Alibaba Cloud
DataV数据可视化

Developer Guide

Issue: 20200318

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, consequent**

ial, 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 contents in Alibaba Cloud documents, including but not limited to pictures, architecture design, page layout, and text description, 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 this document 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.

Document conventions

Style	Description	Example
	A danger notice 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.
	A warning notice 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 restart an instance.
	A caution notice indicates warning information, supplementary instructions, and other content that the user must understand.	 **Notice:** If the weight is set to 0, the server no longer receives new requests.
	A note indicates supplemental instructions, best practices, tips, and other content.	 **Note:** You can use Ctrl + A to select all files.
>	Closing angle brackets are used to indicate a multi-level menu cascade.	Click Settings > Network > Set network type.
Bold	Bold formatting is used for buttons, menus, page names, and other UI elements.	Click **OK**.
`Courier font`	Courier font is used for commands.	Run the `cd /d C:/window` command to enter the Windows system folder.
Italic	Italic formatting is used for parameters and variables.	`bae log list --instanceid` *`Instance_ID`*
[] or [a|b]	This format is used for an optional value, where only one item can be selected.	`ipconfig [-all|-t]`

Style	Description	Example
{} or {a|b}	This format is used for a required value, where only one item can be selected.	`switch {active|stand}`

Contents

1 Quick start

Prepare the environment

1. **Before getting started, you need to install Node.js by visiting the *Node.js website*. We recommend that you install Node.js version 8.0.0 or later (however, versions no later than 10.12.0 is recommended).**

   Note:
   If you have installed Node.js of another version and want to use it for other purposes, we recommend that you install NVM to manage your Node.js versions. With NVM, you can install a new version of Node.js without uninstalling the existing one. The installation address is as follows: *GitHub - creationix/nvm: Node Version Manager - Simple bash script to manage multiple active node.js versions*.

2. **After installing Node.js, run the `node -v` and `npm -v` commands in the Command Line Interface (CLI) to view the Node and NMP versions. (If you are using macOS, run the commands in terminal. If you are using Windows, run the commands in cmd.)**

Install the SDK

1. **Run the following command to install the SDK: (If you are using macOS, run the command in terminal. If you are using Windows, run the command in cmd.)**

   ```
   npm install --registry=https://registry.npm.taobao.org datav-cli -g
   ```

2. **After installing the SDK, run the `datav --version` command to view the SDK version.**

Set the language

If your default language is not English, run the following command in the CLI to set the language to English:

```
bash
datav locale
? What is your language? [English\Chinese\Japanese]
```

If you want to automatically switch the language according to your account domain, run the `datav locale-clear | datav lc` command after logging on to the DataV (Developer Edition) console.

You can also run the `datav locale` command to reset the language.

Log on to the DataV (Developer Edition) console

Run the following command to log on to the DataV (Developer Edition) console: (If you are using macOS, run the command in terminal. If you are using Windows, run the command in cmd.)

```
datav login

? Username: [Enter your user name displayed on the upper-right corner
 of the DataV console. If you are using a RAM account, enter the user
name of your Alibaba Cloud account.]
? Developer Token: [Enter the developer token obtained from the widget
 page in the DataV console.]
? Do you want to set an alias? (Y/n)
? Alias: [Enter an alias if necessary.]
Configuration has been completed.
```

In the CLI, if Configuration has been completed. is displayed, the logon is successful.

Note:
You do not need to log on to the console if you want to create or preview a widget. However, you must log on to the console if you want to publish a widget.

Generate a widget package

Run the following command to generate a widget package: (If you are using macOS, run the command in terminal. If you are using Windows, run the command in cmd.)

```
datav init

? Widget Name (It can contain letters, numbers, and hyphens (-). If
the widget name you specified does not exist, a folder with the same
 name is created locally): [A widget name can only contain letters,
numbers, and hyphens (-).]
? Widget Display Name: [The display name is the name of your widget in
 the widget list of a project.]
? Widget Description:
? Select Template from Widget Template List (Use arrow keys)
```

In the CLI, if The widget has been created. is displayed, the widget package is generated successfully, as shown in the following figure.

Preview a widget

Run the following command to preview a widget: (If you are using macOS, run the command in terminal. If you are using Windows, run the command in cmd.)

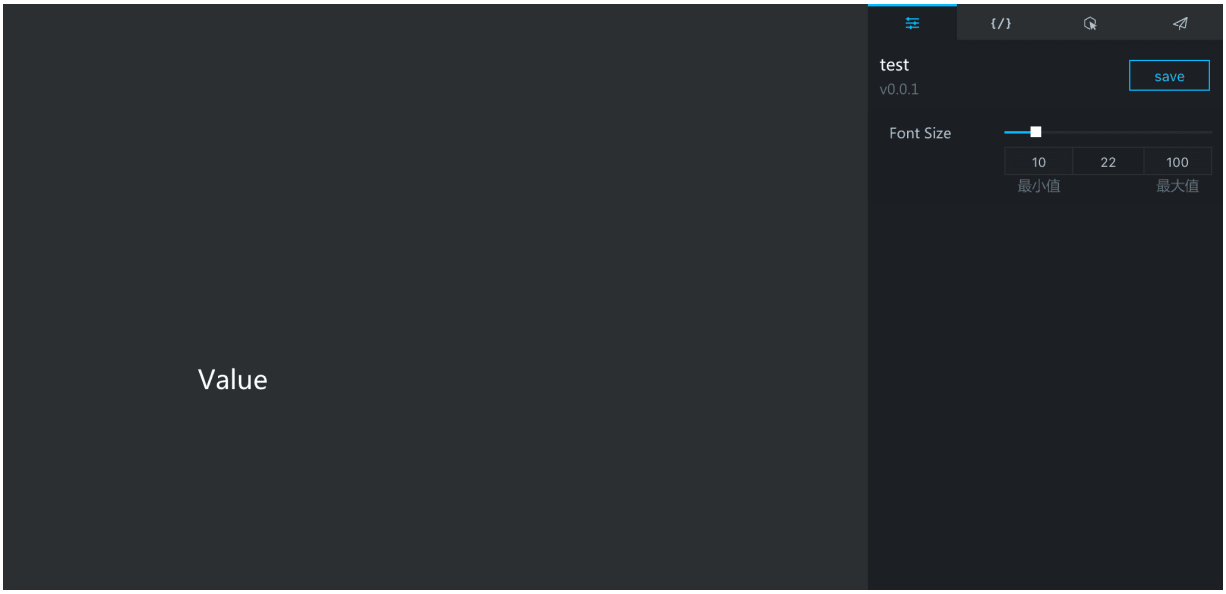
```
cd Widget Name
datav run
```

In the CLI, if The service is enabled. is displayed, the widget preview service is initiated. Your Google Chrome browser will automatically open and navigate to the widget preview page, as shown in the following figure.

Note:

- **If your browser does not open automatically, check whether you have installed Google Chrome. We recommend that you install Google Chrome and then manually open localhost:1111/ using the Chrome browser.**
- **If a port error is displayed, check whether port 1111 is occupied by other applications. You can run the `datav run -p 1112` command to initiate the widget preview service.**
- **If you have opened localhost:1111/ in your browser but no widget preview is displayed, check whether you have directed the localhost to 127.0.0.1 in the hosts file. You can also preview a widget by visiting 127.0.0.1:1111/ in your browser.**

If the Chrome browser opens successfully, the following page is displayed.



