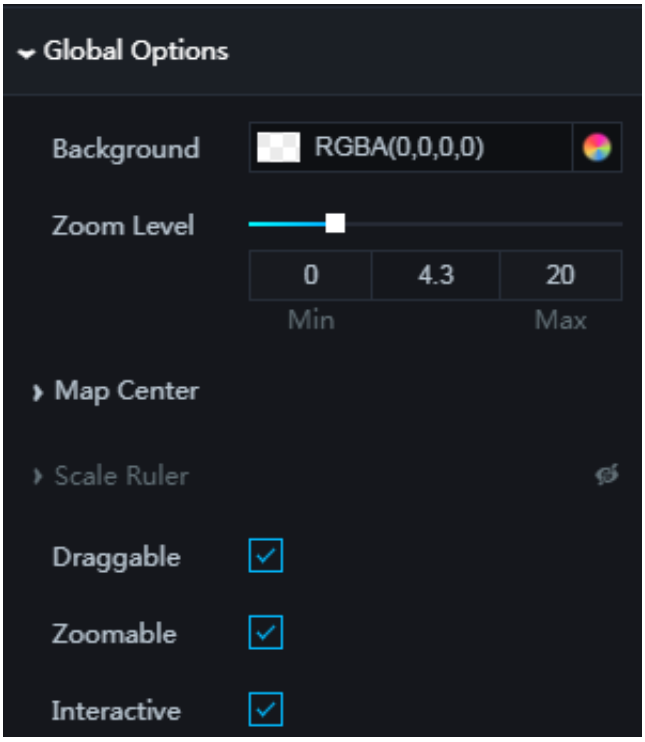
3. Add the isosurface layer widget.

   You can click + next to Child Management, select Isosurface Layer, and click Add Child Widget.



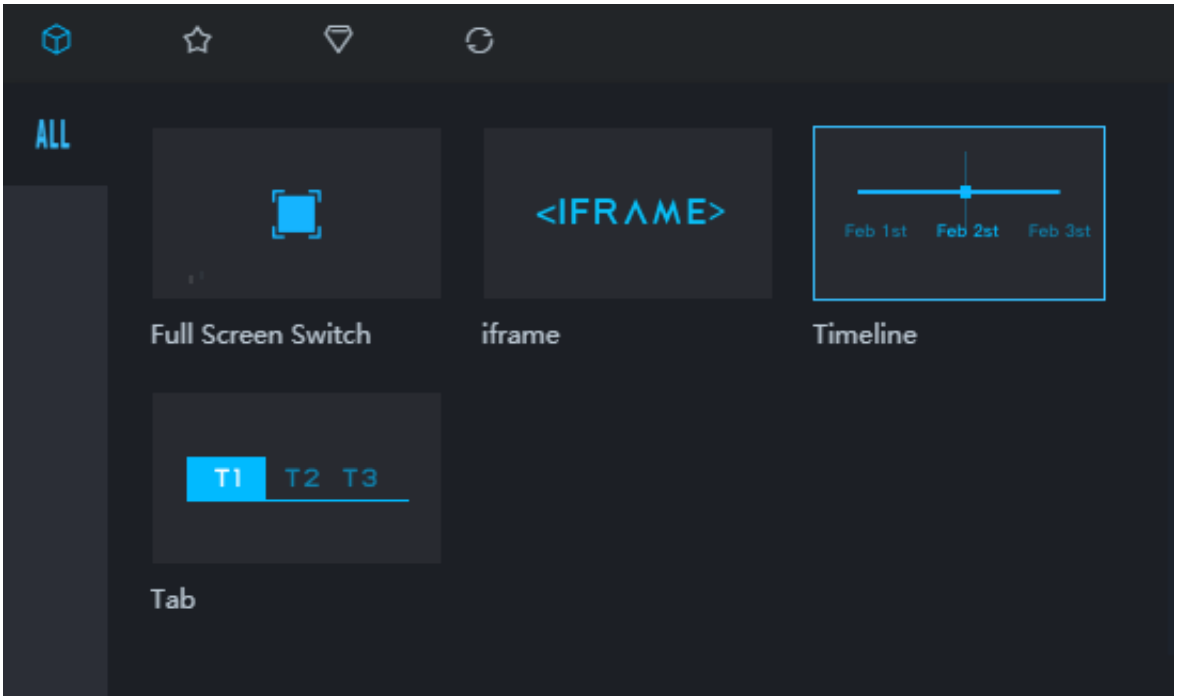
4. Click Global Options and adjust the size of the map.

   You can drag the slider or enter a value to adjust the map size and display area.



