

数据可视化

开发指南

文档版本

01

发布日期

2025-11-14



版权所有 © 华为云计算技术有限公司 2025。保留一切权利。

非经本公司书面许可，任何单位和个人不得擅自摘抄、复制本文档内容的部分或全部，并不得以任何形式传播。

商标声明



HUAWEI和其他华为商标均为华为技术有限公司的商标。

本文档提及的其他所有商标或注册商标，由各自的所有人拥有。

注意

您购买的产品、服务或特性等应受华为云计算技术有限公司商业合同和条款的约束，本文档中描述的全部或部分产品、服务或特性可能不在您的购买或使用范围之内。除非合同另有约定，华为云计算技术有限公司对本文档内容不做任何明示或暗示的声明或保证。

由于产品版本升级或其他原因，本文档内容会不定期进行更新。除非另有约定，本文档仅作为使用指导，本文档中的所有陈述、信息和建议不构成任何明示或暗示的担保。

华为云计算技术有限公司

地址：贵州省贵安新区黔中大道交兴功路华为云数据中心 邮编：550029

网址：<https://www.huaweicloud.com/>

目 录

- 1 快速开发自定义组件..... 1
- 2 组件开发指南..... 5
 - 2.1 组件开发包文件介绍..... 5
 - 2.2 Index.js 规范..... 5
 - 2.3 gui.json 规范..... 8
 - 2.4 UI 类型介绍..... 12
- A 修订记录..... 17

1 快速开发自定义组件

自定义组件开发流程

1. 环境准备
2. 安装开发者工具
3. 生成组件包
4. 组件开发
5. 预览组件
6. 发布组件

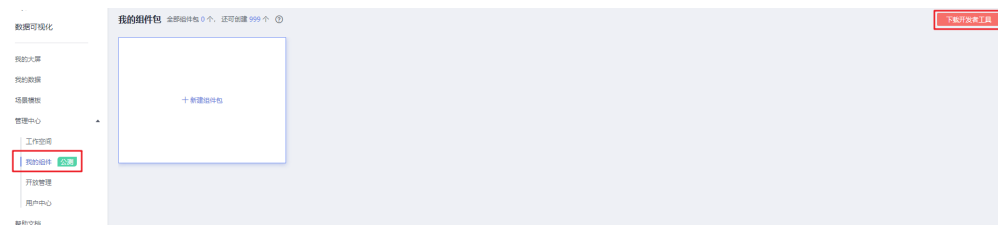
环境准备

进入[NodeJS官网](#)，下载并安装适合自己操作系统的NodeJS版本。NodeJS支持Windows、MacOS或Linux操作系统。

安装开发者工具

1. 登录DLV管理控制台，进入“管理中心 > 我的组件”页面，单击页面右上角的“下载开发者工具”按钮，下载包名如“dlv-cli-x.x.x”的开发者工具，其中包名中的x.x.x代表工具的版本号。

图 1-1 下载工具



2. 将开发者工具“dlv-cli-x.x.x”解压到本地。若在Windows操作系统中请打开命令提示符窗口，若在Linux或Mac操作系统中请打开命令行终端，执行`cd`命令进入dlv-cli-x.x.x目录，然后执行`npm link`命令安装dlv-cli开发者工具。
3. 工具安装后执行`dlv`命令显示如图1-2所示信息表示安装成功，其中当前开发工具信息如表1-1所示。

