


Vue框架二次开发快速入门

使用Vue框架的功能模块：**仪表盘、指标管理、流程引擎、基于模型创建的即席查询。**

本章节介绍如何部署二次开发环境，并以仪表盘为例让用户快速上手Vue框架相关功能的二次开发。

- 1、基础要求
- 2、环境配置
- 3、开发调试
 - 3.1 常见问题
- 4、仪表盘组件新增和修改
 - 4.1 功能修改示例
 - 4.2 组件新增示例
 - 4.2.1 入门版
 - 4.2.2 简单版
- 5、打包部署
 - 5.1 服务器打包
 - 5.2 本地打包
 - 5.3 部署
- 6、学习资料汇总

1、基础要求

 本次环境配置，只需在 “扩展包开发环境” 的基础上增加二开扩展包环境即可，用户预先安装扩展包开发环境，部署方式请参考 [扩展包开发环境部署](#)。

- (1) 通过安装包：[node-v16.14.2-win-x64.zip](#) 安装node环境，详情可参考：[node环境安装文档](#)；
- (2) 此版本已经自带npm，对于二开人员：
 - 学习命令操作可参考：[npm命令使用文档](#)；
 - 学习框架开发可参考：[Vue官网](#)；

2、环境配置

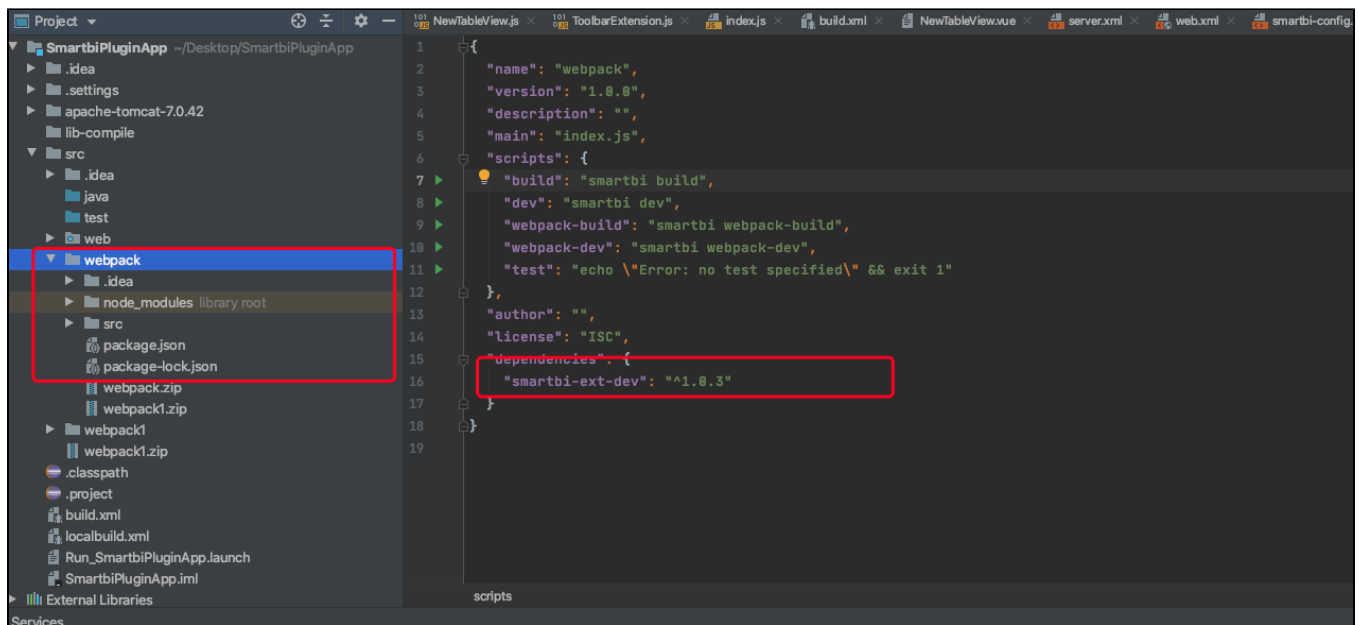
 (1) V10.5以下版本不支持仪表盘二次开发；

(2) 对于仪表盘二次开发：可在原有扩展包基础上加上包：[webpack .zip](#)；（包含下面两个示例的基础文件，下载后按说明可直接使用）

- (1) 进入扩展包根目录的src；
- (2) 解压webpack项目，打开终端控制台cd 到webpack目录，运行下载smartbi二开环境基础包命令：

```
npm install smartbi-ext-env
```