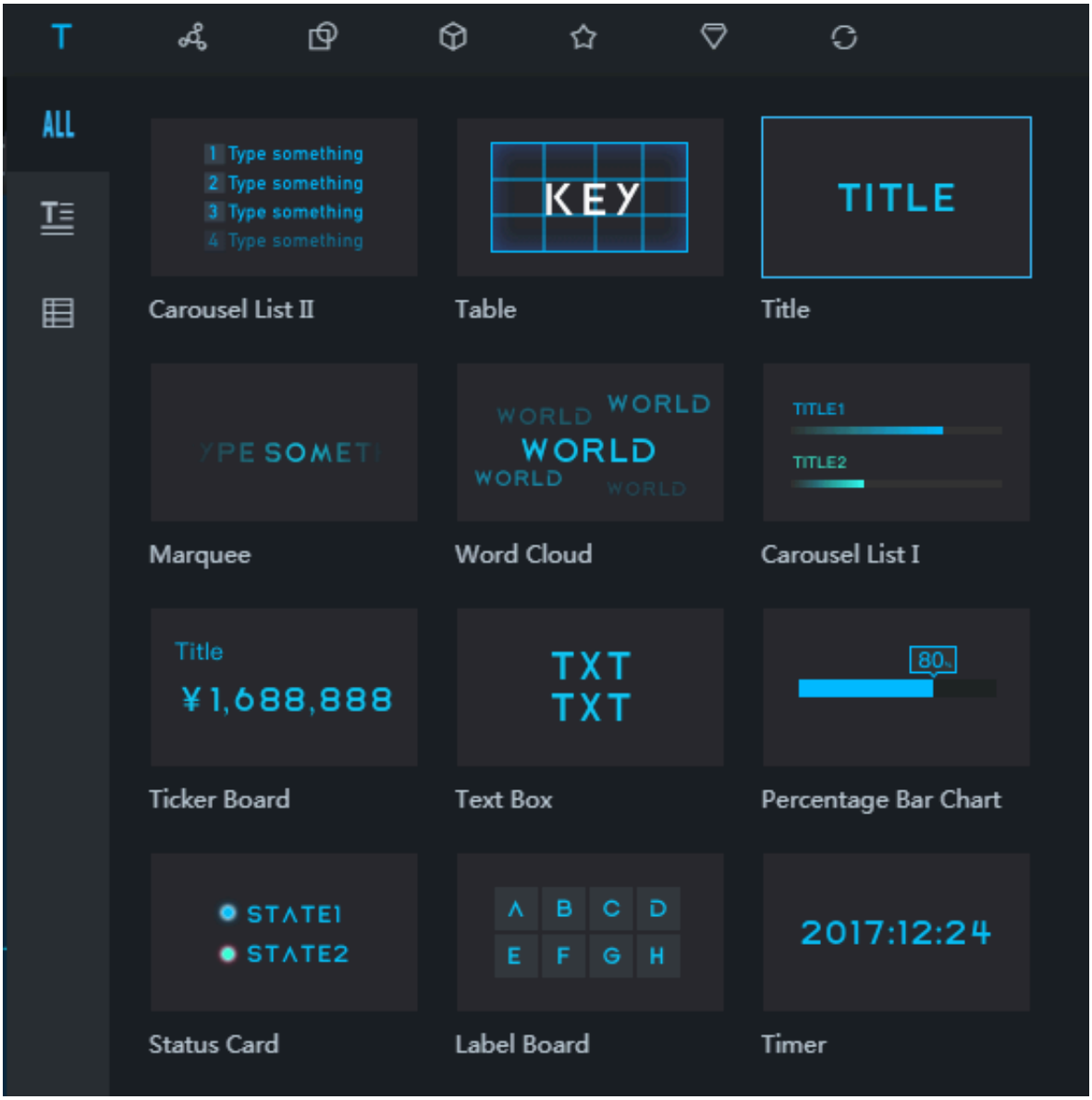
Add a timeline

Choose Interact > Timeline to add a timeline to the map.



Add a title for the map

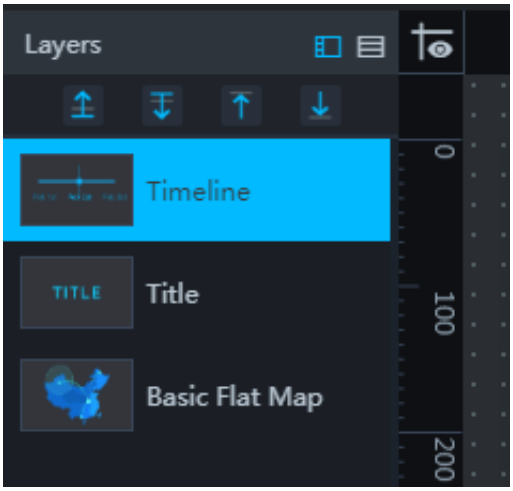
Choose Text > Title to add a title to the map.



