目录

[剧场座位图绘制： 2](#_Toc61109540)

[一、构建基础组件 2](#_Toc61109541)

[1、构建一个剧场的空白区域 2](#_Toc61109542)

[2、构建基础组件：演出舞台 2](#_Toc61109543)

[3、构建基础组件，以座椅为例 3](#_Toc61109544)

[二 .确定组件进入画布的坐标轴 3](#_Toc61109545)

[三 . 画图操作（去重、是否可以放置） 4](#_Toc61109546)

[四．获取选中的区域 5](#_Toc61109547)

[五．删除设施 deleteSelectfacilities（selectFacilitiesList，facilitiesList） 6](#_Toc61109548)

[六、移动组件 6](#_Toc61109549)

[七、复制组件 9](#_Toc61109550)

[票图 11](#_Toc61109551)

[一、选中区域，设置座位的票型 11](#_Toc61109552)

[二、暂存座位的票型数据 13](#_Toc61109553)

[三、保存所有的座位票型 14](#_Toc61109554)

[移动端状态图 16](#_Toc61109555)

[一、选中/取消座位selectSeat（seatId，seatList，selectList） 16](#_Toc61109556)

[二、当前选中座位总价 calculateThePrice（selectList，ticketList） 17](#_Toc61109557)

[三、选中票档后高亮当前对应座位区域/取消选中票档后高亮全场座椅 17](#_Toc61109558)

[四、选座提交 saveOrder（selectList，showid，dramaid） 18](#_Toc61109559)

# **剧场座位图绘制**：

1. 构建基础组件

## 1、构建一个剧场的空白区域

|  |  |  |  |
| --- | --- | --- | --- |
| 参数 | 是否必须 | 说明 | 备注 |
| width | 是 | 场馆长 |  |
| height | 是 | 场馆宽 |  |
| name | 是 | 场馆名称 |  |
| use | 是 | 场馆用途 |  |
| state | 是 | 场馆状态 |  |
| address | 是 | 场馆地址 |  |
| Floor | 否 | 场馆楼层 |  |
| Desc | 否 | 场馆描述 |  |
| Person | 否 | 联系人 |  |
| phone | 否 | 联系电话 |  |
| Rule | 是 | 比例尺 |  |

## 2、构建基础组件：演出舞台

|  |  |  |  |
| --- | --- | --- | --- |
| 参数 | 是否必须 | 说明 | 备注 |
| width | 是 | 舞台长 |  |
| height | 是 | 舞台宽 |  |
| name | 是 | 舞台名称 |  |
| Color | 否 | 舞台颜色 |  |
| state | 是 | 舞台状态 |  |
| Rule | 是 | 比例尺 |  |

## 3、构建基础组件，以座椅为例

|  |  |  |  |
| --- | --- | --- | --- |
| 参数 | 是否必须 | 说明 | 备注 |
| name | 是 | 模型名称 |  |
| img | 是 | 模型图片 |  |
| type | 是 | 组件类型 |  |
| width | 是 | 宽度 |  |
| hight | 是 | 长度 |  |
| id | 是 | 主键 |  |
| state | 是 | 模型状态 |  |
| creatTime | 是 | 创建时间 |  |
| rule | 是 | 比例尺 |  |

二 .确定组件进入画布的坐标轴

enterXY(ModuleComponent){

let mouseFouce = e.pageX // 获取鼠标的进入点

let x= ModuleComponent.x

let y = ModuleComponent.y

处理结果

Return x ,y

}

|  |  |  |  |
| --- | --- | --- | --- |
| 输入参数 | 是否必须 | 说明 | 备注 |
| ModuleComponent | 是 | 组件对象的基础属性 |  |

|  |  |  |
| --- | --- | --- |
| 输出参数 | 说明 | 备注 |
| Module\_X | 组件对象在画布上的X坐标 |  |
| Module\_Y | 组件对象在画布上的Y坐标 |  |