得到的目录结构如下：



在二开环境中，常用命令如下：

- 开发调试命令：

```
npm run dev //
```

- 打包命令：

```
npm run build //
```

3、开发调试

运行开发调试命令：

```
npm run dev
```

每当代码修改的时候，系统会帮你自动编译，刷新页面即可看到新的效果。

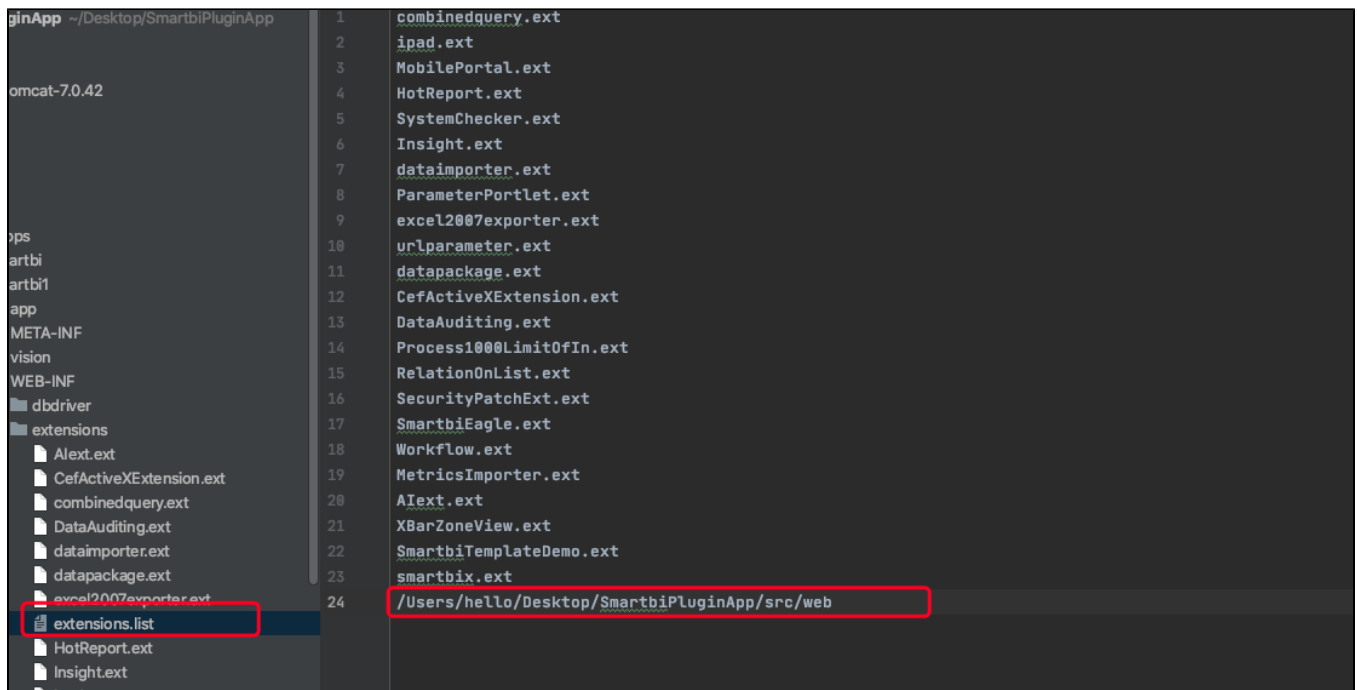
3.1 常见问题

问题：为什么改了代码，没有效果？

解决方法：请仔细查看以下几个步骤：

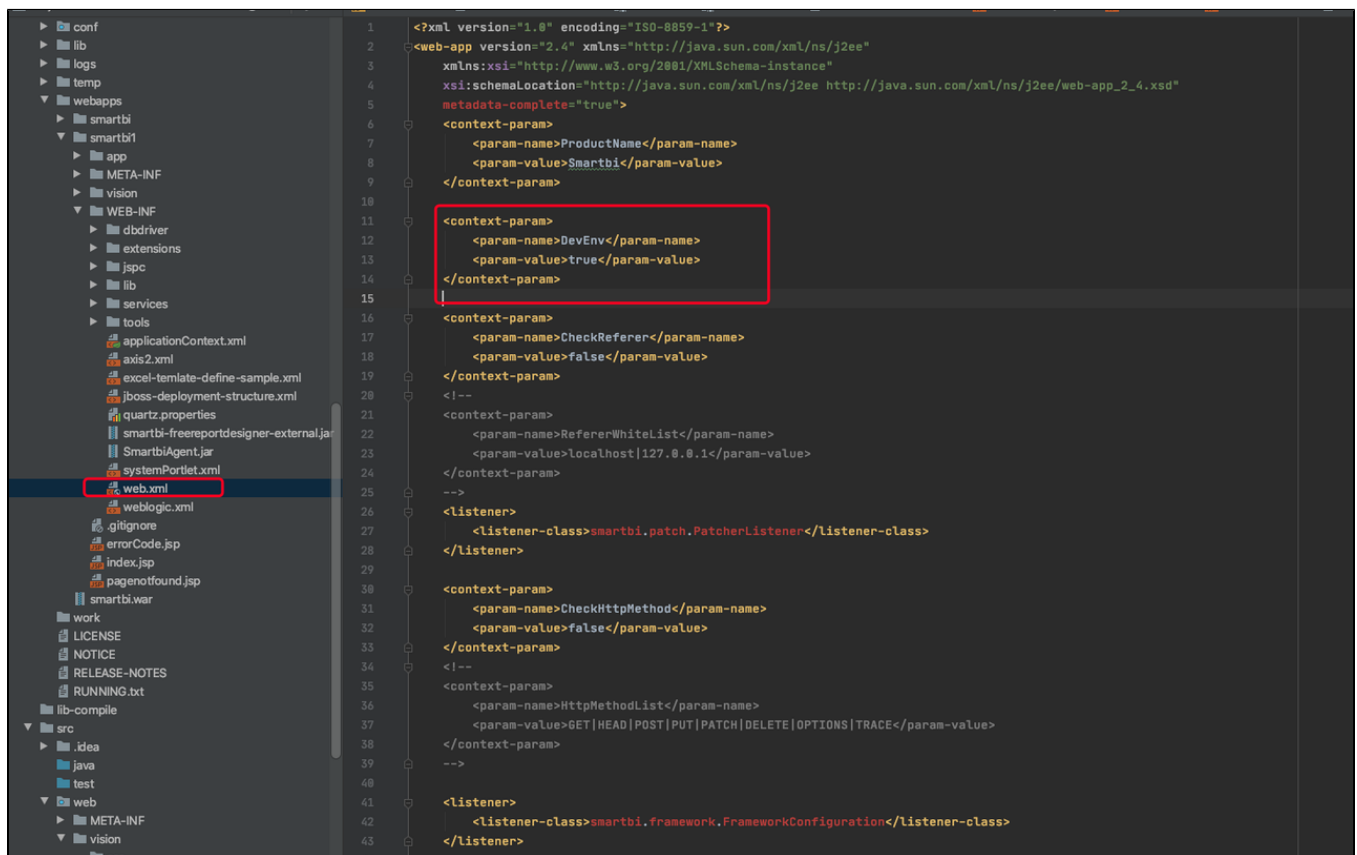
1、如果要调试smartbi，最好在url中增加debug=true参数（如：<http://192.168.1.10:16000/smartbi/vision/index.jsp?debug=true>），否则扩展包会有缓存，代码修改后不会更新。

2、配置extension.list;



3、修改SmartbiPluginApp/apache-tomcat-7.0.42/webapps/smartbi/WEB-INF下的web.xml文件：

- 在文件中加入 DevEnv 对应的四行代码；
- 在文件中找到 ProductName 的 <context-param>；
- 在其下面添加 DevEnv 对应的四行<context-param>，其值为 true。



