目 录

[一、 启动系统 2](#_Toc4743957)

[二、 系统运行 2](#_Toc4743958)

[三、 常用功能操作 3](#_Toc4743959)

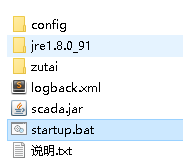
[四、 常用图元属性 7](#_Toc4743960)

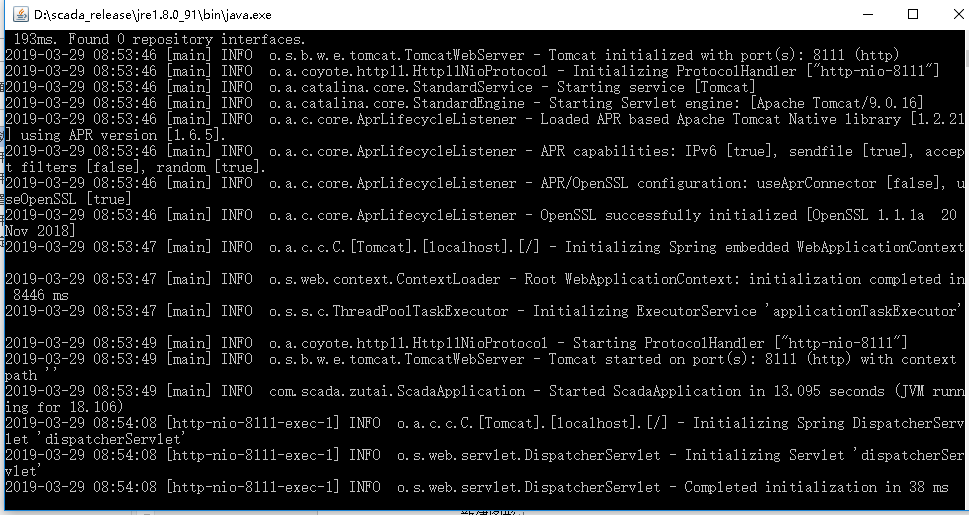
[五、 内置脚本函数 8](#_Toc4743961)

[六、 常用方法 11](#_Toc4743962)

[七、 自定义图元 14](#_Toc4743963)

# 启动系统

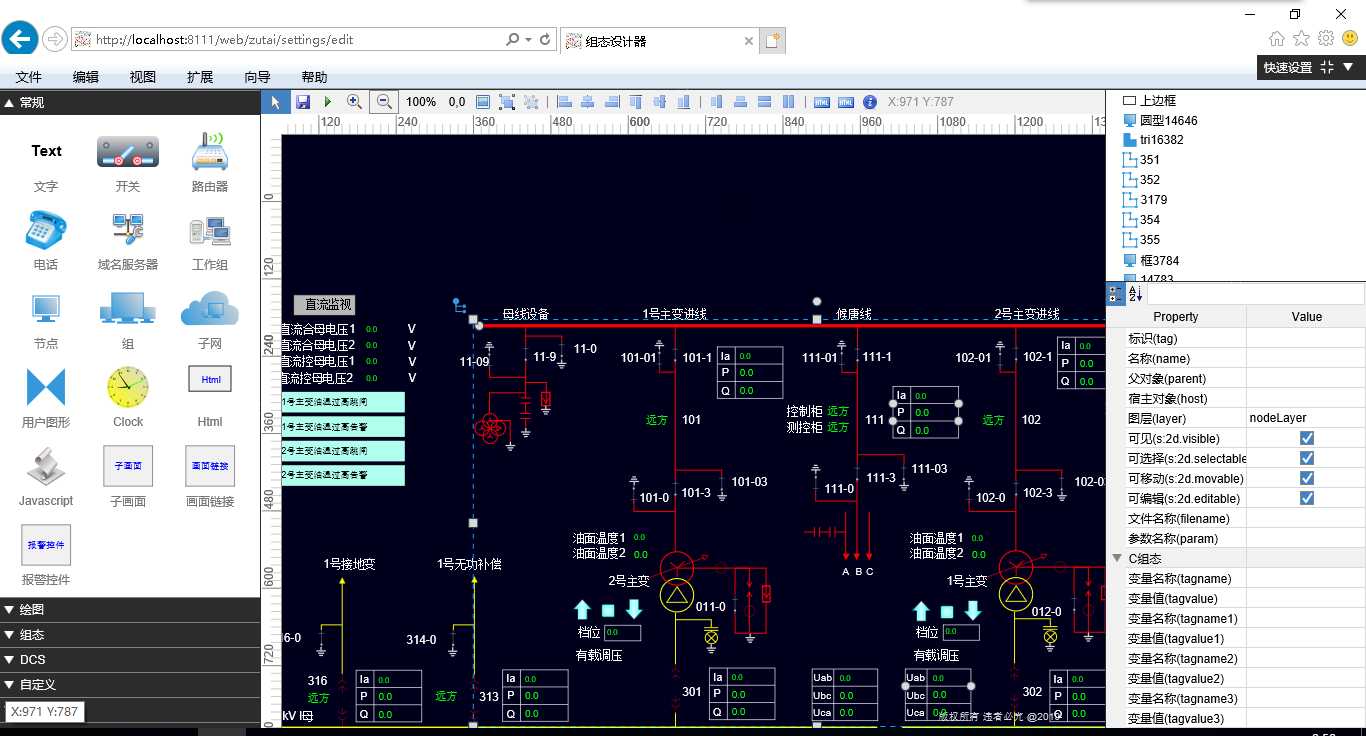
打开软件目录，运行startup.bat,启动完成后显示如下界面，表示系统启动成功。