图 1-2 dlvs 命令



表 1-1 开发工具命令说明

命令	说明
dlvs init	快速初始化组件模板。
dlvs start	启动组件包预览组件。
dlvs package	组件打包。

生成组件包

组件包是DLV服务提供的自定义组件模板，用户可以基于当前模板进行开发。

首先新建一个目录，例如newCom，进入该目录执行**dlvs init**命令创建组件，如图1-3所示，请根据提示信息依次输入新建组件的相关信息。

说明

请不要在开发者工具dlvs-cli-x.x.x的目录下执行**dlvs init**命令，否则会导致开发者工具无法正常使用。

图 1-3 dlv init

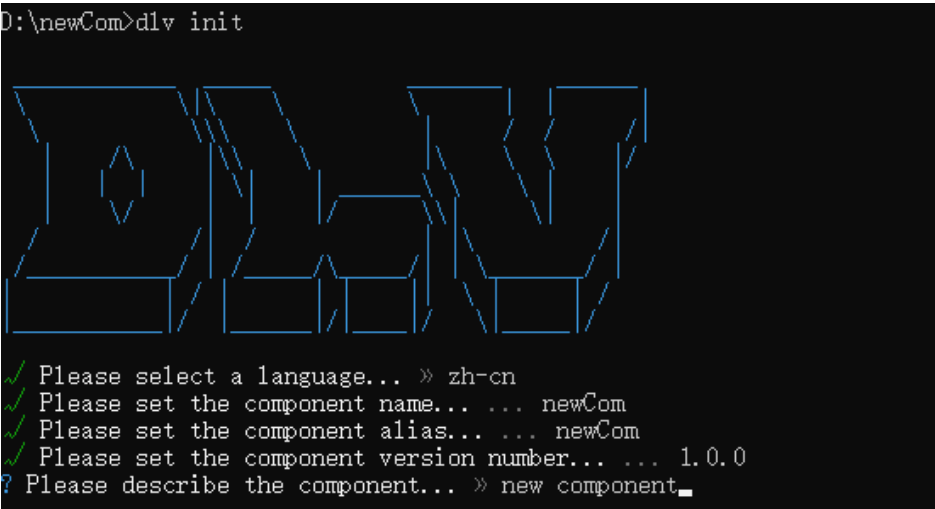


表 1-2 组件信息

组件信息	说明
Please select a language...	通过↑和↓选择语言环境。
Please set the component name...	组件名（包含字母、数字、下划线，且只能以字母和下划线开头，不超过32个字符。若无特殊说明，本文中的名称命名都遵循此规范）。
Please set the component alias...	产品界面中显示的组件名（包含中文、字母、特殊字符，32个字符。）。
Please set the component version number...	组件版本号，默认版本号为1.0.0。
Please describe the component...	组件的描述信息。

当新建目录下生成了相应模板文件后，说明您的组件包已经成功生成。

- └─node_modules # npm依赖包
- └─gui.json # 组件配置
- └─index.js # 组件入口
- └─index.less # 组件样式
- └─package.json # npm模块描述文件

组件开发

生成组件包后，用户可以基于生成的模板进行自定义组件开发，详细的开发指导请参见[组件开发指南](#)。

预览组件

首先进入组件目录中，通过`dlv start`命令预览组件。当命令行显示“服务启动”时，说明预览组件的服务已经启动，Chrome浏览器会被自动打开，并导航到组件预览页。

预览页主要分为中心画布区和右侧工具栏两部分，详细介绍如下：

- 中心画布区
中心画布区用来展现组件，实时观测组件变化的区域；
所有右侧工具栏的配置、数据修改都会实时展示在中心画布的组件上；
组件的黑框代表了组件的容器范围大小，每个方向上的黑框都可以缩放来测试组件在任意方向缩放的表现。
- 右侧工具栏
右侧工具栏分为样式、数据和交互3个面板。

表 1-3 面板介绍

面板	说明
样式	演示页面描述了组件可变动的一些配置项，如果在配置操作，改动会立即生效。
数据	数据页面描述了组件的数据接口配置，数据页的数据一旦改动，组件都会进行相应的改动。
交互	交互页面描述了组件的交互说明。

发布组件

进入组件的目录下，执行`dlv package`命令，在组件目录外会有一个以“组件-版本号”命名的tar.gz压缩包，将此压缩包上传到DLV管理控制台中“我的组件”页面的某一个组件包中，即可发布。

2 组件开发指南

2.1 组件开发包文件介绍

本文介绍开发者在开发DLV自定义组件时，需要遵循的文件结构。
在使用命令`dlv init`生成组件包之后，组件包中包含如下文件。

表 2-1 开发包文件介绍

文件名	说明
gui.json	组件配置，包括样式、数据和交互的配置。
Index.js	组件主入口。
Index.less	组件样式css配置。

2.2 Index.js 规范

Index.js文件是组件的主入口文件，该文件提供了一个示例供您参考，并介绍了index.js文件中常用的组件生命周期或相关函数。

表 2-2 组件方法

函数	说明
refresh()	默认渲染方法，当组件初始化和重绘时被调用。 (自定义实现)
resize()	缩放，当组件被拖拽、缩放时被调用。
loadData()	更新组件数据，当组件加载数据时调用。 (自定义实现)