具体调试技巧可参考：[调试&定位技巧](#)。

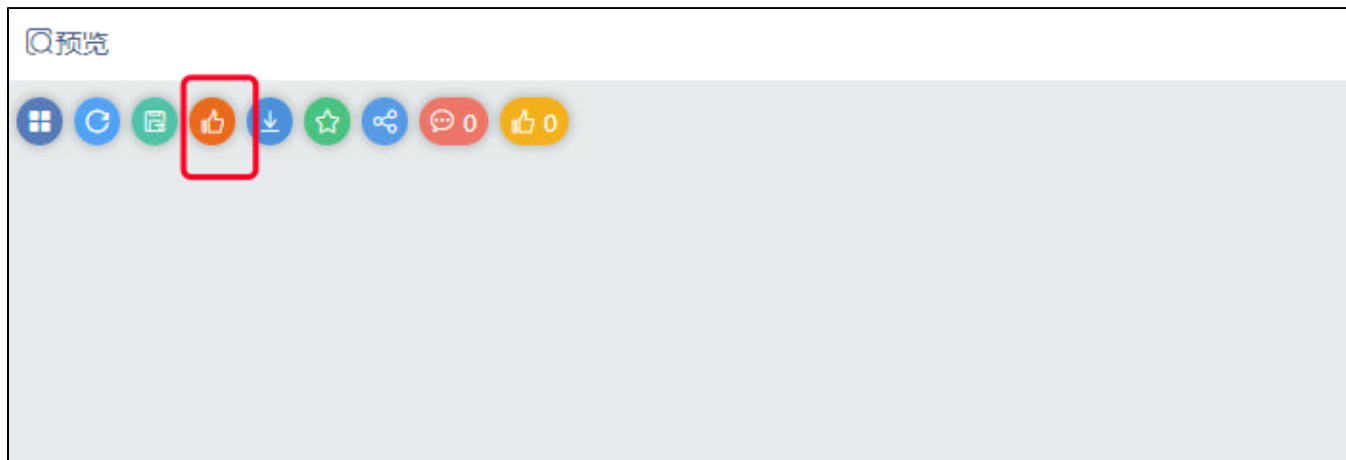
4、仪表盘组件新增和修改

4.1 功能修改示例

(1) 需求场景

在仪表盘预览菜单栏，点击工具栏弹出hello。

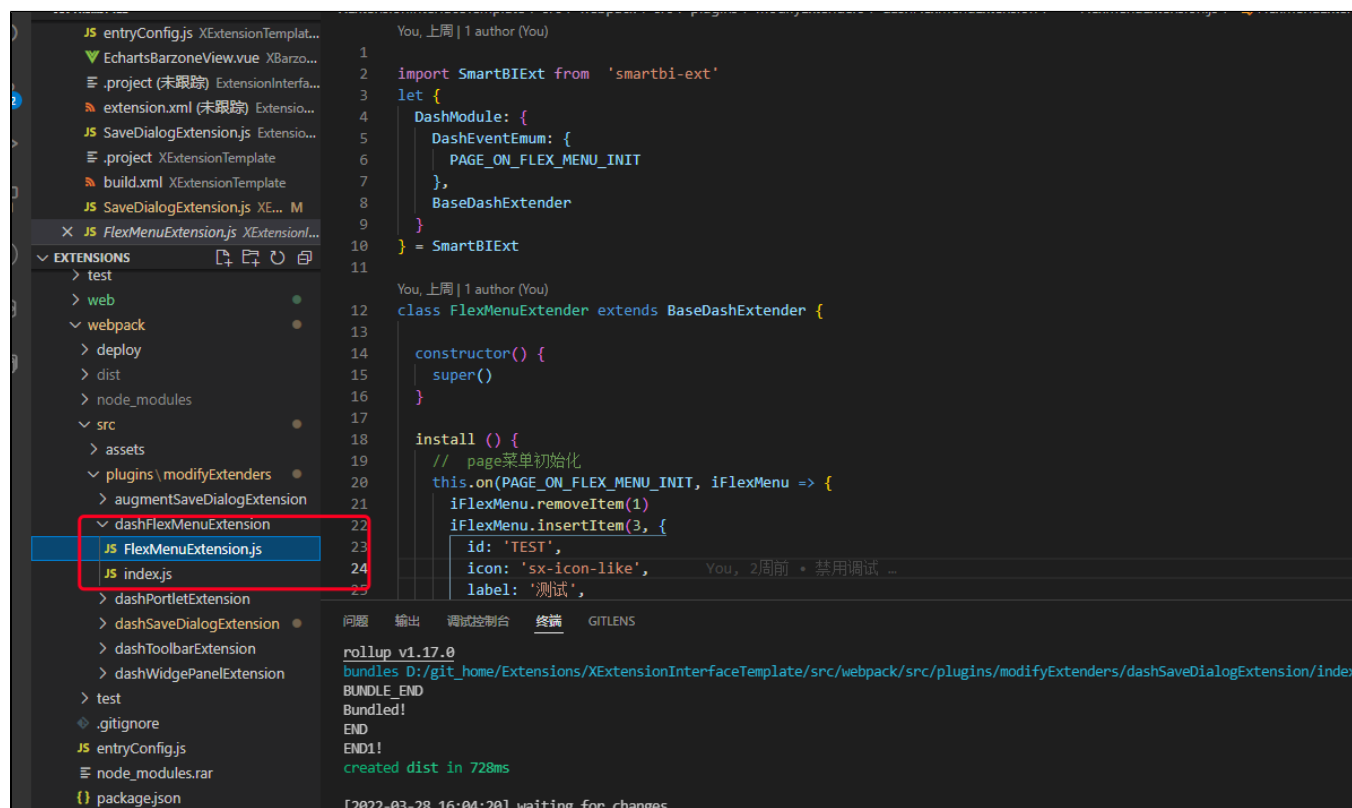
(2) 示例效果



(3) 操作步骤

a. 在根目录下的src/webpack/src/plugins/modifyExtenders下创建组件文件夹，下面以仪表盘预览菜单栏为例：

- 添加文件夹dashFlexMenuExtension;
- 在文件夹下创建index.js及FlexMenuExtension.js两个文件;



b. 在index.js文件中实现provideMetaInfo方法，FlexMenuExtension.vue导出的是一个FlexMenuExtension对象：

```
import FlexMenuExtension from './FlexMenuExtension'
const provideMetaInfo = () => {
  //
  return { implement: FlexMenuExtension }
}
export default {
  provideMetaInfo
}
```

c. FlexMenuExtension.js:



说明

- BaseDashExtender是一个基类，系统提供了内置方法
- install是入口方法
- on方法是向系统添加某个事件的处理程序，系统在这个时机就会触发此方法回调。

```
import SmartBIExt from 'smartbi-ext'
let {
  DashModule: {
    DashEventEnum: {
      PAGE_ON_FLEX_MENU_INIT
    },
    BaseDashExtender
  }
} = SmartBIExt

class FlexMenuExtender extends BaseDashExtender {

  constructor() {
    super()
  }

  install () {
    // page
    this.on(PAGE_ON_FLEX_MENU_INIT, iFlexMenu => {
      iFlexMenu.removeItem(1)
      iFlexMenu.insertItem(3, {
        id: 'TEST',
        icon: 'sx-icon-like',
        label: '',
        color: '#ed6b1f',
        handler: () => {
          alert('')
        }
      })
    })
  }
}

export default FlexMenuExtender
```