三 . 画图操作（去重、是否可以放置）

DrawModule(ModuleComponent,siteList){

组件的坐标点与画布中已经放置的组件集合的坐标点进行比较，如果重合，则返回 false 否则返回true

当返回true时说明本区域可以放置该组件，则siteList.push(ModuleComponent)

}

|  |  |  |  |
| --- | --- | --- | --- |
| 输入参数 | 是否必须 | 说明 | 备注 |
| ModuleComponent | 是 | 组件对象的基础属性 |  |
| siteList | 是 | 画布中放入的组件集合 |  |

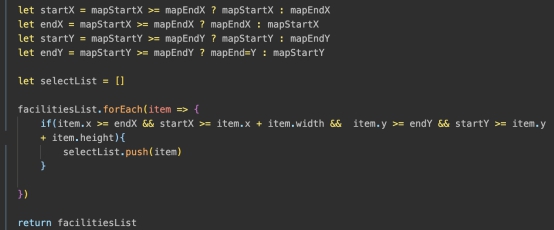
|  |  |  |
| --- | --- | --- |
| 输出参数 | 说明 | 备注 |
| isTrue | 是否可以放置 true可以放置 false不可以 |  |
| siteList | 画布中已经占领的区域集合 |  |

四．获取选中的区域

checkSelectListInZone（startX，mapStartX，mapEndX，mapEndY，facilitiesList）

|  |  |  |  |
| --- | --- | --- | --- |
| 输入参数 | 是否必须 | 说明 | 备注 |
| mapStartX | 是 | 起始X坐标 |  |
| mapStartY | 是 | 起始Y坐标 |  |
| mapEndX | 是 | 结束X坐标 |  |
| mapEndY | 是 | 结束Y坐标 |  |
| facilitiesList | 是 | 设施list |  |

checkSelectListInZone（mapStartX，mapStartY，mapEndX，mapEndY，facilitiesList）{



}

|  |  |  |  |
| --- | --- | --- | --- |
| 输出参数 | 是否必须 | 说明 | 备注 |
| selectList | 是 | 选中设施list |  |

五．删除设施

deleteSelectfacilities（selectFacilitiesList，facilitiesList）

|  |  |  |  |
| --- | --- | --- | --- |
| 输入参数 | 是否必须 | 说明 | 备注 |
| deleteSelectfacilities | 是 | 选中设施列表 |  |
| facilitiesList | 是 | 已有设施列表 |  |

deleteSelectfacilities（selectFacilitiesList，facilitiesList）{



}

|  |  |  |  |
| --- | --- | --- | --- |
| 输出参数 | 是否必须 | 说明 | 备注 |
| facilitiesList | 是 | 全部设施list |  |

六、移动组件

入参type 的值down往下、up往上、left往左、right往右

入参selectComponents 以这个组件为技术

入参venueTargets 舞台对象

入参distance 移动的距离

返回值 changeselectComponents 是selectComponents 改变后的位置(只改变位置)

moveElement(type,selectComponent,venueTarget,distance){

let changeselectComponents=[]

Let isoverlay = false 没有重叠

for(let i=0;i<selectComponents.length;i++){

let selectComponent=selectComponents[i]

Let selectComponentpos=selectComponent.postion

Let {selectComponentposX,selectComponentposY,width,height}=selectComponentpos

根据选择的selectComponentposX,selectComponentposY,width,height 计算出来每个组件的左上、左下、右上、右下的坐标

for(let m=0;m<venueTargets.length;m++){

let venueTarget=venueTargets[i]

Let venueTargetpos=venueTarget.postion

Let {venueTargetposX,venueTargetposY,width,height}=venueTarget

根据选择的venueTargetposX,venueTargetposY,width,height 计算出来场馆中每个组件的左上、左下、右上、右下的坐标

比较selectComponentpos和venueTargetpos是否有重叠

要是有重叠，不能移动 isoverlay = ture

否者可以移动 isoverlay = false

}

If(isoverlay ){

提示不能移动

Return

}

//开始移动

for(let i=0;i<selectComponents.length;i++){

let selectComponent=selectComponents[i]