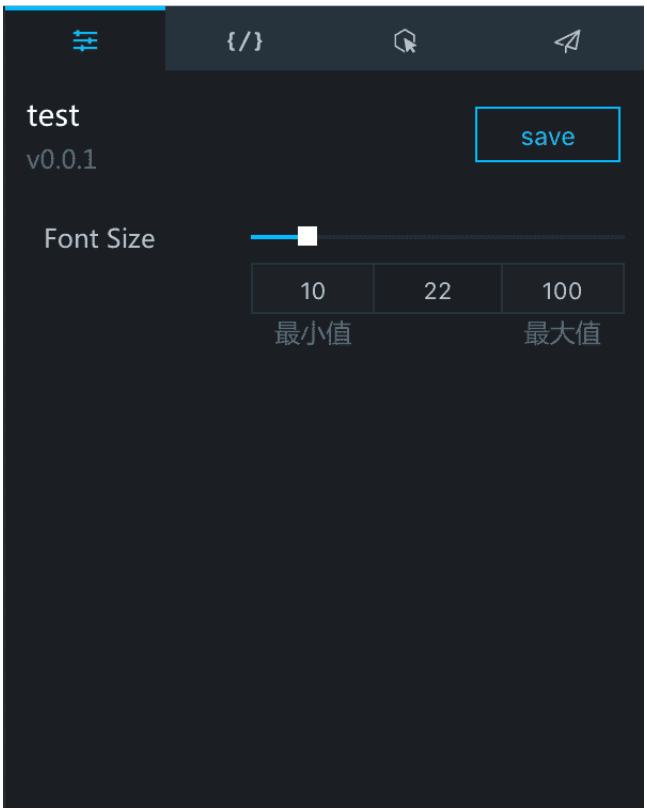
The preview page contains a canvas, a toolbar at the bottom, and a toolbar on the right.

- Canvas
  - The canvas is used to display changes of a widget.
  - All the configuration and data modifications on the right-side toolbar are displayed on the widget in the canvas in real time.
  - The widget border indicates the size of the widget container. You can resize the border to test the widget display.
- Right-side toolbar:

  The right-side toolbar contains four panes: Configuration, Data, Interaction, and Publish.

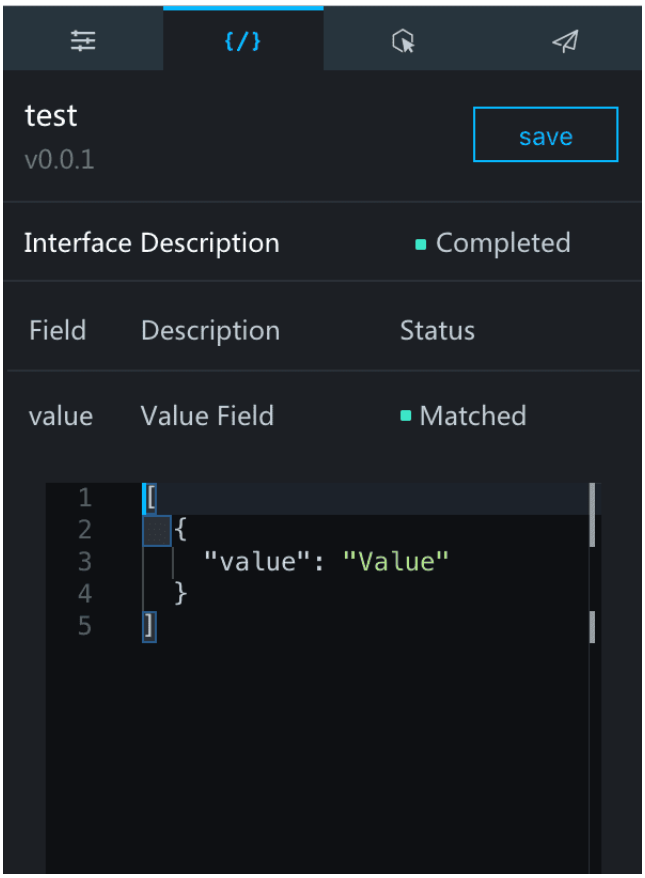
  - Configuration: The configuration pane allows you to configure key items for a widget as needed. The configuration takes effect immediately. For example, if you drag the slider on the right of Font Size, the font size of the displayed

text will be changed. To save the configuration, click Save in the upper-right corner. The configuration will be saved as the default configuration.

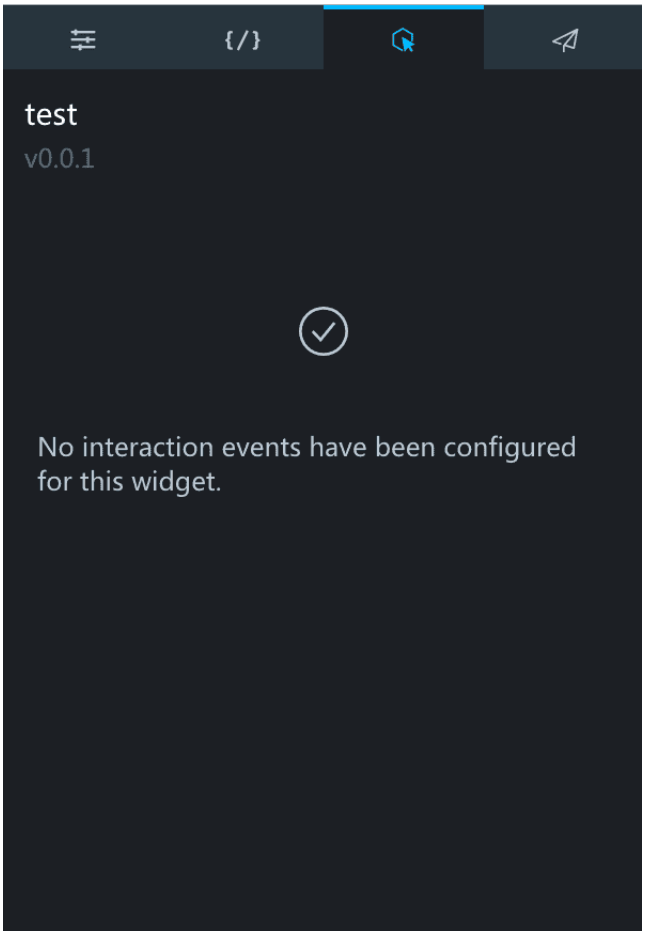


- **Data: The data pane allows you to configure the data APIs for a widget. The data changes will be displayed on the widget accordingly. To save the data**

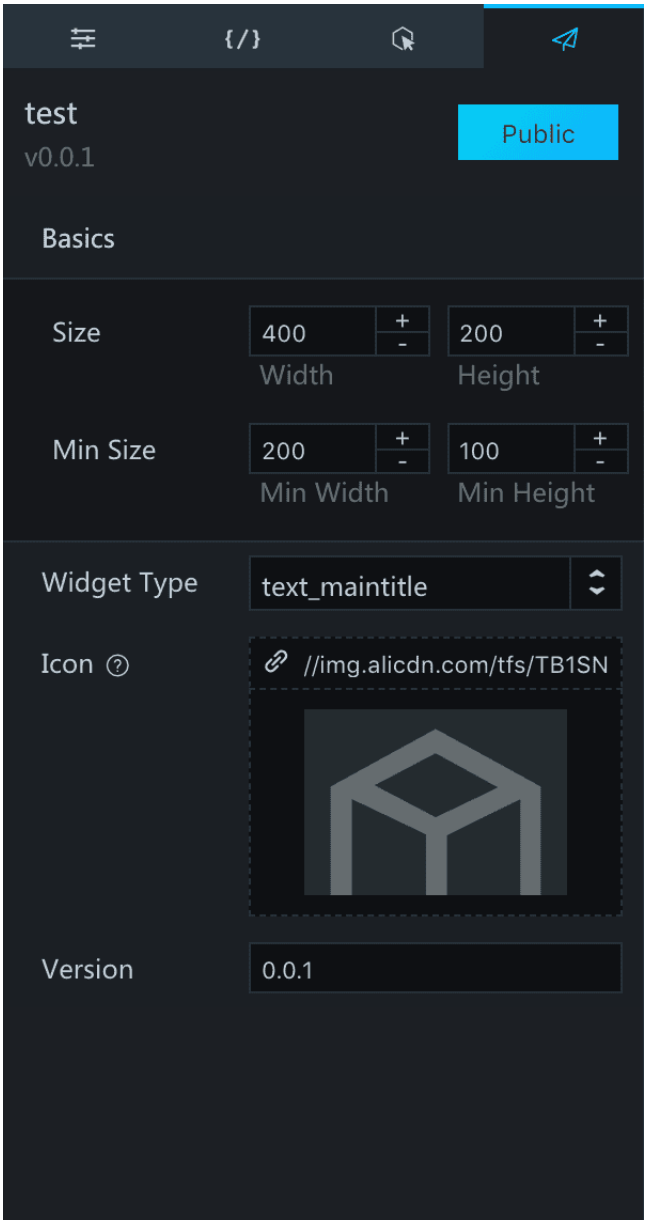
changes, click Save in the upper-right corner. The data will be saved as the default data of the widget.