4.2 组件新增示例



注意

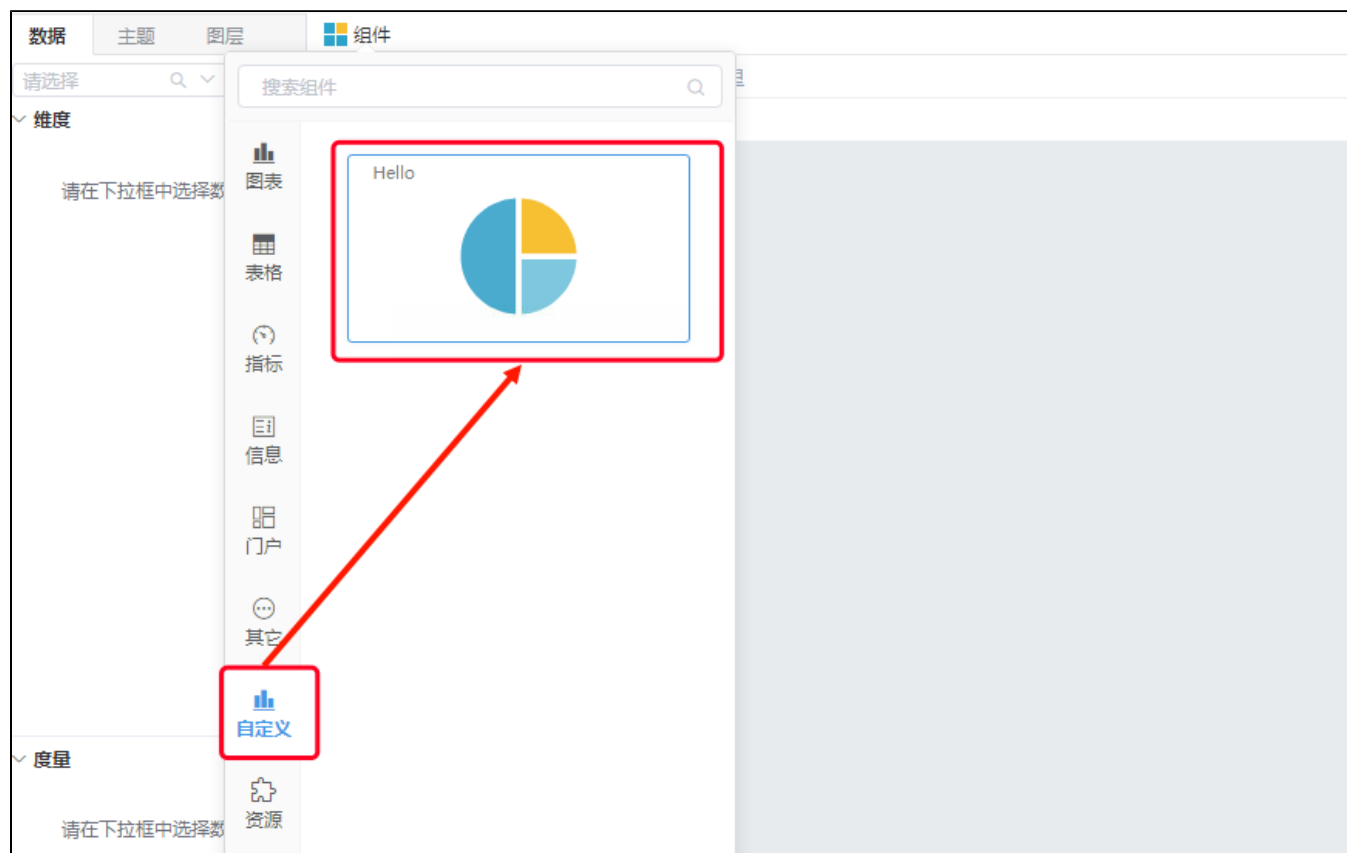
组件新增场景仅支持仪表盘。

4.2.1 入门版

(1) 需求场景

在仪表盘的组件选择器中，新增一个组件，组件中显示一个简单的文字hello。

(2) 示例效果



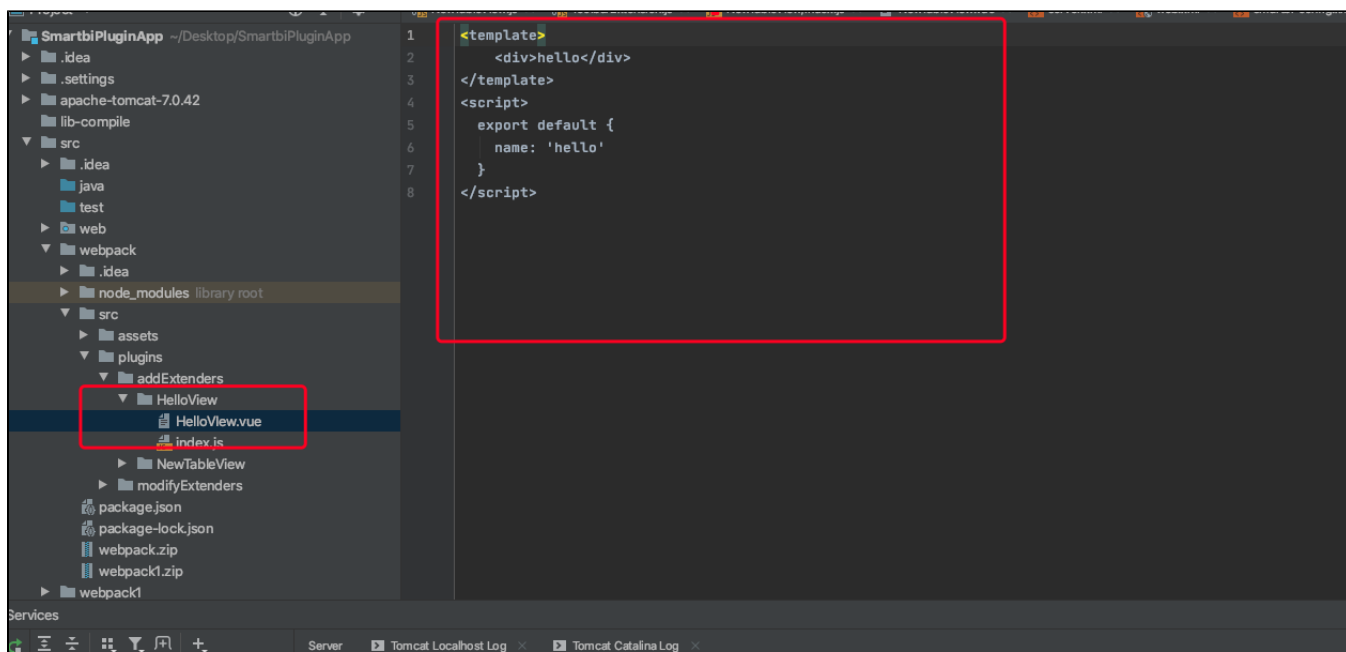
(3) 操作步骤

说明

- index文件是入口文件；
- providerMetaInfo是入口方法；
- vue文件是具体实现的业务文件；

a. 在根目录下的src/webpack/src/plugins/addExtenders中创建组件文件夹：

- 添加文件夹helloView
- 在文件夹下创建index.js及HelloView.vue



b. 在index.js文件中实现provideMetaInfo方法，返回的是一个对象，具体对象的属性可以参考[仪表盘组件接口文档](#)；

```
import HelloView from './HelloView.vue'

const provideMetaInfo = () => { //
  return {
    name: 'DEMO', //
    type: 'HELLO', //
    svg: 'db-left-widget__chart-pie',
    entry: { //
      title: 'Hello', //
      order: 15, //
    },
    //
    portletImplement: HelloView
  }
}

export default {
  provideMetaInfo
}
```