建议安装目录为英文目录。

# 系统运行

在浏览器中打开<http://localhost:8111/web/zutai/settings/edit>进入组态界面编辑界面。

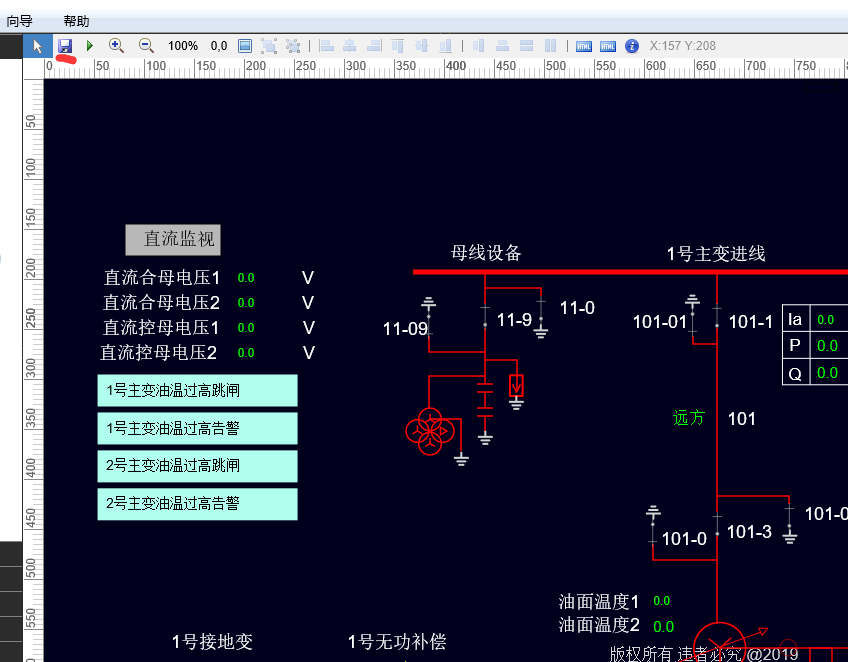


其中最顶端为工具栏，左侧为图元区域，右侧为属性窗口，最中间部分为绘图区域。

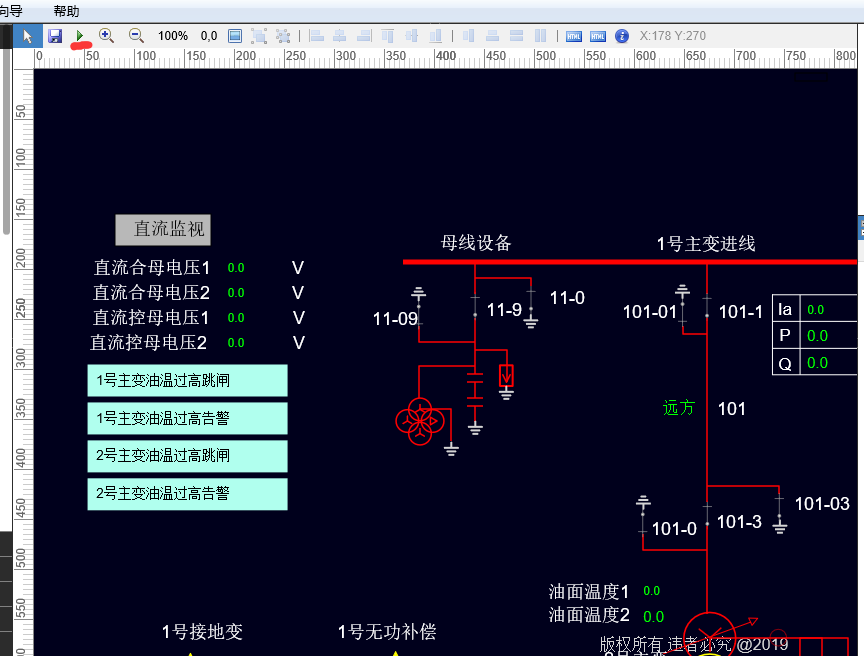
左侧图元工具条可通过拖动到绘图区域。选择图形中具体一个图元，右侧属性窗口将显示图元相关属性，可对图元中的属性进行设置。