Adjust the layers and position

After adding the widgets, you can:

- Adjust the sequence of the layers, select a layer, and change the title of a layer on the Layers pane.



- Adjust the size and position of the widgets on the Configuration pane.

You can also select a widget and drag the widget on the canvas to adjust its position .

2.3.3 Add data

Add data for the map

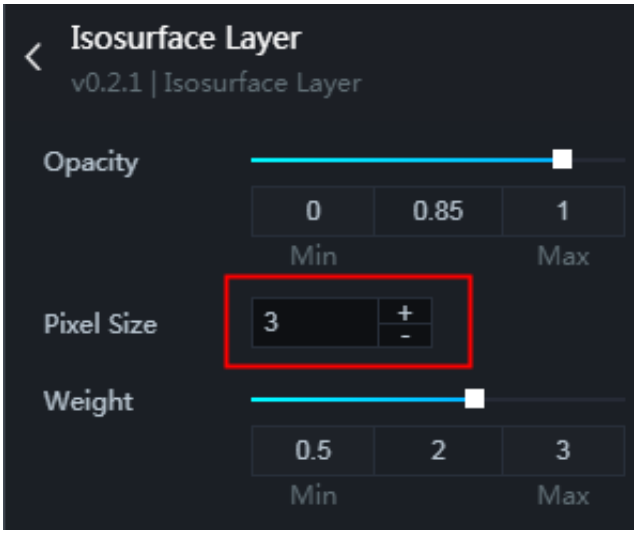
1. On the project, click the map widget.
2. Click the Data pane.

3. On the Child Management tab page, click Isosurface Layer.

   The data used in this example is obtained from all across China. You can use the data directly or modify the data as needed.
4. Click Interpolation Points Data.
5. Configure the data.
   - Data Source Type: The APIs have been specified in Process the APIs and the data has been tested. Therefore, set the data source type to API.
   - URL: Enter the API test URL (`http://127.0.0.1:8888/aqi?date=2017012722`).
6. Click View Data Response. The data response is displayed and the data has been successfully matched.

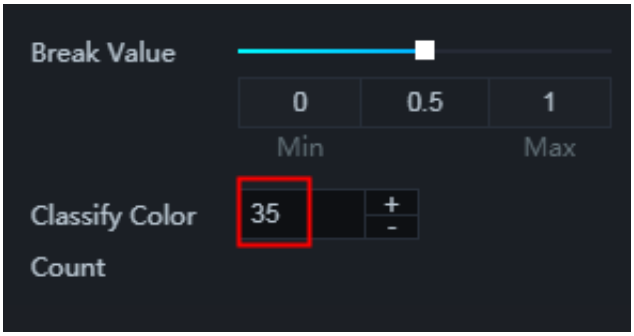
7. Set the style of the isosurface layer widget.
   a. Click the Configuration pane.
   b. Set the Pixel Size. The recommended value is 3.

      Setting larger values for the pixel size allows interpolation to work faster, but also reduces the precision of interpolation results.



   c. Set the Render Type. The recommended type is Linear.

   d. Set the Classify Color Count. The recommended value is 35.



Add data for the timeline

1. On the project, click the timeline widget.
2. Click the Data pane.

3. Set Data Source Type to Static Data.
4. Create the required data according to the examples and replace the static data on the data pane of the timeline widget.