函数	说明
dispatch(String: eventType, object: data)	可选方法，触发组件交互事件。如果该组件支持事件交互，需用户自定义实现该方法，并且此方法需要在gui.json中配置组件交互事件(events)。 - eventType：事件名称。 - data：事件触发时传递的数据。
getRelation()	获取组件数据配置映射字段。

Index.js文件示例如下：

```
import gui from './gui.json';
import './index.less';
//global: echarts jQuery

class App extends BaseChart {
  constructor(props = {}) {
    super(props);
    Object.assign(props);
    this.chart = echarts.init(this.el);
  }
  refresh() {
    this.loadData();
  }
  loadData() {
    let option = this.getOption();
    if (option) {
      this.chart.setOption(option, true);
    }
  }
  resize() {
    this.chart.resize();
  }
  getOption() {
    let styledata = this.config.styledata,
        relation = this.getRelation();
    let yData = [], data = [];
    this.data.forEach(item => {
      yData.push(item[relation.x]);
      data.push(item[relation.y]);
    });
    return {
      backgroundColor: '222',
      grid: {
        top: '20',
        left: '20',
        right: '20',
        bottom: '20',
        containLabel: true
      },
      yAxis: [{
        type: 'category',
        data: yData,
        inverse: true,
        axisTick: {
          show: false
        },
      },
      axisLabel: {
        margin: 10,
        textStyle: {
          fontFamily: styledata['font'],
          fontSize: 24,

```

```
        color: '#fff'
      }
    },
    axisLine: {
      show: false
    }
  ],
  xAxis: [{
    type: 'value',
    axisLabel: {
      show: false
    },
    axisLine: {
      show: false
    },
    splitLine: {
      show: false
    }
  ]],
  series: [{
    type: 'bar',
    barWidth: 14,
    data,
    label: {
      normal: {
        show: true,
        position: 'insideBottomRight',
        formatter: '{c}%',
        distance: 0,
        offset: [30, -20],
        color: '#fff',
        fontSize: 16,
        padding: [5, 15, 10, 15]
      }
    },
    itemStyle: {
      normal: {
        color: new echarts.graphic.LinearGradient(1, 0, 0, 0, [{
          offset: 0,
          color: '#57eabf'
        }, {
          offset: 1,
          color: '#2563f9'
        }], false),
        barBorderRadius: 14
      }
    },
    type: "bar",
    barWidth: 14,
    xAxisIndex: 0,
    barGap: "-100%",
    data: [120, 120],
    itemStyle: {
      normal: {
        color: "#444a58",
        barBorderRadius: 14
      }
    },
    zlevel: -1
  ]
}];
App.gui = gui;export default App;
```

2.3 gui.json 规范

gui.js文件是组件的配置文件。本文介绍gui.js文件的字段详情，您可以参考本文的字段说明，根据自身需求，灵活修改gui.js文件，自定义组件的配置面板。

gui.js规范示例：

```
{
  "name": "newCom",
  "attr": {
    "w": 650,
    "h": 378
  },
  "style": [...],
  "data": {...},
  "event": {...}
}
```

表 2-3 gui.js 参数说明

参数	是否必选	参数类型	描述
name	是	String	组件名称。
attr	否	attr object	组件宽高基础配置(单位：像素)。
style	是	Array of Style object	组件样式面板配置。
data	是	Object	组件数据面板配置，详情请参见 组件数据面板配置 。
event	是	Object	组件交互面板配置，详情请参见 组件交互面板配置 。

表 2-4 attr 参数说明

参数	是否必选	参数类型	描述
w	否	Number	组件宽度(单位：像素)。
h	否	Number	组件高度(单位：像素)。

组件样式配置

```
"style": [
  {
    "label": "global",
    "isExpand": true,
    "children": [
      {
        "label": "Font",
        "name": "font",
```

```
    "type": "fontfamily",
    "value": "幼圆"
  },
  {
    "label": "Color",
    "name": "color",
    "type": "color",
    "value": "rgba(70, 94, 212, 1)"
  }
]
}
```