# 图形运行

图形组态完成后，先保存，然后保存。



保存成功后，可点击运行按钮。

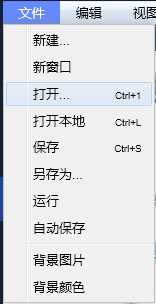


# 常用功能操作

新建图形



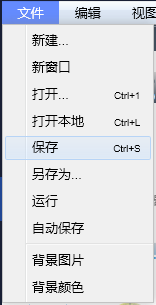
打开图形



打开本地(在本地绘制图形的同时，为防止意外数据丢失，系统会自动保存到本地)



保存文件



设置图形背景图片



设置图形背景颜色



图元属性窗口



具体图形对应属性需根据图形决定。

图元脚本编辑



选择图形中的图元，单击鼠标右键，弹出右键菜单，图元支持数据更新脚本及鼠标单击脚本编写。其中：

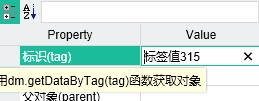
数据更新脚本：系统运行时，对应图元数据发生变化时，触发数据更新脚本执行。

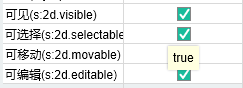
鼠标点击脚本：系统运行时，鼠标单击相应图元时，触发点击脚本执行。

脚本语言为javascript语言。

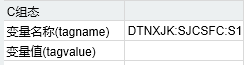
# 常用图元属性

因每个图元都有共性属性也存在一部分自定义属性,需要根据不同图元设置。下面仅列出常用集中属性。





1. 图形挂点



系统支持同一个图元最多可挂载10个点。

# 内置脚本函数

/\*

\* 转换特殊字符

\*/

**function** replaceSpecChars(str)

/\*

\*本地时间转换为GTM

\*/

**function** local2GTM(str)

/\*

\* GTM转换为本地时间

\*/

**function** gtm2Local(str)

/\*

\*询问对话框

\*/

**function** questionbox(msg)

/\*

\*画面跳转

\* name 画面名称（文件名）

\* param 画面参数，一般情况为空

\*/

**function display(name, param)**

/\*

\* 消息提醒

\* title 消息框标题

\* msg 消息内容

\*/

**function** showMessage(title, msg)

/\*

\* 输入对话框

\* title 消息框标题

\* txt 输入默认值

\* action 回调函数 如 function checkval(val) 其中val为返回值

\*/

**function getInput(title, txt, action)**

/\*\*

\* 数字输入框

\* @param title

\* @param name

\* @param val

\* @param action

\* @returns

\*/

**function getNInput(title,name, val, action)**

/\*\*

\* 控制指令下发

\* @devsNames 设备名集合 多个以逗号结束

\* @param cmds 控制命令集合 多个以逗号结束

\* @param cmdPara1s 控制参数1 具体根据实际情况确定

\* @param cmdPara2s 控制参数2 具体根据实际情况确定

\* @returns

\*/

**function executeCmd(devsNames,cmds,cmdPara1s,cmdPara2s)**

/\*\*

\* 查看指标趋势

\* @param 参数 值如所示：tags=

\* @returns

\*/

**function viewrealtrend(param)**

//风机转动

//node 转动对象

//speed 转动速度 数字

//direct 转动方向 true 顺时钟方向 false 逆时钟方向

//是否转动

**function** Rotation(node,speed,direct,flag)

/\*\*

\* 十进制转换成二进制

\* **@param** data

\* **@returns**

\*/

**function** decimalToBinary(data)

/\*\*

\* 二进制转换成十进制

\* **@param** data

\* **@returns**

\*/

**function** binaryToDecimal(data)

/\*\*

\* 按位取值

\* **@param** num 数值

\* **@param** bit 第几位

\* **@returns**

\*/

**function** valueAtBit(num, bit) {

# 常用方法

官网支持文档：

<http://www.hightopo.com/guide/guide-en/core/treetableview/examples/example_structure.html>