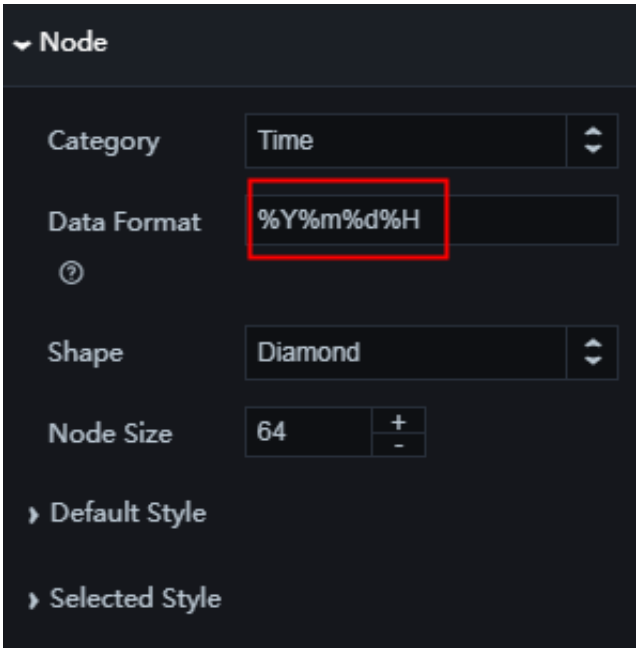
   For example, you can use the data obtained each day at 22:00 from January 22, 2017 to February 2, 2017 as the timeline data.

```
[
{
 " name ": " 22 : 00 ,  January   22 ,  2017 ",
 " date ":  2017012222 ,
 " value ":  2017012222
},
{
 " name ": " 22 : 00 ,  January   23 ,  2017 ",
 " date ":  2017012322 ,
 " value ":  2017012322
},
{
 " name ": " 22 : 00 ,  January   24 ,  2017 ",
 " date ":  2017012422 ,
 " value ":  2017012422
},
{
 " name ": " 22 : 00 ,  January   25 ,  2017 ",
 " date ":  2017012522 ,
 " value ":  2017012522
},
{
 " name ": " 22 : 00 ,  January   26 ,  2017 ",
 " date ":  2017012622 ,
 " value ":  2017012622
},
{
 " name ": " 22 : 00 ,  January   27 ,  2017 ",
 " date ":  2017012722 ,
 " value ":  2017012722
},
{
 " name ": " 22 : 00 ,  January   28 ,  2017 ",
 " date ":  2017012822 ,
 " value ":  2017012822
},
{
 " name ": " 22 : 00 ,  January   29 ,  2017 ",
 " date ":  2017012922 ,
 " value ":  2017012922
},
{
 " name ": " 22 : 00 ,  January   30 ,  2017 ",
 " date ":  2017013022 ,
 " value ":  2017013022
},
{
 " name ": " 22 : 00 ,  January   31 ,  2017 ",
 " date ":  2017013122 ,
 " value ":  2017013122
},
{
 " name ": " 22 : 00 ,  February   1 ,  2017 ",
```

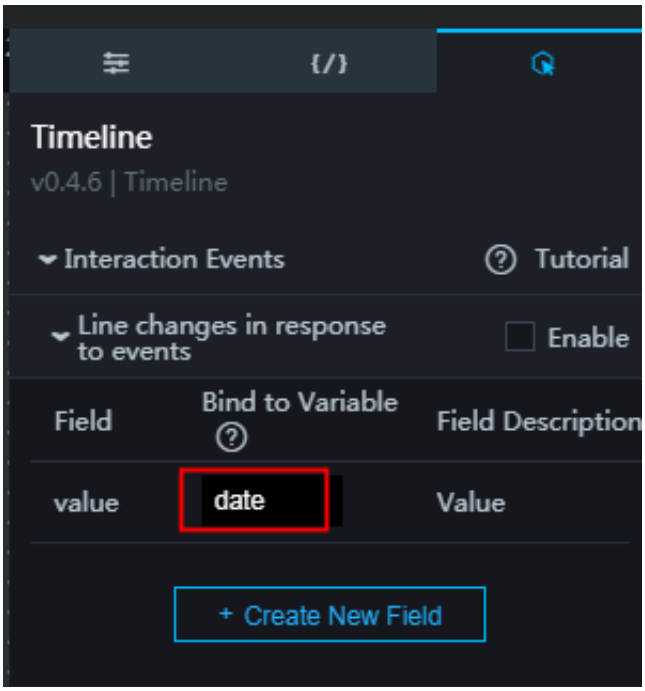
```
 " date ":  2017020122 ,
 " value ":  2017020122
},
{
 " name ": " 22 : 00 ,  February   1 ,  2017 ",
 " date ":  2017020222 ,
 " value ":  2017020222
}
]
```

- `name` : displayed content in a node of the timeline
- `date` : date in the timeline, which can be used as a callback ID
- `value` : date in the timeline