c. 在HelloView.vue中编写自己的业务代码

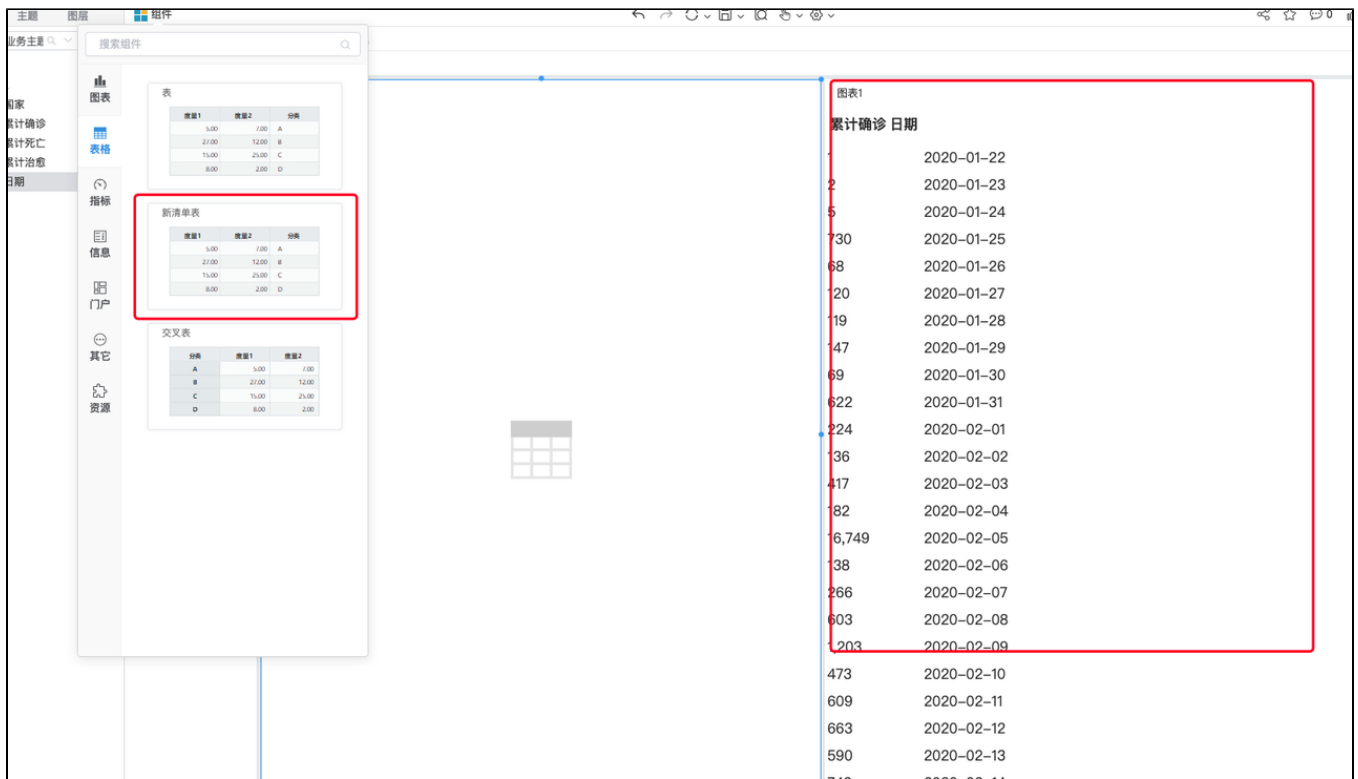
```
<template>
  <div>hello</div>
</template>
<script>
export default {
  name: 'hello'
}
</script>
```