属性获取或设置常用方法

setAttr("属性名",属性值);

setStyle("属性名",属性值);

/\*\*

\*

\* 获取画面图元对象

\*/

**var fanNode=dataModel.getDataByTag("风机139");**

/\*\*

\*

\* 设置画面图元对象值

\*/

**var val=0;**

**fanNode.setAttr("rotation",val);**

**fanNode.iv();**

/\*\*

\*

\* 设置画面图元对象值

\*/

**var switchNode=dataModel.getDataByTag("就地-远方4863");**

**var curVal=switchNode.getAttr("tagvalue");**

**//获取点值**

**var v = data.a("tagvalue");**

**//设置图元属性**

**fanNode.setAttr("title","ffff");**

**//更新**

**fanNode.iv();**

**示例**

**就地远方控制**

**var switchNode=dataModel.getDataByTag("就地-远方4863");**

**var curVal=switchNode.getAttr("tagvalue");**

**curVal=curVal==""?"0":curVal;**

**if(curVal=="0")**

**{**

**switchNode.setAttr("tagvalue",1);**

**}**

**else**

**{**

**switchNode.setAttr("tagvalue",0);**

**}**

**switchNode.iv();**

**//控制指令下发（带数值输入）**

**getNInput('变量输入','调整功率',0,test);**

**function test(val)**

**{**

**var funa=executeCmd('DTNXJK:SJCSFC:S1:S001','dev\_control\_adjust',val,'');**

**}**

**风机转动效果实现**

1. **在页面增加script脚本图元**



1. **在编写转动效果脚本**

//编辑脚本

var flag=true;

//node 转动对象

//speed 转动速度 数字

//direct 转动方向 true 顺时钟方向 false 逆时钟方向

function FanRotation(node,speed,direct) {

currentRotation = 0;

lastTime = new Date().getTime();

setInterval(function() {

//是否转动标记

if(flag)

{

var step = speed / 250;

var time = new Date().getTime();

var deltaTime = time - lastTime;

if(direct)

{

currentRotation += (deltaTime \* Math.PI / 180 \* step);

}

else

{

currentRotation += -(deltaTime \* Math.PI / 180 \* step);

}

node.setAttr("rotation",currentRotation);

node.iv();

lastTime = time;

}

}, 10);

};

var fanNode=dataModel.getDataByTag("风机314");

//设置叶片颜色

fanNode.setAttr("blade.color","blue");

fanNode.iv();

FanRotation(fanNode,25,true,true);

//设置停止转动标记

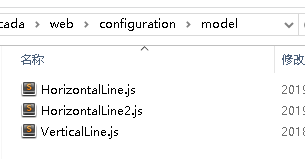
//flag=false;

# 自定义图元

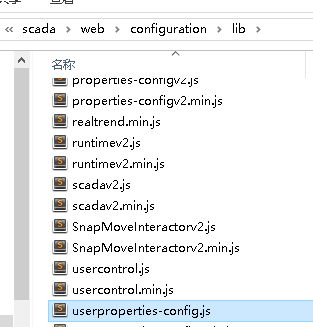
提供自定义图元集成方式。如需自定义编写图元，需提前掌握hightopo 脚本语言开发，具体参考<https://hightopo.com/>官网提供api说明。另因内置库版本的区别，可能会与官网存在一定的区别。