- **Interaction: The interaction pane allows you to set the widget interaction.**



- **Publish: The publish pane displays the configuration of the widget type, the widget version, and icons. To publish a widget, click Publish in the upper-right corner.**



Publish a widget

You can publish a widget using one of the following methods:

- **Method 1 (recommended method)**

  Go to the directory address of the widget and run the `datav publish` command. The widget will be automatically compressed and published on the server where your account domain is located.

- **Method 2**

  Go to the directory address of the widget and run the `datav package` command. Find the .tar.gz package named `widget-version` outside the widget directory and upload the package to the widget page of datav.aliyun.com.

- **Method 3**

  Go to the publish page of Preview Widget and click Publish.

2 Document structure

2.1 Overview

```
--coms-name //Widget name
|--index.js //Widget file that can be directly called by users
|--package.json //Widget configuration
|--readme.md //Widget description
|--history.md //Widget update history
```

2.2 index.js specifications

This topic explains index.js specifications using an index.js template. For more information, see *widget examples*.

Common functions

- `render(array: data, object: config) || updateOptions(object: config)`

  The `render` function is the default rendering method. After a widget is initialized, the widget rendering logic is called. The input parameters are data and config, and the config parameter is optional. The `updateOptions` function receives the updated configuration items as parameters to change the rendering effect.

  - **Both render and updateOptions are rendering methods. If render supports `config` (the second parameter), updateOptions is optional.**
  - **If the `updateOptions` function exists, updateOptions (not `render`) will receive the updated configuration items.**
  - **Re-rendering must be supported. To ensure the same rendering effect, the values of data and config input must be the same input values.**

  Note:
  If the `handler` function is set for the configuration items, this function receives the updated configuration items. You can set the `handler` function to improve the availability of the widgets with complex configuration items.

- `emit(event_name, value)`

  With this function, you can set an event name and a global parameter using a widget in *`package.json`*. For other widgets that are in the same project as the

preceding widget, you can obtain the value of the global parameter by calling the global parameter name using a colon (:) during data configuration.

> **Notice:**
> **The value of `value` must be an object and cannot be a value.**

- `resize(int: width, int: height)`

  This function is called to drag or resize a widget.

- `clear()`

  This function is called to clear a widget.

- `destroy()`

  This function is called to delete or remove a widget.

- `require(*)`

  This function supports JavaScript and HTML.

2.3 package.json specifications

Specifications example of protocol version 2

> **Note:**
> **Pay attention to the notes in the following example:**

```
{
"name": "@leaf/pie-multi-radius-with-title",
 //@Namespace/Widget package name (The namespace is the name of
a widget package. A widget package cannot be published without a
namespace.)
"version": "0.1.5",                              //Version number
"dependencies": {                                //Dependencies,
excluding datav:/com/another datav widget
"bcore": "0.0.9",
"jquery": "2.1.4",                               //All the dependent
version numbers must be locked.
"lodash": "4.6.1"
},
"datav": {                                       //DataV configuration
"cn_name": "doughnut chart with title",                    //Chinese
name of a widget
"protocol": 2,                                   //Protocol version
number
"type": ["regular_pie"],                         //Widget type
"view": {
    "width": 300,
    "height": 200,
    "minWidth": 100,
    "minHeight": 50
},
```

```
"icon": "",                                          //Icon URL of a
widget
"apis": {                                          //Widget API. Multiple
APIs are supported.
  "source": {                                      //API name
    "handler": "render",                           //Processes the widget
function name returned from an API.
    "description" : "doughnut chart API",            //API description
    "fields" : {                                   //Fields required by an
 API. Multiple fields are supported.
      "x" : {                                      //Field name
        "description" : "category",                  //Field descriptio
n
        "type" : "string",                         //Field type
        "optional": true                           //Optional field
      },
      "y" : {
        "description" : "value",
        "type" : "int"
      }
    }
  }
},
"config" : {                                     //Widget configuration,
which is identified by the Editor
  "paxis" : {
    "type" : "group",                              //Type:group. For more
information, see <Description of the config field>.
    "name" : "label",
    "children" : {
      "dx" : {
        "type" : "text",                 //Type:text. For more
information, see <Description of the config field>.
        "name" : "distance from label to center",
        "default" : 220
      }
    }
  },
  "title" : {
    "type" : "group",
    "name" : "title",
    "children" : {
      "value" : {
        "hasVisibility" : "true",
        "visible" : "true",
        "type" : "text",
        "name" : "title",
        "default" : "This is title."
      },
      "font-size" : {
        "type" : "number",                  //Type:number. For more
information, see <Description of the config field>.
        "name" : "font",
        "min" : 10,
        "default" : 32,
        "max" : 100
      },
      "text-align" : {
        "name" : "alignment",
        "type" : "select",                 //Type:drop-down list. For
more information, see <Description of the config field>.
        "options" : [
          {"name" : "left aligned", "value" : "left"},
          {"name" : "right aligned", "value" : "right"},
```

```
          {"name" : "centered", "value" : "center"}
        ],
        "default" : "center"
      },
      "color" : {
        "name" : "font color",
        "type" : "color",                  //Type:color. For more
information, see <Description of the config field>.
        "default" : "#fff"
      },
      "background-color" : {
        "name" : "background",
        "type" : "color",
        "default" : "#000"
      }
    }
  }
},
"api_data":{                            //API data. Multiple APIs are
supported.
  "source" : [                         //API name, which must the same
as that in the apis field. The recommended data size is within 6 KB.
    {"x": "cargo", "y" : 5},
    {"x": "cargo", "y" : 22},
    {"x": "light goods", "y" : 22},
    {"x": "device", "y" : 14},
    {"x": "mineral", "y" : 15},
    {"x": "metal", "y" : 15},
    {"x": "building material", "y" : 12},
    {"x": "food", "y" : 12},
    {"x": "grain", "y" : 28}
  ],
},
"events": {                             //Global parameter (event
configuration)
  "event-name": {                       //Event name
  "description": "event description",        //Event description
    "fields": {                        //Field. Multiple fields are
supported.
      "value": {                       //Field name
        "description": "value description"       //Field description
      }
    }
  }
}
}
}
```

Description of the config field

The `config` field describes the configuration items of a widget. The configuration items can be transferred to a widget or be used to describe the options of the Editor.

The following requirements must be met to set the `config` field:

- **The default settings of a widget must be included.**
- **The input specifications must be clearly described for the Editor to identify.**

- `handler` field in the configuration items
  - If you have set `handler` in a configuration item of `config`, when the configuration item is changed, both the configuration item and the items contained in this configuration item are transferred to the specified custom function in handler.
  - Changes of this configuration item will not be transferred to the default render function. To use this configuration item in render or other functions, combine the configuration and this.config in the custom function.

```
"config": {
"example": {
  "type": "group",
  "name": "chart property",
  "handler": "updateBar",              //Custom function name
  "children": {
    "_innerRadius": {
      "type": "number",
      "name": "inner radius",
      "default": 0.4,
      "range": [
        0,
        1
      ]
    }
  }
}
}
```

- `type` field in the configuration items

Note:

Different Editor UIs are displayed according to different types. You can improve the widget quality and user experience by using the proper types and configuration items.

Type	Description
`group`	Configuration item group, which can contain multiple configuration items. We recommend that you place elements of the same type into the same group.
`text`	Text
`number`	Value
`select`	Drop-down list
`search`	Combo box

Type	Description
color	Color
multicolor	Gradient color (You can select a solid color or multiple colors as gradient colors.)
image	Image
array	Array (An array is used for multiple series and color cycling .)
hidden	Hidden configuration item (The Editor does not set hidden configuration items.)
boolean	Boolean value
radio	Check box
percent	Numerical value
imageSelect	Decorative elements

- type:group

```
"margin": {
"type" : "group",
"name" : "label",
"children" : {
}
}
```

Field name	Description	Required or not	Remarks
type	Type	Yes	N/A
name	Display name	Yes	N/A
children	Elements in a group	No	If this field contains no data , the group is empty.