4.2.2 简单版

(1) 需求场景

在仪表盘的组件选择器中，新增一个清单表组件，具备取数功能。

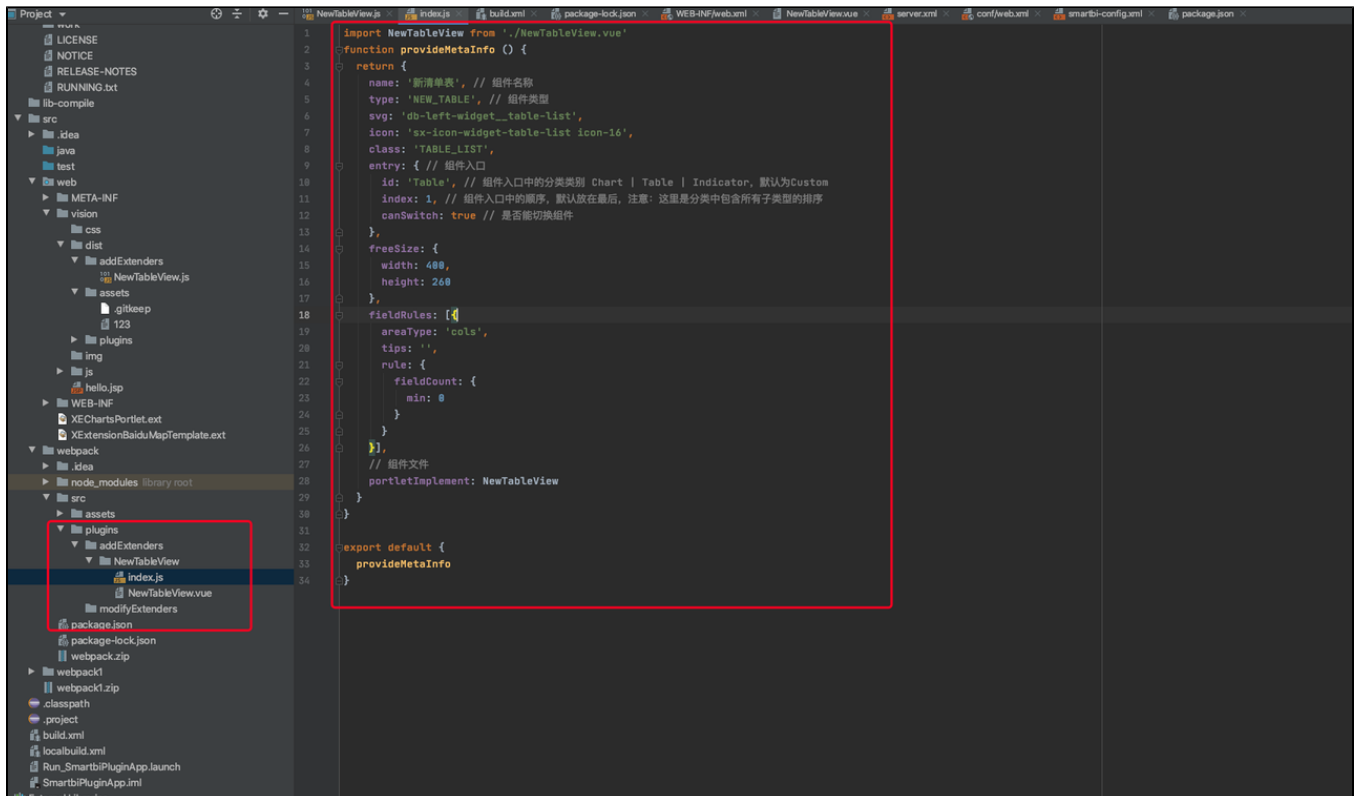
(2) 示例效果



(3) 操作步骤

a. 在根目录下的src/webpack/src/plugins/addExtenders下创建组件文件夹：

- 添加文件夹newTableView；
- 在文件夹下创建index.js及NewTableView.vue；



b. 在index.js文件中实现provideMetaInfo方法，返回的是一个对象，具体对象的属性可以参考[仪表盘组件接口文档](#)；


```

import NewTableView from './NewTableView.vue'
function provideMetaInfo () {
  return {
    name: '', //
    type: 'NEW_TABLE', //
    svg: 'db-left-widget__table-list',
    icon: 'sx-icon-widget-table-list icon-16',
    class: 'TABLE_LIST',
    entry: { //
      id: 'Table', // Chart | Table | IndicatorCustom
      index: 1, //
      canSwitch: true //
    },
    freeSize: {
      width: 400,
      height: 260
    },
    fieldRules: [{ //
      areaType: 'cols',
      tips: '',
      rule: {
        fieldCount: {
          min: 0
        }
      }
    }],
    //
    portletImplement: NewTableView
  }
}

export default {
  provideMetaInfo
}

```

(说明: **fieldRules**: 是字段区配置, 可以理解为行列区和标记区规则配置)

c. 在NewTableView.vue中编写自己的业务代码:

- Smartbi 节点下main节点下, **refresh**方法, 是系统通知组件更新的一个时机
- this.pageProxy.api.**fetchPortletData**(queryFields)是组件主动调用系统取数api, 进行取数。

```

<template>
  <div slot="notEmpty" style="height: 100%;" >
    <table v-if = "gridDataModel">
      <th>
        <td
          v-for="field in gridDataModel.getFields()"
          :key="field.uniqueId">
            {{ field.alias }}
        </td>
      </th>
      <tr>
        v-for="row in gridDataModel.getRowCount()"
        :key="row">
          <td
            v-for="field in gridDataModel.getFields()"
            :key="row + field.uniqueId">
              {{ gridDataModel.getDisplayValueByRowAndUid(row - 1, field.uniqueId) }}
            </td>
          </tr>
        </table>
      </div>
    </template>
    <script>
    export default {
      name: 'NewTableView',

```

```

data () {
  return {
    options: {},
    gridDataModel: null
  }
},
props: {
  pageProxy: {}
},
smartbi: {
  main: {
    refresh () {
      this.fetchData()
    }
  }
},
methods: {
  async fetchData () {
    console.log(123)
    let queryFields = this.generateQueryFields()
    if (queryFields.length === 0) {
      return
    }
    let options = await this.pageProxy.api.fetchPortletData(queryFields)
    let {
      gridDataModel, //
      gridData, //
      queryFields: fields, //
      drillableFields, //
      drilledFieldPath //
    } = options
    this.options = options
    this.gridDataModel = gridDataModel
    // commitRefresh
    this.pageProxy.portlet.commitRefresh(true)
  },
  generateQueryFields () {
    let queryFields = this.pageProxy.field.getFieldsByZones(['cols'])
    // groupBygroupBygroupBy
    let tooltipFields = this.pageProxy.field.getFieldsByZone('tooltip')
    return queryFields.concat(tooltipFields.map(field => {
      return this.pageProxy.field.createGroupByField(field, false)
    })))
  }
}
}
</script>

```

5、打包部署

- **本地打包**：系统会帮自动打包vue文件变成js，本地直接再打包成扩展包；
- **服务器打包**：开发者要运行打包命令，将vue文件打包成js文件，然后开发者将扩展包代码提交到仓库，服务器打包的时候会运行build.xml 脚本进行打包；
- **两者区别**：本地打包帮你直接打包扩展包，服务器打包是将你的编译后的js文件提交到服务器打包；
- **部署**：将服务器打好的包，打开系统监控，扩展包管理页面进行上传。

5.1 服务器打包

- (1) 运行npm run build命令；
- (2) 将扩展包代码提交仓库；
- (3) 当服务器打包的时候会运行扩展包代码中的的build.xml 文件 打包成扩展包；

5.2 本地打包



本地打包非必要时候，不建议使用！

如果有需要本地打包的话，请配置添加localBuild.xml文件。

- (1) 复制原来的build.xml文件，修改名字为localBuild.xml；
- (2) 在target标签下添加排除webpack路径下的内容；

```
50         <arg line="/c &quot;cd ${basedir}/src/webpack &amp;&amp; npm run build &quot; "/>
51     </exec>
52 </target>
53
54 <target name="jar">
55     <echo file="${basedir}/src/web/META-INF/version.txt" message="${today}" />
56     <jar destfile="${basedir}/dist/${ext_name}.ext" duplicate="preserve">
57         <fileset dir="${basedir}/src/web">
58             <exclude name="**/.cvsignore" />
59             <exclude name="**/webpack/" />
60         </fileset>
61     </jar>
62     <!-- <delete file="${basedir}/src/web/META-INF/version.txt"/> -->
```

```
<exclude name="**/webpack/" />
```

- (3) 添加webpack命令；

```
<target name="webpack" description="webpack npm build">
    <exec executable="cmd.exe">
        <arg line="/c &quot;cd ${basedir}/src/webpack &amp;&amp; npm run build &quot; "/>
    </exec>
</target>
```

```
<target name="webpack" description="webpack npm build">
    <exec executable="cmd.exe">
        <arg line="/c "cd ${basedir}/src/webpack && npm run build " "/>
    </exec>
</target>
```