具体可参考\web\configuration\model目录下已有图元编写方式编写自定义图元

编辑好的图元直接保存到该目录即可，系统自动装载。另外需注意，为了保证图元的兼容性和识别性，文件名以英文名方式命名。

当个自定义图元保存到目录下即可,具体图元写法参考已有图元。

自定义图元属性在此文件下增加

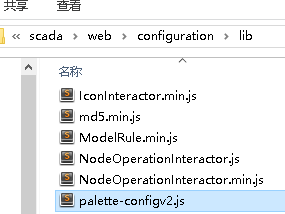


在文件末尾增加



主要属性命名规则图元名称+ \_properties

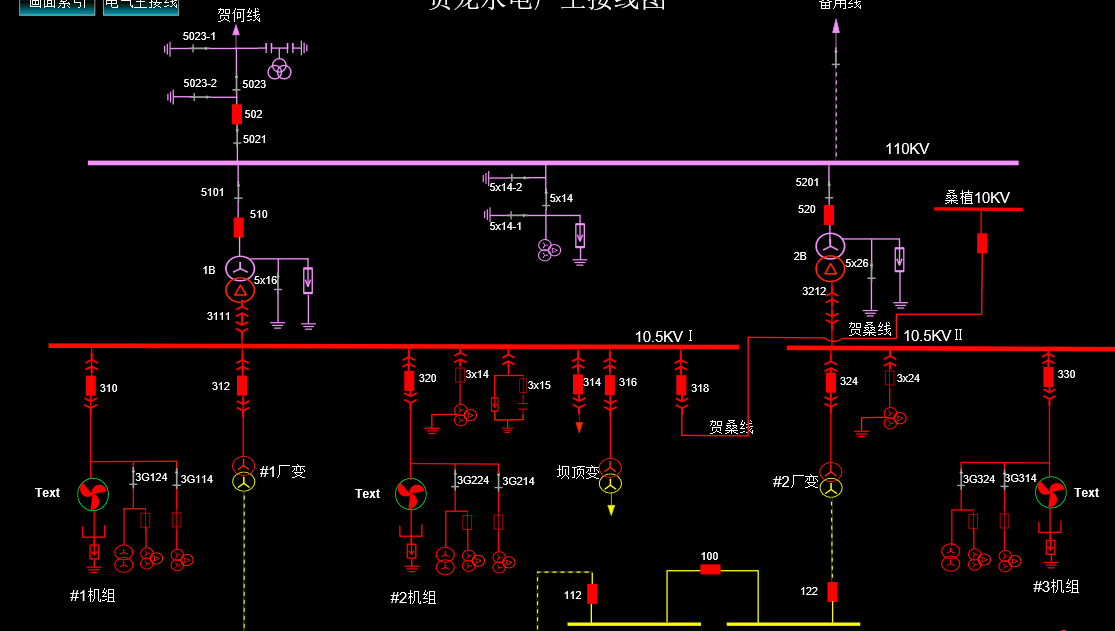
分配图元分组

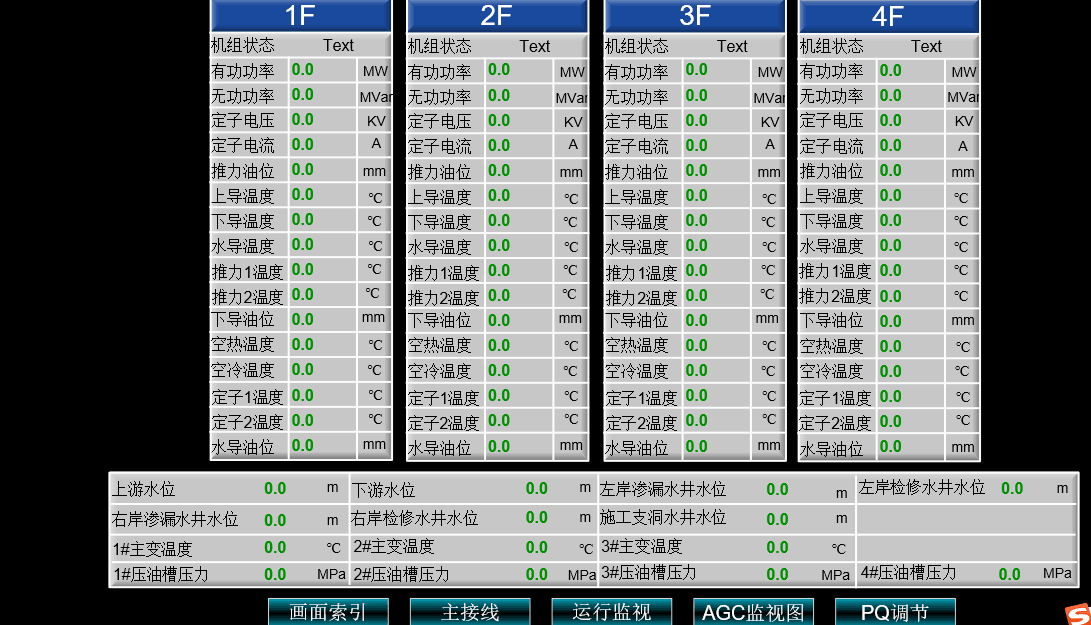


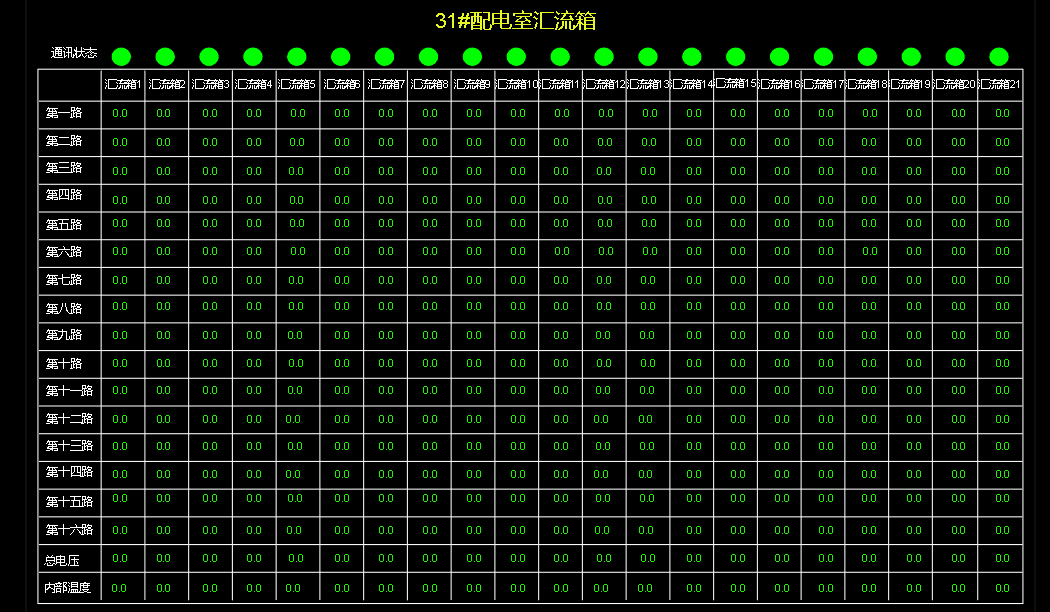
在文件对应分组最后增加

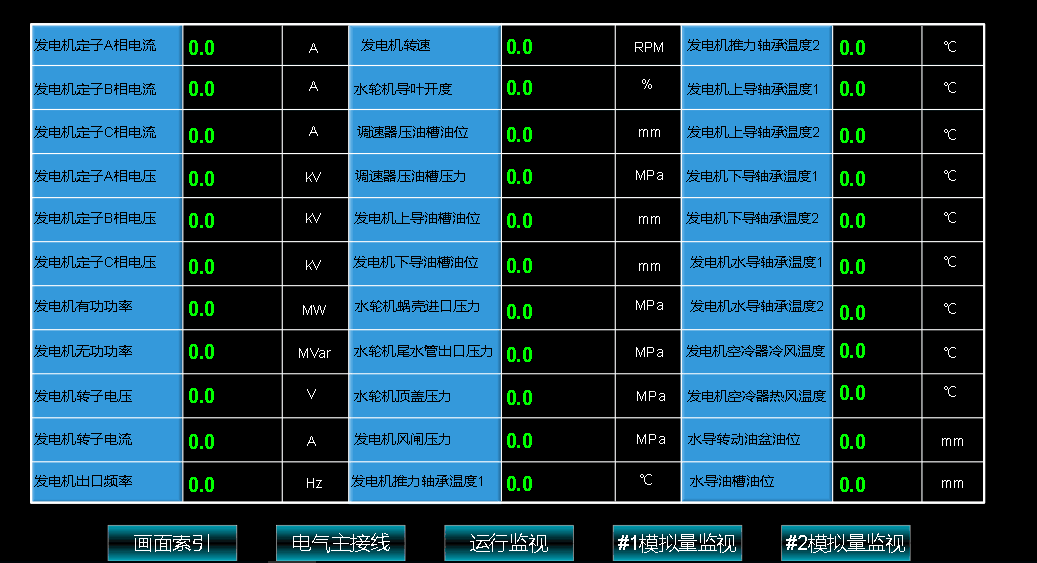


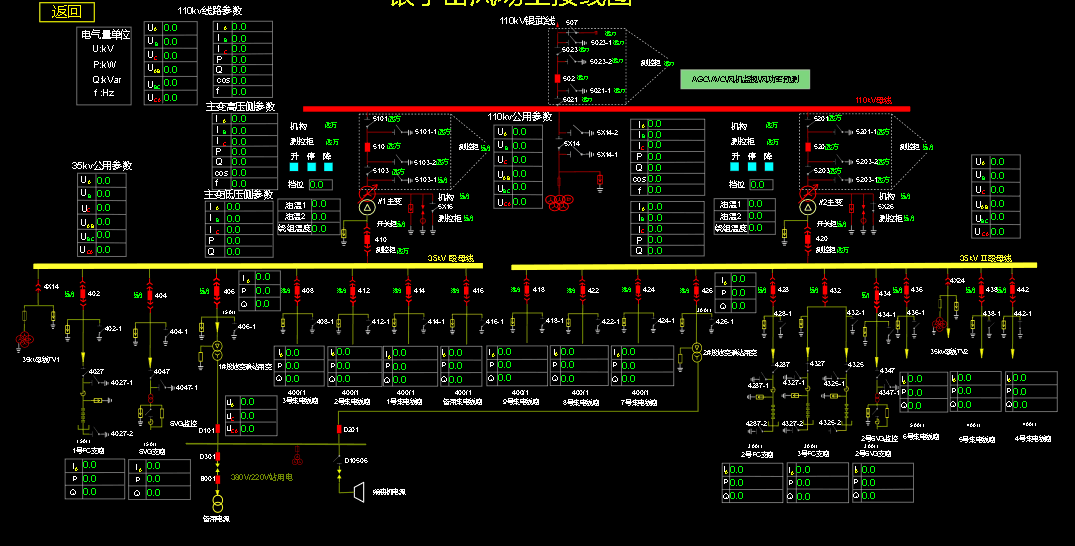