- type:text

```
{
"type" : "text",
"name" : "title",
"default": "This is title."
}
```

Field name	Description	Required or not	Remarks
type	Type	Yes	N/A

Field name	Description	Required or not	Remarks
name	Display name	Yes	N/A
default	Default display value	No	If this field contains no data, the display value is empty.

- type:number

```
{
"type" : "number",
"name" : "font size",
"default": 22,
"min": 10,
"max": 55,
"range": [10, 55],
"range": {
"min" : 10,
"max" : 55
}
}
```

Field name	Description	Required or not	Remarks
type	Type	Yes	N/A
name	Display name	Yes	N/A
default	Default display value	N/A	If this field contains no data, 16 is displayed.
min	Minimum value	No	If one of the min, range[0], and range.min fields is set, the minimum value is successfully set. If the min, range[0], and range.min fields are not set, the value of type is set to text, and proper program running cannot be ensured.
max	Maximum value	No	If one of the min, range[1], and range.max fields is set, the maximum value is successfully set. If the min, range[1], and range.max fields are not set, the value of type is set to text, and proper program running cannot be ensured.

Field name	Description	Required or not	Remarks
range	Value range	No	This field can be an array (for example, [10, 55]) or an object (for example, {min: 10, max: 55}). If the range, min, and max fields are not set, the value of type is set to text, and proper program running cannot be ensured.

Note:
If a field has a maximum value or a minimum value, a slider is displayed on the UI.

- **type:select**

```
{
"name" : "alignment",
"type" : "select",
"options" : [
 {"name" : "left aligned", "value" : "left"},
 {"name" : "right aligned", "value" : "right"},
 {"name" : "centered", "value" : "center"}
 ],
"default" : "center"
}
```

Field name	Description	Required or not	Remarks
type	Type	Yes	N/A
name	Display name	Yes	N/A
default	Default display value	No	If this field contains no data, options[0] is displayed.
options	Options in a drop-down list	Yes	This field is an array, for example, [{ name: '' ,value: ''}]. If this field is not set, the value of type is set to text.

- **type:search**

```
{
"name": "font",
"type": "search",
"default": "Microsoft Yahei",
"range": [
```

```
{
  "Microsoft Yahei": "Microsoft Yahei"
},
{
  "SimSun": "SimSun"
},
{
  "SimHei": "SimHei"
},
{
  "LiSu": "LiSu"
},
{
  "YouYuan": "YouYuan"
},
"tahoma",
"arial",
"sans-serif"
],
"description": "Select a font that is already installed on your
system. If your system does not have this font, the default font
is used for the title."
}
```

Field name	Description	Required or not	Remarks
type	**Type**	**Yes**	**N/A**
name	**Display name**	**Yes**	**N/A**
default	**Default display value**	**No**	**If this field contains no data,** `options[0]` **is displayed.**
range	**Options in a drop-down list**	**Yes**	**This field can be in one of the following formats:** `[{name: '' , value: ''}]` **(an array),** `[{key: value }]`**, and** `[value]`**. If this field is not set, the value of type is set to text.**

- **type:color**

```
{
"name" : "font color",
"type" : "color",
"default" : "#fff"
}
```

Field name	Description	Required or not	Remarks
type	**Type**	**Yes**	**N/A**
name	**Display name**	**Yes**	**N/A**

Field name	Description	Required or not	Remarks
default	Default display value	No	If this field contains no data, #000 is displayed.

- type:multicolor

```
{
"name" : "gradient color",
"type" : "multicolor",
"default" : {
"style": "single",
"value": "#ccc",
"from": "#000",
"to": "#fff",
"angle": 0
}
}
```

Field name	Description	Required or not	Remarks
type	Type	Yes	N/A
name	Display name	Yes	N/A
default	Default display value	Yes	The value of style is single or double, and the value of value is single. The values of the from, to, and angle fields indicate the gradient range when the style field is set to double.

If style is set to single, the following actual data value is displayed:

```
{
"style": "single",
"value": "#xxx"
}
```

If style is set to double, the following actual data value is displayed:

```
{
"style": "double",
"from": "#xxx",
"to": "#xxx",
"angle": 90
```

```
}
```

- **type:image**

```
{
"name" : "background image",
"type" : "image",
"default": "http://datav.oss-cn-hangzhou.aliyuncs.com/uploads/
images/c4ba3c6518c1997f4baa612a600c3fbe.png"
}
```

Field name	Description	Required or not	Remarks
type	**Type**	**Yes**	**N/A**
name	**Display name**	**Yes**	**N/A**
default	**Default display value**	**No**	**If this field contains no data , no image is displayed.**

Note:

If the image is a local static file, place the file into the resources directory of the widget, for example, *./resources/xxxx.png*.

- **type:array**

```
{
"name" : "series",
"type" : "array",
"default": [{"name": "series 1", value: "arrival"},{"name": "
series 2", value: "departure"}]
"child": {
"name" : "series<%=i+1%>",
"type" : "object",
"child": {
  "name": {
    "name" : "series value",
    "type" : "text",
    "default": "series"
  },
  "value": {
    "name" : "series name",
    "type" : "text",
    "default": ""
  }
}
}
```

```
}
```

Field name	Description	Required or not	Remarks
type	Type	Yes	N/A
name	Display name	Yes	N/A
default	Default value	Yes	Object array. The property of the object must be the same as the field name specified in the child field.
child	Object name	Yes	The type field must be set to an object.
child	Object property	Yes	The child field is used to define the object property and the default value.

Note:

The default value of a series must be an object array. The property of the object in the array must be defined in child, and the display name, type, and default value of the object property must be set.

- **type:hidden**

```
{
"type" : "hidden",
"name": "value",
"default": 22
}
```

Field name	Description	Required or not	Remarks
type	Type	Yes	The Editor does not set this field.
name	Display name	Yes	N/A
default	Default value	No	The value is not displayed by default, but is transferred to a function.

- **type:boolean**

```
{
"type" : "boolean",
"name" : "display",
"default": true
```

```
}
```

Field name	Description	Required or not	Remarks
type	Type	Yes	N/A
name	Display name	Yes	N/A
default	Default value	No	If this field contains no data, false is displayed.

- type:radio

```
{
  "name":"font weight",
  "type":"radio",
  "default":1,
  "list":[
    {
      "name":"normal",
      "value":1
    },
    {
      "name":"bold",
      "value":2
    }
  ]
 }
```

Field name	Description	Required or not	Remarks
type	Type	Yes	N/A
name	Display name	Yes	N/A
list	Drop-down list	Yes	An object array that contains the name and value fields
default	Default value	No	If this field is not set, the default value is empty.

- type:percent

```
"percent": {
"type": "percent",
"name": "numerical value",
"default": 20
```

```
}
```

Field name	Description	Required or not	Remarks
type	**Type**	**Yes**	**N/A**
name	**Display name**	**Yes**	**N/A**
default	**Default value**	**No**	**If this field contains no data, the default value is empty.**

- **type: imageSelect**

```
"imageSelect":
{
"name": "decorative element",
"type": "imageSelect",
"default": "gif1",
"range": [
{
"name": "gif1",
"value": "gif1",
"url": "https://img.alicdn.com/tps/TB1tFMtPXXXXXXyXpXXXXXXXXXX-
1920-1080.gif"
},
{
"name": "gif2",
"value": "gif2",
"url": "https://img.alicdn.com/tps/TB1Pg3pPXXXXXcxXpXXXXXXXXXX-
1920-1080.gif"
}
]
}
```

Field name	Description	Required or not	Remarks
type	**Type**	**Yes**	**N/A**
name	**Display name**	**Yes**	**If this field contains no data, a key (for example, `margin`) is used as the display name.**
default	**Default value**	**No**	**If this field contains no data , options[0] is displayed.**

Field name	Description	Required or not	Remarks
range	**Options in a drop-down list**	**Yes**	**This field is an array, for example,** [{name :'',value: '', url: ''}].

