1. 本工程已经配置好了关于运行后端和网页前端的webstorm工程设置。单击webstorm运行会自动打开后端并用chrome打开127.0.0.1:5555。或者:手动开启服务器后端:node main.js然后浏览器访问:127.0.0.1:5555
2. 服务器在darwin系统和win系统上会自动屏蔽EOS相关的功能。
3. 访问地址127.0.0.1:5555会自动跳转到127.0.0.1:5555/mainPage.html, mainPage.html即为电脑上的主要页面.

WebSocket 地址: 'ws://127.0.0.1:5556'

Websocket指令:所有指令都按照JSON格式发送,服务器有关websocket的处理代码都在backend.js文件的CreateServer函数当中.

1. 获取目前后端仿真器的所有数据:

**let** command = {};  
command['type'] = 'require';  
command['content'] = '';  
websocket.send(JSON.stringify(command));

服务器返回的数据格式如下:

{

"type": "require",

"content": "",

"result": {

"market": {

"userPriceMap": {}, //这个不用管

"userPriceMap\_JsonUse": { //这个里面存储每个房屋主人的报价

"home11": "90.31",

"home12": "90.17",

"home13": "90.06",

"home14": "90.18",

"home15": "90.96",

"home21": "90.51",

"home22": "90.13",

"home23": "90.89",

"home24": "90.91",

"home25": "90.52"

}

},

"community": {

"idHomeMap": {}, //这个不用管

"homes": { //总共10个房屋的记录,这里只显示一个

"home11": { //用户ID

"state": true, //用户电源状态

"identification": {

"name": "User0", //身份记录,Name是用户名字,与id不同

"id": "home11"

},

"meter": { //电表记录

"id": "MeterId0",

"reading": 5,

"previousReading": 4

},

"cleanEnergy": 5, //用户剩余清洁能源

"fossilEnergy": 320, //用户剩余化石能源

"consumptionSpeed": 1, //消耗能源的速度,优先使用清洁能源

"usedEnergyType": "clean" //目前使用的能源类型

},

"home12":…….

},

"battery": [

{

"identification": {

"name": "battery",

"id": "battery0"

},

"capacity": 5000, //电池容量

"energy": 2659, //目前储能

"ratio": 0.5318, //电量百分比

"chargeLowThres": 0.3, //百分比低于这么多,从电网充电

"chargeHighThres": 0.5, //百分比高于这么多,停止从电网充电

"isInCriticalState": false //目前是否从电网充电,false为没有

}

],

"generator": [ //总共3台发电机,太阳能,风能和电网

{

"identification": { //发电机ID

"name": "solar",

"id": "solar0"

},

"meter": { //发电机的电表

"id": "solarMeter0",

"reading": 600,

"previousReading": 500

},

"generationSpeed": 100 //发电速度

},

{

"identification": {

"name": "wind",

"id": "wind0"

},

"meter": {

"id": "windMeter0",

"reading": 250,

"previousReading": 200

},

"generationSpeed": 50

}

],

"grid": [

{

"identification": {

"name": "grid",

"id": "grid0"

},

"meter": {

"id": "gridMeter0",

"reading": 2000,

"previousReading": 1600

},

"generationSpeed": 400

}

]

},

"secondPerTick": 3 //仿真器每3秒会步进一次,即用户消耗电,发电机发电进电池,检测电池是否需要从电网充电以及市场的拍卖等.

}

}

1. 充值化石能源: type指定目前指令的作用,content为指令内容,对于充值化石能源而言,content格式为”{充值客户房屋的ID} {充值量}”

**let** command = {};

command['type'] = 'chargeFossil';  
command['content'] = 'home11 10';  
websocket.send(JSON.stringify(command));

服务器返回数据:

{

"type": "chargeFossil",

"content": "home11 10",

"result": "ok" //这个字段用来表示是否成功执行,如果为false,请检查指令格式和充值对象ID是否存在.

}

1. 更改清洁能源的报价: type指定目前指令的作用,content为指令内容,对于当前指令而言,content格式为”{客户房屋的ID} {报价}”

在每个周期当中.市场会根据当前报价,选出最高价进行交易.所以如果是测试需要,请把价格控制在91以上可以确保每次交易都会是这个用户.(用户home2x的初始报价是9~10之间的一个随机数, 用户home1x的报价是9~9.5之间的随机数,用户home1x的报价可以通过电脑前端修改,home2x不可通过前端修改)

**let** command = {};

command['type'] = 'changePrice';  
command['content'] = 'home11 91';  
dataSocket.send(JSON.stringify(command));

服务器返回数据:

{

"type": "changePrice",

"content": "home11 91",

"result": "ok"

}