5. Set the timeline style.
   a. Click the Configuration pane.
   b. Click Node and set Data Format to %Y%m%d%H.



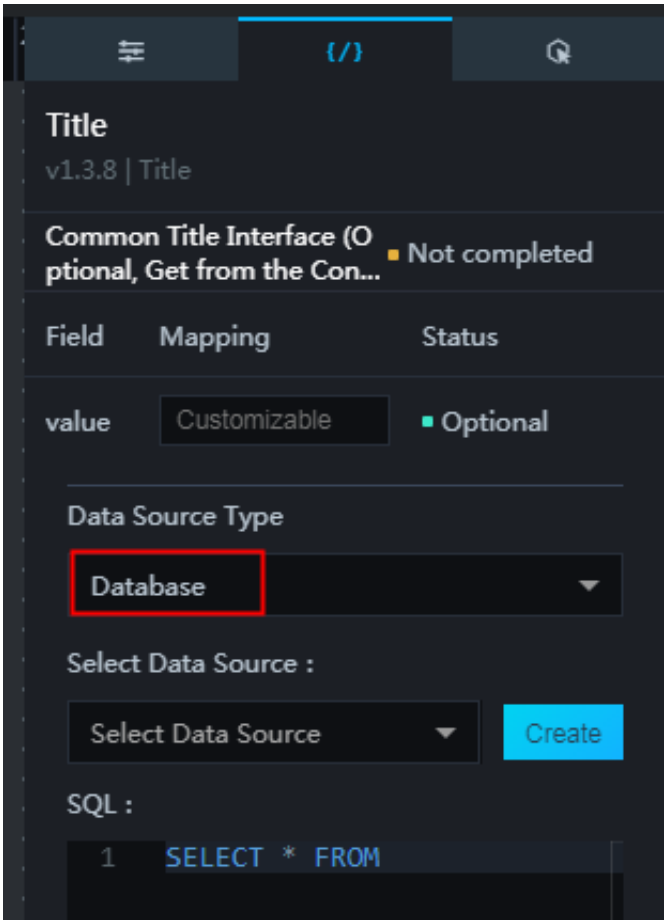
   c. Click Interaction and set the value of the callback ID to `date`.



Set the map title

1. On the project, click the title widget.
2. Click the Data pane.

3. Set Data Source Type to Database.



4. In the Select Data Source list, select a database.

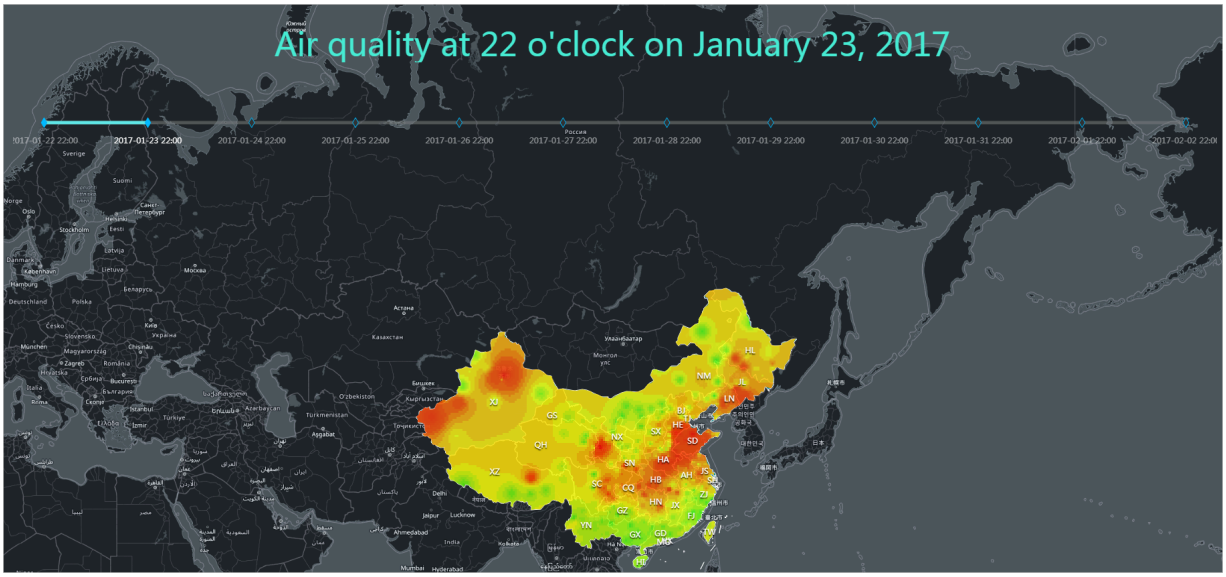
   If no database is available, you can click Create to create a database as promoted. For more information about how to create a database, see Configure data sources.

5. Enter the following command in the SQL area:

```
select   to_char ( to_timesta  mp (: date ,' YYYYMMDDHH  24 '),'
YYYY  ( year )  mm  ( month )  DD  ( day )  HH  ( 24 - hour
format )')||' air   quality '  as   value ;
```

   `: date` : actual value corresponding to the callback ID

You can also add legends as needed. The following figure shows the display effect of the project.



2.4 View a project

You can view a project after you publish it according to Publish a project. The following figure shows the display effect a project.