Style参数为数组形式，其中每个数据元素将形成一个样式配置项，每个数据元素还能包含相关的子配置项，最多支持三层，具体参数说明如下：

表 2-5 Style 参数说明

参数	是否必选	参数类型	描述
label	是	String	配置项标签名称。
isExpand	否	Boolean	是否支持折叠或展开子配置项。true：支持，false：不支持。
type	是	String	配置项UI类型，DLV所开放的UI类型请参见 支持的Type类型 。
name	是	String	配置项key值，用户在命名时要注意唯一性（只限于当前组件）。
配置项的其他属性参数	否	-	不同UI类型具有不同的属性参数，请参见 UI类型介绍 。
children	否	Array of children object	配置项所拥有的子配置项，如下图所示全局样式下包含字体和近似曲线两个配置。

表 2-6 children 参数说明

参数	是否必选	参数类型	描述
label	是	String	子配置项标签名称。
type	是	String	子配置项UI类型，DLV所开放的UI类型请参见 UI类型介绍 。
name	是	String	子配置项key值，用户在命名时要注意唯一性（只限于当前组件）。
子配置项的其他属性参数	是	-	不同UI类型具有不同的属性参数，请参见 UI类型介绍 。

组件数据面板配置

data：组件的数据面板相关配置。

示例如下：

```
"data": {
  "fields": {...},
  "config": {...}
}
```

表 2-7 data 参数说明

参数	是否必选	参数类型	描述
fields	是	Object	数据面板中的字段映射区域，请参见 •fields 。
config	是	Object	数据面板中的静态数据区域，请参见 •config

- fields**

示例中fields参数如下：

```
"fields": {
  "x": {
    "value": "",
    "desc": "x"
  },
  "y": {
    "value": "",
    "desc": "y"
  },
  "s": {
    "value": "",
    "desc": "s",
    "type": "series",
    "gui": [
      {
        "label": "",
        "isChecked": true,
        "children": [
          {
            "label": "Name",
            "name": "series.name",
            "type": "input",
            "value": ""
          },
          {
            "label": "Color",
            "name": "series.color",
            "type": "color",
            "value": "rgba(195,53,53,1)"
          }
        ]
      }
    ]
  }
}
```

Fields参数为多个key: value形式的对象，其中key值为字段名称，key值所对应的value值的定义如下：

表 2-8 Fields value 参数说明

参数	是否 必选	参数类型	描述
value	是	String	所映射的源数据中的字段名称。
desc	是	String	字段名称描述，和key值保持一致。
type	否	String	值固定为series，如果存在type参数，则表明该字段为系列字段（表示同一个图表中，显示两组或两组以上数据时，用该字段区分），其中每个系列可与字段所对应的相关值做值映射。示例中，s字段包含一个默认系列和对应数值1的系列，其中每个系列还包含一些组件样式配置。
gui	否	Array	当字段类型type为series时，才存在gui参数，该参数将列出该系列所需的样式配置。

- **config**
示例中config参数如下：

```
"config": {
  "data": [
    {
      "x": "2018",
      "y": 78,
      "s": 1
    },
    {
      "x": "2016",
      "y": 55,
      "s": 1
    },
    {
      "x": "2017",
      "y": 68,
      "s": 1
    },
    {
      "x": "2018",
      "y": 48,
      "s": 1
    },
    {
      "x": "2019",
      "y": 70,
      "s": 1
    },
    {
      "x": "2020",
      "y": 85,
      "s": 1
    }
  ]
}
```

Config参数中的data参数所对应的数据即为在数据面板中显示的静态数据（上传组件包后可连通动态数据源），数据格式需根据用户的源数据进行定义。

组件交互面板配置

event：组件的交互相关配置。

示例如下：

```
"event": {
  "数据变化": {
    "enable": false,
    "fields": {
      "value": ""
    }
  }
}
```

Event为key: value形式的对象，其中key值为交互事件名称定义，value值为交互事件的相关配置，具体如下：

表 2-9 event value 参数说明

参数	是否必选	参数类型	描述
Enable	是	Boolean	是否开启该交互事件。true：开启，false：关闭。
Fields	是	Object	至少包含一个key: value，交互事件所对应的字段映射关系，其中key值为交互时所传递的参数名称，value值为在组件数据中所获取的值作为传递值。