- (4) 修改dist命令编译部分如下，添加webpack相关的编译；

```
</target>

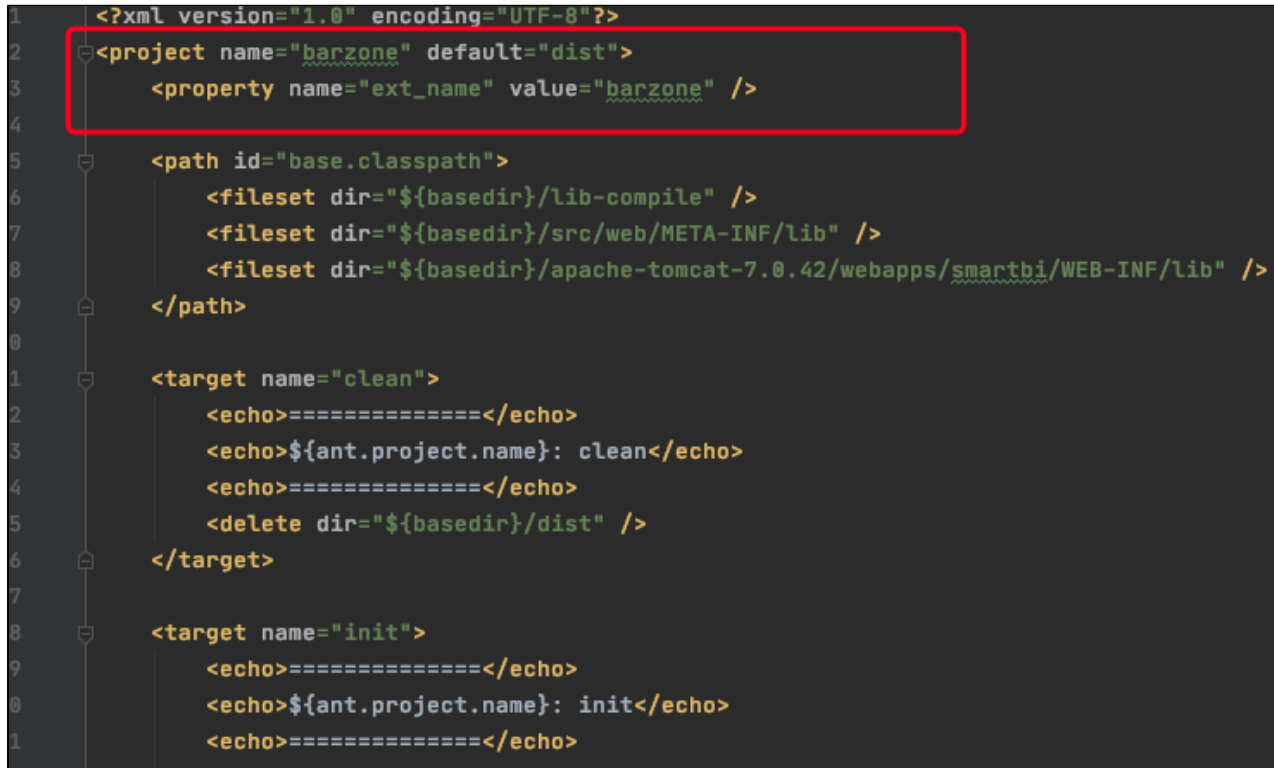
<target name="dist" depends="init">
    <parallel failonany="true">
        <echo>=====</echo>
        <antcall target="webpack" />
        <echo>webpack npm build</echo>
        <echo>${ant.project.name}: dist</echo>
        <echo>=====</echo>
        <antcall target="compile" />
    </parallel>
    <antcall target="jar" />
</target>
</project>
```

```

<target name="dist" depends="init">
  <parallel failonany="true">
    <echo>=====</echo>
    <antcall target="webpack" />
    <echo>webpack npm build</echo>
    <echo>${ant.project.name}: dist</echo>
    <echo>=====</echo>
    <antcall target="compile" />
  </parallel>
  <antcall target="jar" />
</target>

```

(5) 修改打包名称:

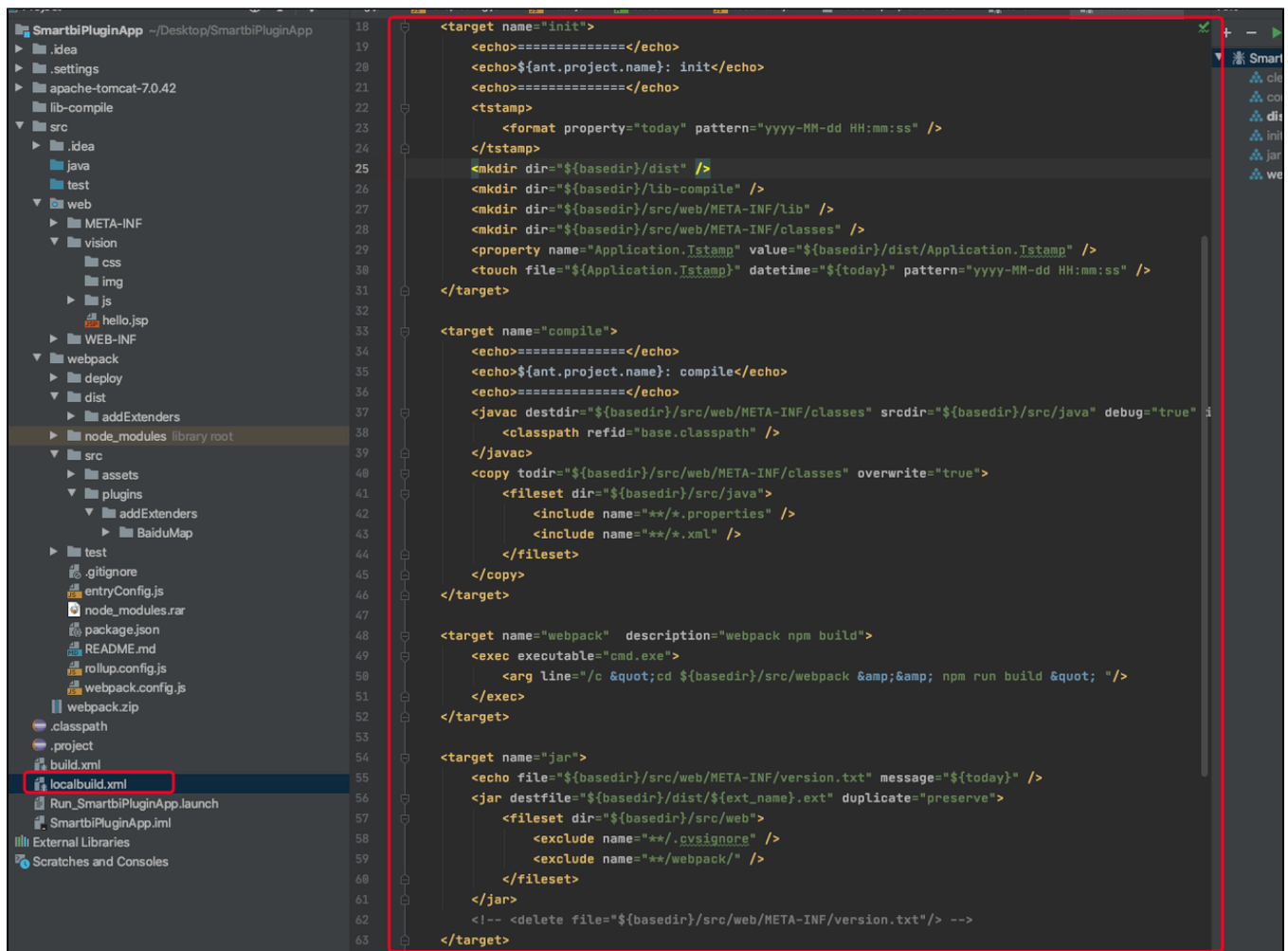


```

1  <?xml version="1.0" encoding="UTF-8"?>
2  <project name="barzone" default="dist">
3    <property name="ext_name" value="barzone" />
4
5    <path id="base.classpath">
6      <fileset dir="${basedir}/lib-compile" />
7      <fileset dir="${basedir}/src/web/META-INF/lib" />
8      <fileset dir="${basedir}/apache-tomcat-7.0.42/webapps/smartbi/WEB-INF/lib" />
9    </path>
10
11   <target name="clean">
12     <echo>=====</echo>
13     <echo>${ant.project.name}: clean</echo>
14     <echo>=====</echo>
15     <delete dir="${basedir}/dist" />
16   </target>
17
18   <target name="init">
19     <echo>=====</echo>
20     <echo>${ant.project.name}: init</echo>
21     <echo>=====</echo>

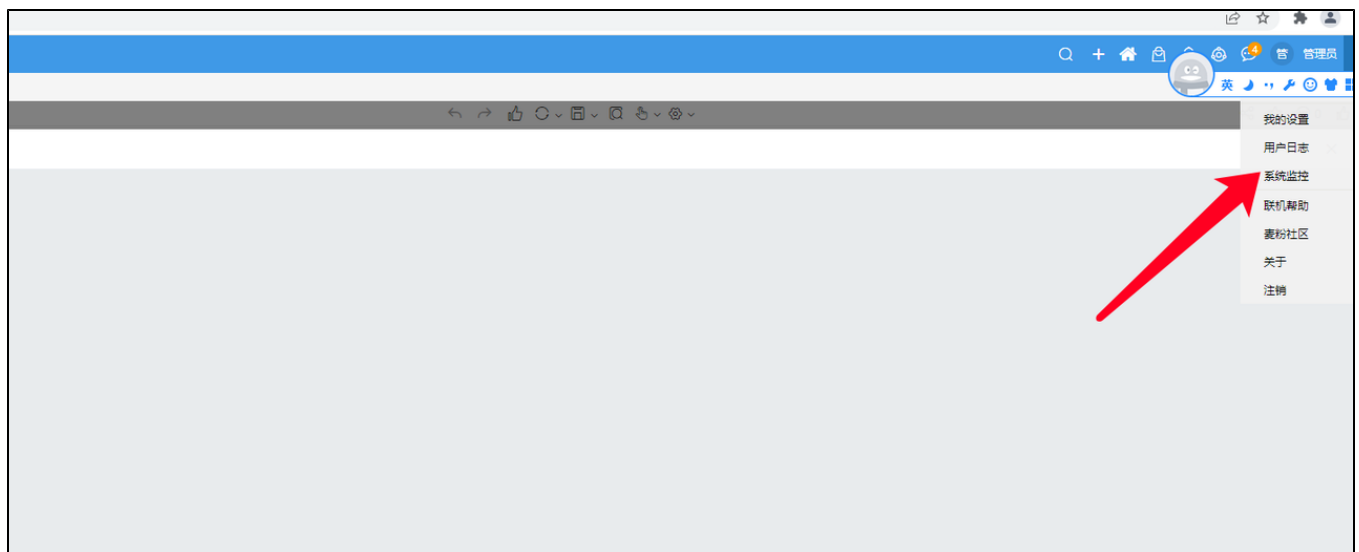
```