Let selectComponentpos=selectComponent.postion

Let {selectComponentposX,selectComponentposY,width,height}=selectComponentpos

If(type==”left” || type==”right” ){

If(type==”left” ){

distance=-distance

}else{

distance=distance

}

selectComponentpos.selectComponentposX=selectComponentposX+distance

}

Else {

If(type==”down” ){

distance=-distance

}else{

distance=distance

}

selectComponentpos. selectComponentposY=selectComponentposY+distance

}

changeselectComponents.push(selectComponent)

}

}

return changeselectComponents

}

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 输入参数 | 值 | 是否必须 | 说明 | 备注 |
| type | down、up、left、right | true | 移动组件的方向 |  |
| selectComponents | array | true | 将要移动组件 |  |
| venueTargets | array | true | 场馆对象 |  |
| distance | 比如 1px | true | 选择组件移动的距离 |  |

|  |  |  |
| --- | --- | --- |
| 输出参数 | 值 | 说明 |
| changeselectComponents | 和selectComponents的数据一样 |  |

七、复制组件

/\*\*

\* @param {Object} selectComponents 选中的组件

\* @param {Object} type 复制的方向

\* @param {Object} count 复制的个数

\* @param {Object} venueTargets 场馆的对象

\* distance 移动的距离

\*/

copycomponents(selectComponents,type,count,venueTargets,distance){

let changecopycomponents=[]

let filterchangecopycomponents=[]

/\*\*

\* 遍历选中组件第一个的位置

\* 和最后一个位置

\* 计算出来第一个和最后一个中心位置的距离

\*/

let selectdistance=0

let firsteledis={

x:0,

y:0

}

let lasteledis={

x:0,

y:0

}

for(let i=0;i<selectComponents.length;i++){

遍历获取firsteledis和lasteledis位置

}

selectdistance就是firsteledis和lasteledis位置的距离

//根据type移动x或者y值

for(let i=0;i<selectComponents.length;i++){

let selectComponent=selectComponents[i]

根据type和selectdistance计算每个选中组件的坐标 changecopycomponents.push(selectComponent)

}

for(let k=0;k<selectComponents.length;k++){

let flag=false

//在这里获取组件的面积的边界的坐标

for(let i=0;i<venueTargets.length;i++){

//在场馆里面画布中的组件获取组件的面积的边界的坐标

let venueTarget=venueTargets[i]

//在这里比较选中的组件和场馆的组件是否重叠

} filterchangecopycomponents.push(selectComponent)

}

return filterchangecopycomponents

}

|  |  |  |  |
| --- | --- | --- | --- |
| 输入参数 | 值 | 是否必须 | 说明 |
| selectComponents | [] | 必须 | 选中的组件 |
| type | Down、up、left、right | 必须 | 复制的方向 |
| count | Number | 必须 | 复制的个数 |
| venueTargets | [] | 必须 | 场馆 |

|  |  |  |
| --- | --- | --- |
| 输出参数 | 值 | 说明 |
| filterchangecopycomponents | selectComponents 一样的数据 |  |

# 票图

## 一、选中区域。设置座位的票型

setectArea( mousestartpostion,mouseendpostion,seatList ){

Let selctseatelist=[ ]

Let {mousestartpostionx,mousestartpostiony}=mousestartpostion

Let {mouseendpostionx,mouseendpostiony}=mouseendpostion

获取到鼠标的长方形的边界左边

Let rectangle={

leftTop:10,

leftBottom:10,

rightTop:10,

rightBottom:100

}

For(let i=0;i<seatList .length;i++ ){

Let seat=seatList[i]