2.4 UI 类型介绍

本文介绍DLV支持的控件（组件配置项）。您可以通过gui.json文件中定义的类型字段，来定义控件的类型和配置。

支持的 Type 类型

Type字段支持以下类型：

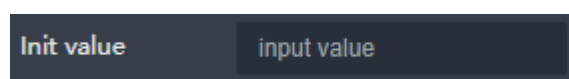
- Input：文本输入框。
- Number：数值控件，支持输入，支持定制最大值、最小值等。
- Select：下拉选择器，支持开启过滤和自定义输入。
- Color：颜色选择器。
- Checkbox：勾选框。
- Slider：滑动条。

文本输入框

表 2-10 文本输入框参数说明

字段名	是否必选	参数类型	描述
placeholder	否	String	值为空时的提示内容。
value	是	String	文本输入框的值。

示例如下：



```
{
  "label": "Init value",
  "name": "initvalue",
  "type": "input",
  "placeholder": "input value",
  "value": ""
}
```

数值控件

数值控件:

表 2-11 数值控件参数说明

字段名	是否必选	参数类型	描述
value	是	Number	数值控件的值，默认为0。
min	否	Number	可调整的最小值，默认为30000。
max	否	Number	可调整的最大值，默认为-30000。
precision	否	Number	小数数值保留位置，默认为0。
step	否	Number	调整数值的大小幅度，默认为1。

示例如下：



```
{
  "label": "Size",
  "name": "size",
  "type": "number",
  "min": 0.1,
  "max": 1.5,
  "precision": 1,
  "step": 0.1,
```

```
"value": "0.6"
}
```

下拉选择器

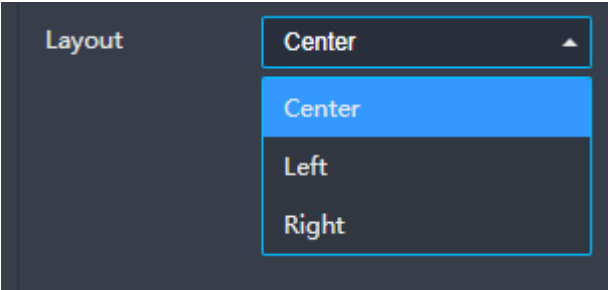
表 2-12 下拉选择器参数说明

字段名	是否 必选	参数类型	描述
value	是	string	下拉选择器选中的值。
data	是	Array<Object >	下拉选择器中包含的所有可选值。
isSearch	否	Boolean	下拉选择器是否支持搜索功能，默认为 false。

data示例如下：

```
{
  "label": "Layout",
  "name": "initvalue",
  "type": "select",
  "data": [
    {
      "key": "Center",
      "value": "center"
    },
    {
      "key": "Left",
      "value": "left"
    },
    {
      "key": "Right",
      "value": "right"
    }
  ],
  "value": "center"
}
```

图 2-1 下拉选择器



颜色选择器

表 2-13 颜色选择器参数说明

字段名	是否必选	参数类型	描述
value	是	String	颜色选择器所选择的值（可为渐变色或echarts颜色值）。
gradient	否	Boolean	是否支持渐变色，默认为true。

示例如下：

```
{
  "label": "Color",
  "name": "color",
  "type": "color",
  "value": "rgba(70,94,212,1)",
  "gradient": true
},
```

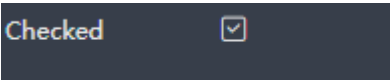
勾选框

表 2-14 勾选框参数说明

字段名	是否必选	参数类型	含义
value	是	Boolean	勾选框的值，true或false

示例如下：

图 2-2 勾选框



```
{
  "label": "Checked",
  "name": "checked",
  "type": "checkbox",
  "value": true
}
```

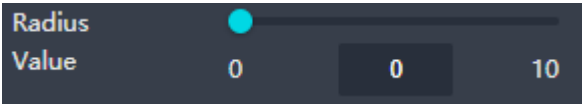
滑动条

表 2-15 滑动条参数说明

字段名	是否必选	参数类型	含义
value	是	Number	滑动条的值，默认为0。
min	否	Number	滑动条的最小值，默认为0。
max	否	Number	滑动条的最大值，默认为100。
step	否	Number	滑块滚动一格的大小，默认为1。

示例如下：

图 2-3 滑动条



```
{
  "label": "Radius Value",
  "name": "radius",
  "type": "slider",
  "value": 0,
  "min": 0,
  "max": 10,
  "step": 1
}
```

A 修订记录

发布日期	修改说明
2020-08-14	第一次正式发布。