(6) 效果如下:



5.3 部署

(1) 打开系统监控



(2) 上传扩展包

服务器:

设置

当前服务器

导出所有

概述

监控

性能

网络

日志

会话

缓存

位图回收活动

堆打印

对象池

配置

扩展包

所有字体

wsd地址

log信息

SQL/MDX监控

导出引擎

模型管理

实验监控

服务监控

计算节点

作业流监控

提示: 进行上传、应用、禁用、重新加载扩展包操作期间, 服务器不能正常访问, 执行这些操作时建议选择服务器空闲时间。

名称	别名	描述	版本	打包日期	优先级	启用
1	ipadextension	MobileExtension	2.0	2022-01-22 16:28:22	20	是
2	mobileportal	移动浏览器	1.0		40	是
3	smartbieagle	SmartbiEagle	1.0	2022-01-22 16:30:32	80	是
4	smartbi	SmartbiX	1.0		100	是
5	augmenteddataset	AugmentedDataSet	1.0		100	是
6	xextensioninterfacetemplate	XExtensionInterfaceTemplate	1.0	2022-03-24 09:50:29	100	是
7	smartbitemplatedemo		1.0		100	是
8	xbarzoneview	XBarZoneView	1.0		100	是
9	alex	Alex	1.0	2022-01-22 16:30:33	100	是
10	relationlist	表关系列表显示	1.0		100	是
11	process1000limitofin	处理in函数1000限制的函数	1.0	2022-01-22 16:30:32	100	是
12	dataauditing	DataAuditing	1.0	2022-01-22 16:28:39	100	是
13	celactiveextension	CefActiveExtension	1.0		100	是
14	urlparameter	URLParameterExtension	1.0	2022-01-22 16:28:34	100	是
15	excel2007exporter	Excel2007Exporter	1.0		100	是
16	parameterportlet	ParameterPortlet	1.0		100	是
17	dataimporter	数据导入	1.0	2022-01-22 16:28:34	100	是
18	insight	Insight	1.0		100	是
19	systemchecker	系统检查器	1.0		100	是
20	hotreport	HotReport	1.0		100	是
21	combinedquery	CombinedQuery	1.0	2022-01-22 16:28:22	100	是
22	metricsimporter	指标导入			200	是
23	workflow	workflow	1.0		200	是
24	datapackage	DataPackage	1.0	2022-01-22 16:28:33	200	是

(3) 选择扩展包

提示: 进行上传、应用、禁用、重新加载扩展包操作期间, 服务器不能正常访问, 执行这些操作时建议选择服务器空闲时间。

名称	别名	描述	版本	打包日期	优先级	启用
	MobileExtension	移动设备扩展包, 支持IPAD、IPHONE与安卓设备	2.0	2022-01-22 16:28:22	20	是
	移动浏览器	移动浏览器				
	SmartbiEagle	SmartbiEagle				
	SmartbiX	SmartbiX				
	AugmentedDataSet	AugmentedDataSet				
	XExtensionInterfaceTemplate	XExtensionInterfaceTemplate				
	XBarZoneView	XBarZoneView				
	Alex	Alex				
	表关系列表显示	表关系列表显示				
	处理in函数1000限制的函数	处理in函数1000限制的函数				
	DataAuditing	DataAuditing				
	CefActiveExtension	CefActiveExtension				
	URLParameterExtension	URLParameterExtension				
	Excel2007Exporter	Excel2007Exporter				
	ParameterPortlet	ParameterPortlet				
	数据导入	数据导入				
	Insight	Insight				
	系统检查器	系统检查器				
	HotReport	HotReport				
	CombinedQuery	CombinedQuery				
	指标导入	指标导入功能			200	是
	workflow	workflow	1.0		200	是
	DataPackage	DataPackage	1.0	2022-01-22 16:28:33	200	是

打开

名称: XExtensionInterfaceTemplate.ext, 修改日期: 2022/3/24 9:50, 类型: EXT 文件, 大小: 20 KB

文件名(N):, EXT 文件 (*.ext), 打开(O), 取消

6、学习资料汇总

- (1) node环境安装: <https://www.cnblogs.com/xinaixia/p/8279015.html>
- (2) npm命令使用: <https://www.jianshu.com/p/4643a8e43b79>
- (3) Vue框架开发: <https://cn.vuejs.org/index.html>