Custom configuration item display and hiding

This section describes how to use show **to display or hide a custom configuration item.**

```
{
"xAxis": {
    "type": "group",
    "name": "x-axis",
    "children": {
        "show": { //Displays a check box on the right side of the x-
axis. This check box does not determine whether a similar property is
displayed.
             "type": "boolean",
             "name": "display",
             "default": true
        },
        "color": {
            "type": "color",
            "name": "color",
            "default": "#ccc",
            "show": [            //Determines whether to display a
property.
                ["show", "$eq", true]     //Condition under which the
color is displayed: The color is displayed only when the show filed of
 a similar property is set to true.
            ]
        }
    }
}
}
```

apis and api_data fields

- apis **field**

  This field defines the name of a widget API and the fields required by the API. Multiple APIs and fields are supported. For more information, see the notes in the preceding example.

  ```
  "apis": {                                  //Widget API. Multiple APIs
  are supported.
  "source": {                                //API name
  "handler": "render",                        //Processes the widget
  function name returned from an API.
  "description" : "doughnut chart API",          //API description
  ```

```
"fields" : {                                  //Fields required by an API
. Multiple fields are supported.
 "x" : {                                      //Field name
   "description" : "category",                      //Field description
   "type" : "string",                          //Field type
   "optional": true                            //Optional field
 },
 "y" : {
   "description" : "value",
   "type" : "int"
 }
}
},
"source2":{
      ...                                          //API 2
}
}
```

- `api_data` **field**

  This field defines the API data. Multiple APIs are supported. The API name must be the same as that in `apis`**. The recommended data size is within 6 KB. For more information, see the notes in the preceding example.**

```
"api_data":{                             //API data. Multiple APIs are
supported.
  "source" : [                          //API name, which must the same
 as that in the apis field. The recommended data size is within 6 KB
.
    {"x": "cargo", "y" : 5},
    {"x": "cargo", "y" : 22},
    {"x": "light goods", "y" : 22},
    {"x": "device", "y" : 14},
    {"x": "mineral", "y" : 15},
    {"x": "metal", "y" : 15},
    {"x": "building material", "y" : 12},
    {"x": "food", "y" : 12},
    {"x": "grain", "y" : 28}
  ],
},
```

events specifications

The widget interactions in DataV are implemented through the emit and events description. The emit is triggered according the events description (event description in package.json).

- `events` **description**

  The events field, similar to `config`**, is contained in the datav field. The events field defines the interaction event name, event description, and the variable name. For more information, see the notes in the preceding example.**

```
package.json
{
"events": {
```

```
    "click-me": {                 //custom event name
        "description": "click to trigger an event",             //
Event description field
        "fields": {            //Multiple variables and correspond
ing description can be defined.
            "value": {         //Field name, which can be customized
 as needed
                "description": "callback value"
            }
        }
    }
}
}
```

- **emit triggering**

  The emit is a basic function that can throw the event name and required data of a widget as parameters. In this way, other widgets can obtain the parameter values according to the variable names. For more information, see the notes in the preceding example.

```
index.js
render: function () {
    this.container.on('click', () => {
        this.emit('click-me', data)            // The value of data
 must be an object and cannot be a value. The property name is the
variable name.
    })
}
```

type field

The `type` field defines the widget type. For more information, see the notes in the preceding example.

```
type: ["regular_bar", ...]  //A widget can belong to different groups
. Multiple widget types are separated using underscores (_), for
example, "type1_type2".
    { // Common widget type
      regular: "basic chart",
      map: "map",
      figure: 'indicator',
      network: "networks",
      text: "text",
      decorate: "supporting graphics"
    }
```

3 Guide to ECharts widget encapsulation

This topic explains how to encapsulate ECharts widgets using the example *Bar Animation Delay* on the ECharts website. ECharts like this echarts.datatool type used as an example here are not directly compatible with DataV. Therefore, to encapsulate this chart type into a widget, you need to first extract the configurations and data of this EChart and then convert them into a supported format.

Extract configurations and data

According to the configurations in the left-side code bar for this example, you can extract configurations and data from the `option` variable. The complete code is shown as follows.

```
var xAxisData = [];
var data1 = [];
var data2 = [];
for (var i = 0; i < 100; i++) {
    xAxisData.push('Catagory' + i);
    Data1.push (math. sin (I/5) * (I/5-10) + I/6) * 5 );
    data2.push((Math.cos(i / 5) * (i / 5 -10) + i / 6) * 5);
}
option = {
    title: {
        text: 'Bar Animation Delay'
    },
    legend: {
        data: ['bar', 'bar2'],
        align: 'left'
    },
    toolbox: {
        // y: 'bottom',
        feature: {
            magicType: {
                type: ['stack', 'tiled']
            },
            dataView: {},
            saveAsImage: {
                pixelRatio: 2
            }
        }
    },
    tooltip: {},
    xAxis: {
        data: xAxisData,
        silent: false,
        splitLine: {
            show: false
        }
    },
    yAxis: {
    },
    series: [{
        name: 'bar',
```

```
        type: 'bar',
        data: data1,
        animationDelay: function (idx) {
            return idx * 10;
        }
    }, {
        name: 'bar2',
        type: 'bar',
        data: data2,
        animationDelay: function (idx) {
            return idx * 10 + 100;
        }
    }],
    animationEasing: 'elasticOut',
    animationDelayUpdate: function (idx) {
        return idx * 5;
    }
};
```

In the preceding code, you can find the following data item configurations:

- **option.xAxis.data**
- **option.series[x].data**

To convert these configurations into data elements that are recognized by DataV, you must perform the following operations.

```
[
  {
    x: 'An item value of option, xAxis, and data',
    y: 'An item value of option, series[x], and data',
    s: 'An item value of option, series[x], and name'
  }
]
```

Except for the preceding three data items, the rest are configurations.

Write package.json

Write the preceding configurations and data into `package.json` **based on** *package.json specifications* **. You need to note that:**

- **You can delete configurations that are not required by your widgets.**
- **If the configuration that you need is not in the example code, you can obtain it from** *ECharts Chart Configuration*.

- **Several ECharts configuration items are supported by DataV, except for the following items:**
  - **Configuration items that are functions.**
  - **Configuration items of some ECharts chart type, such as, echarts.datatool.xxx.**
  - **Configuration items that contain more than one item type, such as both text and number types.**
- **The configuration schema must match that of ECharts. If they cannot be matched due to incompatibility with DataV, you need to convert them in `index.js` so that they can match. Through this process, even ECharts like echarts.datatool, which are not directly supported by DataV, can be made compatible with DataV once converted.**