# 典型案例

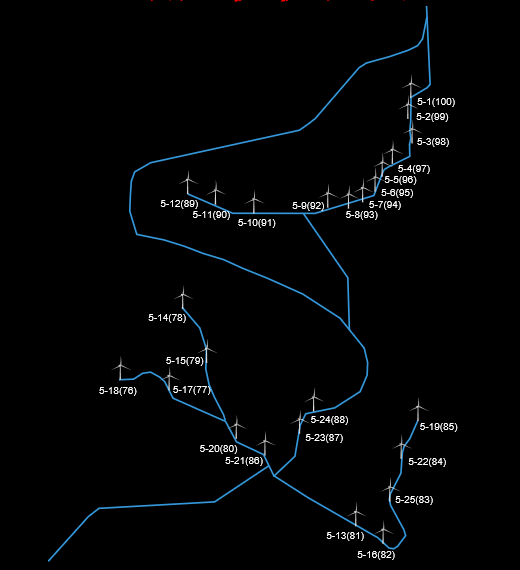


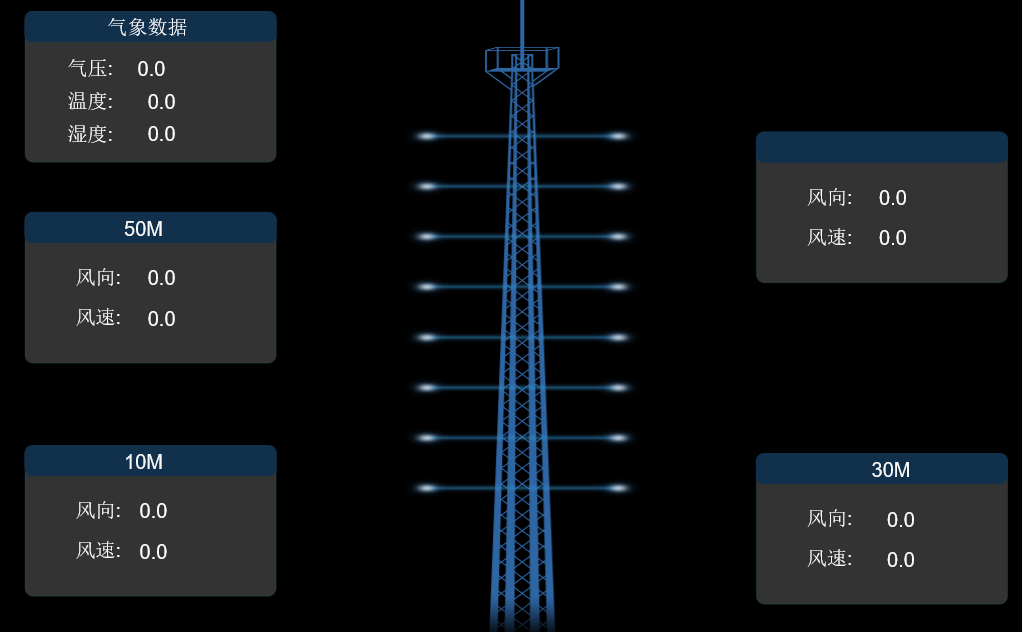


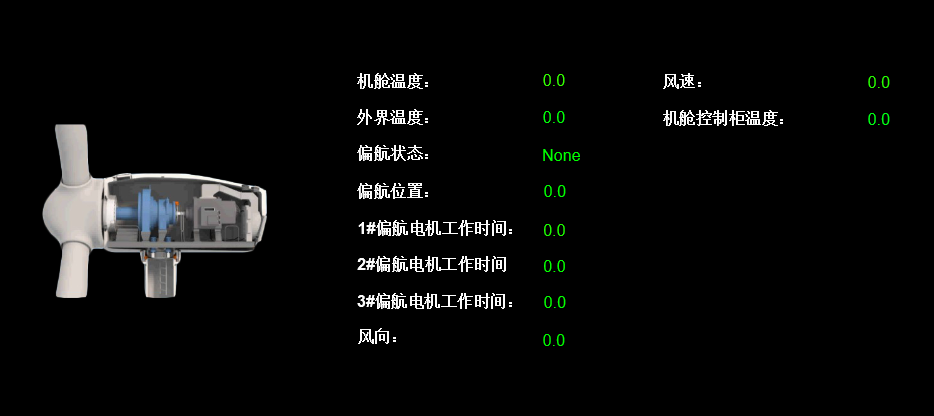






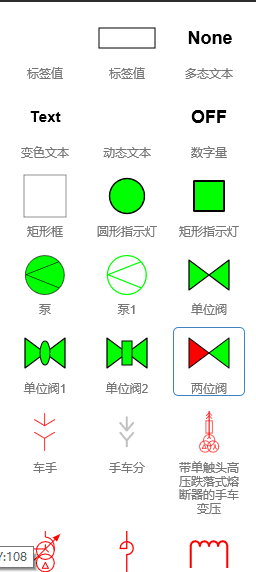
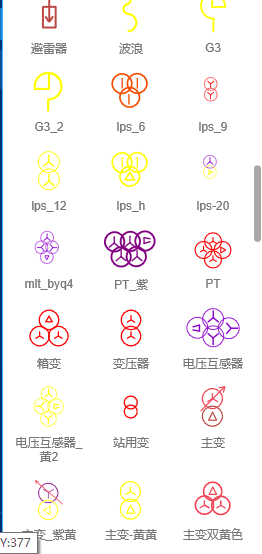