Let isoverboundary=false //是否超出边界

获取到座位的边界区域

Let seatboundary={

leftTop:10,

leftBottom:10,

rightTop:10,

rightBottom:100

}

比较seatboundary是否在rectangle内

如果在鼠标选择的区域内，isoverboundary=true

//把座位放在结果里面

If(isoverboundary){

Selctseatelist.Push(

seat

)

}

}

Return

}

|  |  |  |  |
| --- | --- | --- | --- |
| 参数 | 值 | 是否必须 | 说明 |
| mousestartpostion | {x:10,y:20} | 必须 | 鼠标按下的开始位置 |
| mouseendpostion | {x:20,y:40} | 必须 | 鼠标按下的结束位置 |
| seatList | [ ] | 必须 | 所有的座位 |

|  |  |  |  |
| --- | --- | --- | --- |
| 结果 | 值 |  | 说明 |
| Selctseatelist | [] |  | 选择座位的数据 |

## 二、暂存座位的票型数据

StoreSeatTicket( Selctseatelist, tempTicketList,ticketobj){

Let ticketList=[]

Let Selctseatelists=[]

Let SelctseateTickets=[]

Selctseatelists=Selctseatelist.concat(tempTicketList)

Selctseatelists.forEach(v=>{

SelctseateTickets.push({

Row:ticketobj.row,

Code:ticketobj.Code,

Price:ticketobj.Price,

Area:ticketobj.Area,

States:ticketobj.States,

})

})

Return SelctseateTickets

}

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 参数 | 值 | 必须 | 说明 |  |
| Selctseatelist.Row | 1 | 必须 | 第几排 |  |
| Selctseatelist.Code | 2 | 必须 | 几号 |  |
| Selctseatelist.Price | 10 | 必须 | 价格 |  |
| Selctseatelist.Area | 100 | 必须 | 票档 |  |
| Selctseatelist.States | number | 必须 | 1是可以售卖2不可售卖 |  |
| tempTicketList | [] | 必须 | 暂存的选中座位 |  |

返回值

|  |  |  |  |
| --- | --- | --- | --- |
| 参数 | 值 |  | 说明 |
| SelctseateTickets | [] |  |  |

## 三、保存所有的座位票型

saveSeatTicket(SelctseateTickets,totalCostPrice){

Let costPrice=0

For(let i=0;i<Selctseatelists.length;i++){

Let price=Selctseatelists[i].price

costPrice+=price

}

If(costPrice>totalCostPrice){

提示超过总的成本

}

调用后台接口

}

入参

|  |  |  |  |
| --- | --- | --- | --- |
| 参数 | 值 | 必须 | 说明 |
| Selctseatelists.code | [] | 必须 | 几号 |
| Selctseatelists.row | 1 | 必须 | 几排 |
| Selctseatelist.Price | 10 | 必须 | 价格 |
| Selctseatelist.Area | 100 | 必须 | 票档 |
| Selctseatelis.States | 1 | 必须 | 1售卖 2不可以售卖 |

返回结果

|  |  |  |  |
| --- | --- | --- | --- |
| 返回值 |  |  |  |
| State | true |  |  |

# 移动端状态图

一、选中/取消座位selectSeat（seatId，seatList，selectList）

|  |  |  |  |
| --- | --- | --- | --- |
| 输入参数 | 是否必须 | 说明 | 备注 |
| seatId | 是 | 选中座位id |  |
| seatList | 是 | 座位数组 |  |
| selectList | 是 | 已选座位数组 |  |

selectSeat（seatId，seatList，selectList）{



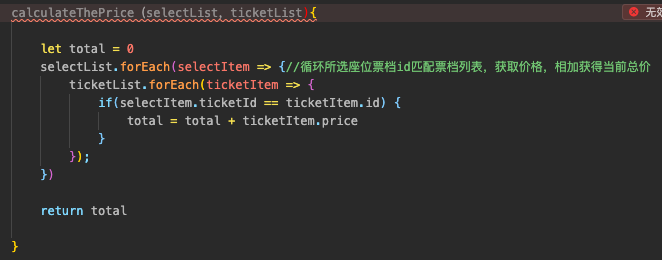
}

|  |  |  |  |
| --- | --- | --- | --- |
| 输出参数 | 是否必须 | 说明 | 备注 |
| selectList | 是 | 选中座位list |  |