The following is part of the ECharts-chart converted package.json from the preceding example.

```
{
  "datav": {
    "cn_name": "Bar chart",
    "icon": "",
    "protocol": 2,
    "type": [
      "regular_bar"
    ],
    "View ":{
      "width": "600",
      "height": "200",
      "minWidth": "40",
      "minHeight": "20"
    },
    "apis": {
      "source": {
        "handler": "render",
        "description": "Interface description of echarts animation
delay chart",
        "fields": {
          "x": {
            "description": "Value of the x axis"
          },
          "y": {
            "description": "Value of the y axis"
          },
          "s": {
            "description": "Value of the y axis",
            "optional": true
          }
        }
      }
    },
    "config": {
      "legend": {
        //...
      },
      "grid": {
```

```
        //...
      },
      "xAxis": {
        "name": "x axis",
        "type": "group",
        "children": {
          "show": {
            "name": "Display",
            "type": "boolean",
            "default": false
          },
          "offset": {
            "name": "Offset",
            "type": "number",
            "default": 0
          },
          "type": {
            "name": "type",
            "type": "text",
            "default": "category"
          },
          "name": {
            "name": "name",
            "type": "text",
            "default": ""
          },
          "nameLocation": {
            "name": "Name location",
            "type": "text",
            "default": "end"
          },
          "nameTextStyle": {
            "name": "Name stype",
            "type": "group",
            "children": {
              "color": {
                "name": "Color",
                "type": "color",
                "default": "rgba(0,0,0,0)"
              },
              "fontStyle": {
                "name": "Font style",
                "type": "text",
                "default": "normal"
              },
              "fontWeight": {
                "name": "Font weight",
                "type": "text",
                "default": "normal"
              },
              "fontFamily": {
                "name": "Font family",
                "type": "text",
                "default": "sans-serif"
              },
              "fontSize": {
                "name": "Font size",
                "type": "number",
                "default": 10
              }
            },
            "fold": true
          },
          "nameGap": {
```

```
            "name": "Gap between names",
            "type": "number",
            "default": 15
          },
          "nameRotate": {
            "name": "Name rotation",
            "type": "number",
            "default": null
          },
          "inverse": {
            "name": "Inverse",
            "type": "boolean",
            "default": false
          },
          "boundaryGap": {
            "Name": "Boundary gap ",
            "type": "boolean",
            "default": true
          },
          "min": {
            "name": "Min",
            "type": "text",
            "default": "dataMin"
          },
          "max": {
            "name": "Max",
            "type": "text",
            "default": "dataMax"
          },
          "scale": {
            "name": "Auto scaling",
            "type": "boolean",
            "default": false
          },
          "splitNumber": {
            "name": "Split number",
            "type": "number",
            "default": 5
          },
          "minInterval": {
            "Name": "Min interval",
            "type": "number",
            "default": 0
          },
          "logBase": {
            "name": "Log base",
            "type": "number",
            "default": 10
          },
          "silent": {
            "name": "Silent",
            "type": "boolean",
            "default": false
          },
          "triggerEvent": {
            "name": "Trigger event",
            "type": "boolean",
            "default": false
          },
          "axisLine": {
            "name": "Axis line",
            "type": "group",
            "children": {
              "show": {
```

```
            "name": "Display",
            "type": "boolean",
            "default": false
          },
          "onZero": {
            "name": "On zero",
            "type": "boolean",
            "default": true
          },
          "lineStyle": {
            "name": "Line style",
            "type": "group",
            "children": {
              "color": {
                "name": "Color",
                "type": "multicolor",
                "default": {
                  "style": "single",
                  "value": "rgba(255,255,255,. 8)"
                }
              },
              "width": {
                "name": "Width",
                "type": "number",
                "default": 1
              },
              "type": {
                "name": "Type",
                "type": "text",
                "default": "solid"
              },
              "opacity": {
                "name": "Opacity",
                "type": "number",
                "default": 1
              }
            },
            "fold": true
          }
        },
        "fold": true
      },
      "axisTick": {
        "name": "Axis tick",
        "type": "group",
        "children": {
          "show": {
            "name": "Display",
            "type": "boolean",
            "default": false
          },
          "alignWithLabel": {
            "name": "Align with label",
            "type": "boolean",
            "default": false
          },
          "interval": {
            "name": "Interval",
            "type": "number",
            "default": 0
          },
          "inside": {
            "name": "Inside",
            "type": "boolean",
```

```
            "default": false
          },
          "length": {
            "name": "Length",
            "type": "number",
            "default": 5
          },
          "lineStyle": {
            "name": "Line style",
            "type": "group",
            "children": {
              "color": {
                "name": "Color",
                "type": "multicolor",
                "default": {
                  "style": "single",
                  "value": "rgba(255,255,255,. 8)"
                }
              },
              "width": {
                "name": "Width",
                "type": "number",
                "default": 1
              },
              "type": {
                "name": "Type",
                "type": "text",
                "default": "solid"
              },
              "opacity": {
                "name": "Opacity",
                "type": "number",
                "default": 1
              }
            },
            "fold": true
          }
        },
        "fold": true
      },
      "axisLabel": {
        "name": "Axis label",
        "type": "group",
        "children": {
          "show": {
            "name": "Show",
            "type": "boolean",
            "default": true
          },
          "interval": {
            "name": "Interval",
            "type": "number",
            "default": 13
          },
          "inside": {
            "name": "Inside",
            "type": "boolean",
            "default": false
          },
          "rotate": {
            "name": "Rotation",
            "type": "number",
            "default": 0
          },
```

```
                "margin": {
                  "name": "Margin",
                  "type": "number",
                  "default": 8
                },
                "showMinLabel": {
                  "name": "Show minimum label",
                  "type": "boolean",
                  "default": true
                },
                "showMaxLabel": {
                  "name": "Show maximum label",
                  "type": "boolean",
                  "default": true
                },
                "textStyle": {
                  "name": "Text style",
                  "type": "group",
                  "children": {
                    "color": {
                      "name": "Color",
                      "type": "color",
                      "default": "rgba(255,255,255,. 8)"
                    },
                    "fontStyle": {
                      "name": "Font style",
                      "type": "text",
                      "default": "normal"
                    },
                    "fontWeight": {
                      "name": "Font weight",
                      "type": "text",
                      "default": "normal"
                    },
                    "fontFamily": {
                      "name": "Font family",
                      "type": "text",
                      "default": "sans-serif"
                    },
                    "fontSize": {
                      "name": "Font size",
                      "type": "number",
                      "default": 10
                    },
                    "align": {
                      "name": "Alignment",
                      "type": "select",
                      "range": [
                        {
                          "name": "Auto",
                          "value": "auto"
                        },
                        {
                          "name": "Left",
                          "value": "left"
                        },
                        {
                          "name": "Center",
                          "Value": "center"
                        },
                        {
                          "name": "Right",
                          "value": "right"
                        }
```

```
                ],
                "default": ""
              },
              "baseline": {
                "name": "Baseline",
                "type": "text",
                "default": ""
              }
            },
            "fold": true
          }
        },
        "fold": true
      },
      "splitLine": {
        "name": "split line",
        "type": "group",
        "children": {
          "show": {
            "name": "show",
            "type": "boolean",
            "default": false
          },
          "interval": {
            "name": "interval",
            "type": "number",
            "Default": 0
          },
          "lineStyle": {
            "name": "Line style",
            "type": "group",
            "children": {
              "width": {
                "name": "Width",
                "type": "number",
                "default": 1
              },
              "type": {
                "name": "Type",
                "type": "text",
                "default": "solid"
              },
              "opacity": {
                "name": "Opacity",
                "type": "number",
                "default": 1
              }
            },
            "fold": true
          }
        },
        "fold": true
      },
      "splitArea": {
        "name": "split area",
        "type": "group",
        "children": {
          "interval": {
            "name": "interval",
            "type": "number",
            "default": 0
          },
          "show": {
            "name": "show",
```

```
            "type": "boolean",
            "default": false
          },
          "areaStyle": {
            "name": "Area style",
            "type": "group",
            "children": {
              "opacity": {
                "name": "Opacity",
                "type": "number",
                "default": 1
              }
            },
            "fold": true
          }
        },
        "fold": true
      },
      "axisPointer": {
        "name": "Axis pointer",
        "type": "group",
        "children": {
          "show": {
            "name": "Show",
            "type": "boolean",
            "default": true
          },
          "type": {
            "name": "Type",
            "type": "select",
            "default": "line",
            "range": [
              {
                "name": "Line pointer",
                "value": "line"
              },
              {
                "name": "Shadow pointer",
                "value": "shadow"
              }
            ]
          },
          "snap": {
            "name": "Snap",
            "type": "boolean",
            "default": false
          },
          "value": {
            "name": "Initial value",
            "type": "number",
            "default": null
          },
          "status": {
            "name": "Status",
            "type": "boolean",
            "default": false
          },
          "label": {
            "name": "Label",
            "type": "group",
            "children": {
              "show": {
                "name": "Show",
                "type": "boolean",
```

```
            "default": false
          },
          "precision": {
            "name": "precision",
            "type": "number",
            "default": "'auto'"
          },
          "margin": {
            "name": "Margin",
            "type": "boolean",
            "default": 3
          },
          "textStyle": {
            "name": "Text style",
            "Type": "group ",
            "children": {
              "color": {
                "name": "Color",
                "type": "color",
                "default": "#ffffff"
              },
              "fontStyle": {
                "name": "Font style",
                "type": "text",
                "default": "normal"
              },
              "fontWeight": {
                "name": "Font weight",
                "type": "text",
                "default": "normal"
              },
              "fontFamily": {
                "name": "Font family",
                "type": "text",
                "default": "sans-serif"
              },
              "fontSize": {
                "name": "Font size",
                "type": "number",
                "default": 10
              }
            },
            "fold": true
          },
          "backgroundColor": {
            "name": "Background color",
            "type": "text",
            "default": "auto"
          },
          "borderColor": {
            "name": "Border color",
            "type": "text",
            "default": ""
          },
          "borderWidth": {
            "name": "Border width",
            "type": "text",
            "default": ""
          }
        },
        "fold": true
      },
      "lineStyle": {
        "name": "Line style",
```

```
      "type": "group",
      "show": [
        [
          "type",
          "$eq",
          "line"
        ]
      ],
      "children": {
        "color": {
          "name": "Color",
          "type": "multicolor",
          "default": {
            "style": "single",
            "value": "rgba(0,0,0,0)"
          }
        },
        "width": {
          "name": "Width",
          "type": "number",
          "default": 1
        },
        "type": {
          "name": "Type",
          "type": "text",
          "default": "solid"
        },
        "opacity": {
          "name": "Opacity",
          "type": "number",
          "default": 1
        }
      },
      "fold": true
    },
    "shadowStyle": {
      "name": "Shadow style",
      "type": "group",
      "show": [
        [
          "type",
          "$eq",
          "shadow"
        ]
      ],
      "children": {
        "color": {
          "name": "Color",
          "type": "multicolor",
          "default": {
            "style": "single",
            "value": "rgba(150,150,150,0.3)"
          }
        },
        "opacity": {
          "name": "Opacity",
          "type": "number",