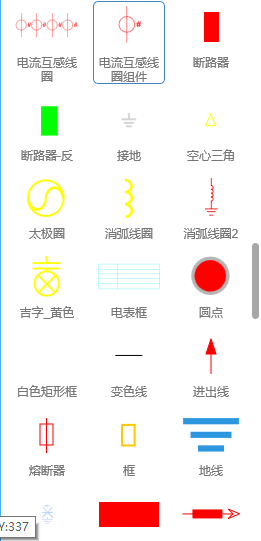
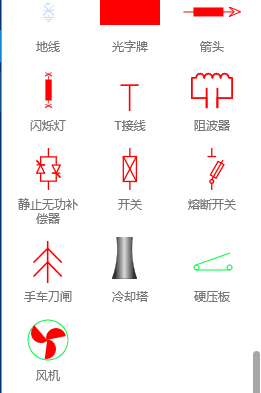
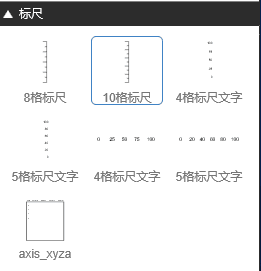


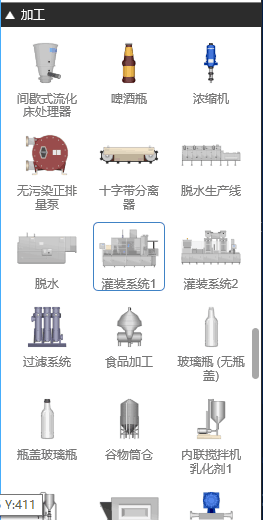
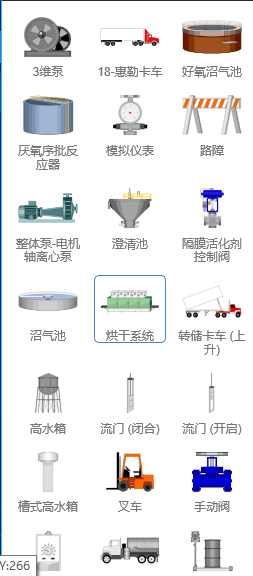




# 图元

系统已集成丰富的图元。

**** **** 