二、当前选中座位总价 calculateThePrice（selectList，ticketList）

|  |  |  |  |
| --- | --- | --- | --- |
| 输入参数 | 是否必须 | 说明 | 备注 |
| selectList | 是 | 选中设施列表 |  |
| ticketList | 是 | 已有设施列表 |  |

calculateThePrice（selectList，ticketList）{



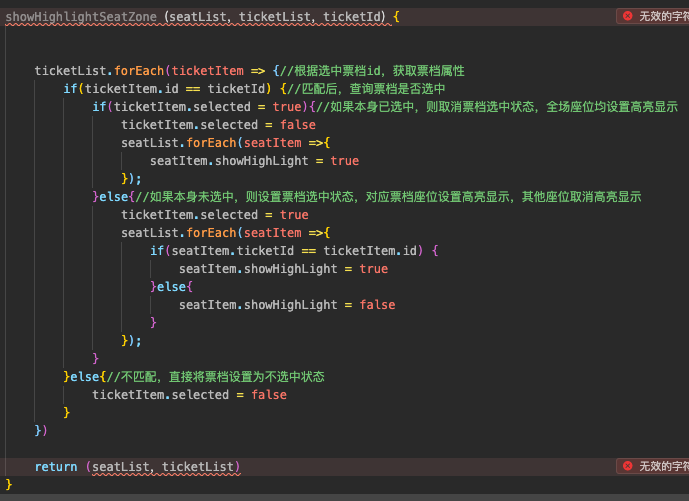
}

|  |  |  |  |
| --- | --- | --- | --- |
| 输出参数 | 是否必须 | 说明 | 备注 |
| totalPrice | 是 | 当前已选座位价格 |  |

三、选中票档后高亮当前对应座位区域/取消选中票档后高亮全场座椅 showHighlightSeatZone（seatList，ticketList，ticketId）

|  |  |  |  |
| --- | --- | --- | --- |
| 输入参数 | 是否必须 | 说明 | 备注 |
| ticketId | 是 | 选中票档id |  |
| seatList | 是 | 座位数组 |  |
| ticketList | 是 | 票档list |  |

showHighlightSeatZone（seatList，ticketList，ticketId）{



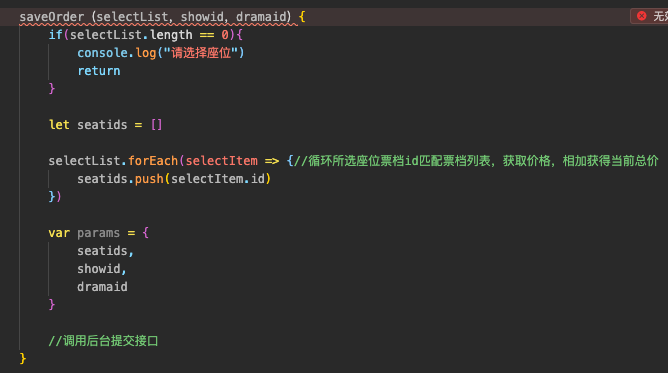
}

|  |  |  |  |
| --- | --- | --- | --- |
| 输出参数 | 是否必须 | 说明 | 备注 |
| ticketList | 是 | 票档list |  |
| seatList | 是 | 座位数组 |  |

四、选座提交 saveOrder（selectList，showid，dramaid）

|  |  |  |  |
| --- | --- | --- | --- |
| 输入参数 | 是否必须 | 说明 | 备注 |
| selectList | 是 | 选中座位list |  |
| showid | 是 | 场次id |  |
| dramaid | 是 | 剧目id |  |

saveOrder（selectList，showid，dramaid）{



}