          "default": 1
        }
      },
      "fold": true
    },
    "handle": {
      "name": "Handle",
```

```
                "type": "group",
                "children": {
                  "show": {
                    "name": "Show",
                    "type": "boolean",
                    "default": false
                  },
                  "size": {
                    "name": "Size",
                    "type": "number",
                    "default": 45
                  },
                  "margin": {
                    "name": "Margin",
                    "type": "number",
                    "default": 50
                  },
                  "color": {
                    "name": "Color",
                    "type": "text",
                    "default": "#333"
                  },
                  "throttle": {
                    "name": "Throttle",
                    "type": "number",
                    "default": 40
                  }
                },
                "fold": true
              }
            },
            "fold": true
          }
        },
        "fold": true
      },
      "yAxis": {
        //...
      },
      "tooltip": {
        //...
      },
      "series": {
        "name": "Series",
        "type": "array",
        "fold": true,
        "default": [
          {
            "name": "bar",
            "type": "bar",
            "legendHoverLink": true,
            "coordinateSystem": "cartesian2d",
            "label": {
              "normal": {
                "show": false,
                "textStyle": {
                  "color": "#000",
                  "fontStyle": "normal",
                  "fontWeight": "normal",
                  "fontFamily": "sans-serif",
                  "fontSize": 10
                }
              },
              "emphasis": {
```

```
            "show": false,
            "textStyle": {
              "color": "#000",
              "fontStyle": "normal",
              "fontWeight": "normal",
              "fontFamily": "sans-serif",
              "fontSize": 10
            }
          }
        },
        "itemStyle": {
          "normal": {
            "color": {
              "style": "single",
              "value": "#00c2ff"
            },
            "borderColor": {
              "style": "single",
              "value": "#000"
            },
            "borderWidth": 0,
            "borderType": "solid",
            "barBorderRadius": 0,
            "opacity": 1
          },
          "emphasis": {
            "color": {
              "style": "single",
              "value": "#00c2ff"
            },
            "borderColor": {
              "style": "single",
              "value": "#000"
            },
            "borderWidth": 0,
            "borderType": "solid",
            "opacity": 1
          }
        },
        "stack": "",
        "barWidth": "50%",
        "barMinHeight": 0,
        "barGap": "30%",
        "barCategoryGap": "20%",
        "silent": false
      },
      {
        "name": "bar2",
        "type": "bar",
        "legendHoverLink": true,
        "coordinateSystem": "cartesian2d",
        "label": {
          "normal": {
            "show": false,
            "textStyle": {
              "color": "#000",
              "fontStyle": "normal",
              "fontWeight": "normal",
              "fontFamily": "sans-serif",
              "fontSize": 10
            }
          },
          "emphasis": {
            "show": false,
```

```
            "textStyle": {
              "color": "#000",
              "fontStyle": "normal",
              "fontWeight": "normal",
              "fontFamily": "sans-serif",
              "fontSize": 10
            }
          }
        },
        "itemStyle": {
          "normal": {
            "color": {
              "style": "single",
              "value": "#5bffb0"
            },
            "borderColor": {
              "style": "single",
              "value": "#000"
            },
            "borderWidth": 0,
            "borderType": "solid",
            "barBorderRadius": 0,
            "opacity": 1
          },
          "emphasis": {
            "color": {
              "style": "single",
              "value": "#5bffb0"
            },
            "borderColor": {
              "style": "single",
              "value": "#000"
            },
            "borderWidth": 0,
            "borderType": "solid",
            "opacity": 1
          }
        },
        "stack": "",
        "barWidth": "50%",
        "barMinHeight": 0,
        "barGap": "30%",
        "barCategoryGap": "20%",
        "silent": false
      }
    ],
    "child": {
      "type": "object",
      "name": "Series <%= i+1 %>",
      "child": {
        "name": {
          "name": "Name",
          "type": "text",
          "default": ""
        },
        "legendHoverLink": {
          "name": "Legend hover link highlight",
          "type": "boolean",
          "default": true
        },
        "coordinateSystem": {
          "name": Coordinate system",
          "type": "text",
          "default": "cartesian2d"
```

```
        },
        "label": {
          "name": "Label",
          "type": "group",
          "children": {
            "normal": {
              "name": "Normal",
              "type": "group",
              "children": {
                "show": {
                  "name": "Show",
                  "type": "boolean",
                  "default": false
                },
                "textStyle": {
                  "name": "Text style",
                  "type": "group",
                  "children": {
                    "color": {
                      "name": "Color",
                      "type": "color",
                      "default": "#000"
                    },
                    "fontStyle": {
                      "name": "Font style",
                      "type": "text",
                      "default": "normal"
                    },
                    "fontWeight": {
                      "name": "Font weight",
                      "type": "text",
                      "default": "normal"
                    },
                    "fontFamily": {
                      "name": "Font family",
                      "type": "text",
                      "default": "sans-serif"
                    },
                    "fontSize": {
                      "name": "Font size",
                      "type": "number",
                      "default": 10
                    }
                  },
                  "fold": true
                }
              },
              "fold": true
            },
            "emphasis": {
              "name": "Emphasis",
              "type": "group",
              "children": {
                "show": {
                  "name": "Show",
                  "type": "boolean",
                  "default": false
                },
                "textStyle": {
                  "name": "Text style",
                  "type": "group",
                  "children": {
                    "color": {
                      "name": "Color",
```

```
                    "type": "color",
                    "default": "#000"
                  },
                  "fontStyle": {
                    "name": "Font style",
                    "type": "text",
                    "default": "normal"
                  },
                  "fontWeight": {
                    "name": "Font weight",
                    "type": "text",
                    "default": "normal"
                  },
                  "fontFamily": {
                    "name": "Font family",
                    "type": "text",
                    "default": "sans-serif"
                  },
                  "fontSize": {
                    "name": "Font size",
                    "type": "number",
                    "default": 10
                  }
                },
                "fold": true
              }
            },
            "fold": true
          }
        },
        "fold": true
      },
      "itemStyle": {
        "name": "Item style",
        "type": "group",
        "children": {
          "normal": {
            "name": "Normal",
            "type": "group",
            "children": {
              "color": {
                "name": "Color",
                "type": "multicolor",
                "default": {
                  "style": "single",
                  "value": "rgba(0,0,0,0)"
                }
              },
              "borderColor": {
                "name": "Border color",
                "type": "multicolor",
                "default": {
                  "style": "single",
                  "value": "#000"
                }
              },
              "borderWidth": {
                "name": "Border width",
                "type": "number",
                "default": 0
              },
              "borderType": {
                "name": "Border type",
                "type": "text",
```

```
                    "default": "solid"
                  },
                  "barBorderRadius": {
                    "name": "Bar border radius",
                    "type": "number",
                    "default": 0
                  },
                  "opacity": {
                    "name": "Opacity",
                    "type": "number",
                    "default": 1
                  }
                },
                "fold": true
              },
              "emphasis": {
                "name": "Emphasis",
                "type": "group",
                "children": {
                  "color": {
                    "name": "Color",
                    "type": "multicolor",
                    "default": {
                      "style": "single",
                      "value": "rgba(0,0,0,0)"
                    }
                  },
                  "borderColor": {
                    "name": "Border color",
                    "type": "multicolor",
                    "default": {
                      "style": "single",
                      "value": "#000"
                    }
                  },
                  "borderWidth": {
                    "name": "Border width",
                    "type": "number",
                    "default": 0
                  },
                  "borderType": {
                    "name": "Border type",
                    "type": "text",
                    "default": "solid"
                  },
                  "opacity": {
                    "name": "Opacity",
                    "type": "number",
                    "default": 1
                  }
                },
                "fold": true
              }
            },
            "fold": true
          },
          "stack": {
            "name": "Stack",
            "type": "text",
            "default": ""
          },
          "barWidth": {
            "name": "Bar width",
            "type": "text",
```

```
            "default": "50%"
          },
          "barMinHeight": {
            "name": "Minimum bar height",
            "type": "number",
            "default": 0
          },
          "barGap": {
            "name": "Bar gap",
            "type": "text",
            "default": "30%"
          },
          "barCategoryGap": {
            "name": "Bar Category Gap",
            "type": "text",
            "default": "20%"
          },
          "silent": {
            "name": "Silent",
            "type": "boolean",
            "Default": false
          }
        }
      }
    },
    "animation": {
      "name": "Animation",
      "type": "boolean",
      "default": true
    },
    "animationThreshold": {
      "name": "Animation threshold",
      "type": "number",
      "default": 2000
    },
    "animationDuration": {
      "name": "Animation duration",
      "type": "number",
      "default": 1000
    },
    "animationEasing": {
      "name": "Animation easing",
      "type": "text",
      "default": "elasticOut"
    }
  },
  "api_data": {
    "source": [
      {
        "x": "Source 0",
        "y": 0,
        "s": "bar"
      },
      {
        "x": "Source 0",
        "y": -50,
        "s": "bar2"
      },
      {
        "x": "Source 1",
        "y": -8.901463875624668,
        "s": "bar"
      },
      {
```

```
            "x": "Source 1",
            "y": -47.18992898088751,
            "s": "bar2"
          },
          {
            "x": "Source 2",
            "y": -17.025413764148556,
            "s": "bar"
          },
          {
            "x": "Source 2",
            "y": -42.54426104547181,
            "s": "bar2"
          },
          {
            "x": "Source 3",
            "y": -24.038196249566663,
            "s": "bar"
          },
          {
            "x": "Source 3",
            "y": -36.290773900754886,
            "s": "bar2"
          },
          {
            "x": "Source 4",
            "y": -29.66504684804471,
            "s": "bar"
          },
          {
            "x": "Source 4",
            "Y":-28.71517529663627,
            "s": "bar2"
          },
          {
            "x": "Source 5",
            "y": -33.699527649688676,
            "s": "bar"
          },
          {
            "x": "Source 5",
            "y": -20.146937097399626,
            "s": "bar2"
          },
          {
            "x": "Source 6",
            "y": -36.00971978255796,
            "s": "bar"
          },
          {
            "x": "Source 6",
            "y": -10.94374119697364,
            "s": "bar2"
          },
          {
            "x": "Source 7",
            "y": -36.541005056170455,
            "s": "bar"
          },
          {
            "x": "Source 7",
            "y": -1.4752538113770308,
            "s": "bar2"
          },
```

```
        {
          "x": "Source 8",
          "y": -35.31542466107655,
          "s": "bar"
        },
        {
          "x": "Source 8",
          "y": 7.893046603320797,
          "s": "bar2"
        },
        {
          "x": "Source 9",
          "y": -32.427752866005996,
          "s": "bar"
        },
        {
          "x": "Source 9",
          "y": 16.81528588241657,
          "s": "bar2"
        },
        {
          "x": "Source 10",
          "y": -28.038563739693934,
          "s": "bar"
        },
        {
          "x": "Source 10",
          "y": 24.979206795219028,
          "s": "bar2"
        },
        {
          "x": "Source 11",
          "y": -22.364693082297347,
          "s": "bar"
        },
        {
          "x": "Source 11",
          "y": 32.11821023962515,
          "s": "bar2"
        },
        {
          "x": "Source 12",
          "y": -15.667600860943732,
          "s": "bar"
        },
        {
          "x": "Source 12",
          "y": 38.02096119056733,
          "s": "bar2"
        },
        {
          "x": "Source 13",
          "y": -8.240217424060843,
          "s": "bar"
        },
        {
          "x": "Source 13",
          "y": 42.53821720798438,
          "s": "bar2"
        },
        {
          "x": "Source 14",
          "y": -0.3929067389459173,
          "s": "bar"
```

```
        },
        {
          "x": "Source 14",
          "y": 45.58667093073836,
          "s": "bar2"
        },
        {
          "x": "Source 15",
          "y": 7.560799717904647,
          "s": "bar"
        },
        {
          "x": "Source 15",
          "y": 47.14973738101559,
          "s": "bar2"
        },
        {
          "x": "Source 16",
          "y": 15.318054209871054,
          "s": "bar"
        },
        {
          "x": "Source 16",
          "y": 47.275355710354944,
          "s": "bar2"
        },
        ...
      ]
    }
  }
}
```

Write index.js

To write the *index.js* file, complete the following steps. For more information, see *index.js specifications*.

1. **In the initaization function, run *EChart.init*.**
2. **In the rendering function, run *chart.setOption*.**
3. **In the scaling function, run *chart.resize*.**
4. **In the clearing function, run *chart.clear*.**
5. **In the destruction function, run *chart.dispose*.**

4 FAQ

How and where can I find widget templates?

In the command line, install datav-cli and use the `datav init` command to download and install widget templates. For more information, see *Install DataV development tools*.

Why did my widgets fail to publish?

Widgets may fail to publish due to one of the following reasons:

- **Widgets are not authorized to be published.**

  Widgets are not authroized to be published because they are not preceded by the widget package name (the `name` field in *`package.json`*). The correct widget name format is @namespace/xxx (with @namespace being the widget package name).
- **Widgets are still under review or their review failed.**

  If your widget has not passed review after much time, contact Alibaba Cloud technical support.

How do I write a data event so that, when I trigger the event, other widgets listen?

To write a data event so that, when the event is triggered, other widgets can listen, the procedure is as follows:

1. **In the *`package.json`* file, under the datav field, define the event name events in the correct format.**
2. **In the *`index.js`* file, use the `this.emit(event_name,value)` method to send the event name and parameters.**

   Note:
   The `value` parameter must be an object, rather than a basic type value.
3. **In the Data pane, other widgets must use the format of a colon plus a parameter name (: xxx) to call the corresponding data of this parameter.**

How can I write an ECharts widget?

For information about how to write an ECharts widget, see *ECharts widget encapsulation guide*.

Why did I my local picture fail to upload?

If you need to use a local static file as a resource path, you need to create a resources file in the root directory where your widget is located, put resources, such as pictures into the directory, and call the path, such as, `./resources/xxx.png` .

Why am I unable to find the packages of my program?

If you continue to receive an error that indicates that packages were not found after you run the `npm install` command, remove the datav-cli tool and reinstall it.

Why have my widgets not passed review yet?

If your widgets have not passed review after much time, contact Alibaba Cloud technical support.

Why do I suddenly not have permission to publish widgets?

If you are using the DataV Developer Edition and have not uploaded any widgets or tutorials for two months, your permission will be remoked. You can re-apply for the permission or contact Alibaba Cloud technical support.

Why did my widgets work properly locally, but an error occured when they were used in DataV?

This may have been caused by one of the following reasons:

- **Setting data to null is not supported by the widget.**
- **jQuery was modified in the widget code. More specifically, jquery have been reassigned.**
- **The window object was modifed in the widget code.**
- **When several widgets were added, modified configurations did not take effect, check if your widget codes have a CSS file.**
- **The DOM model was modified outside of the container.**
- **When using jQuery to access the DOM model, the `$('.classname')` operation is not allowed. Rather, you must search for DOM nodes in the container.**

Why am I unable to find my widgets on the edit page?

To view your widgets, go to My Widgets on the top navigation bar.

Why are buyers unable to find my widgets after they bought them?

Buyers probably cannot find your widgets because your widget classification may not be set properly. For more information, see *type fields*.