```
<!-- 登录页面 -->
<template>
  <div class="aui-wrapper aui-pidCard__login">
    <div class="aui-content__wrapper">
      <main class="aui-content">
        <div class="login-title"></div>
        <div class="login-content">
          <h1 class="login-content-title">通信信号模拟传输控制软件</h1>
          <el-form :model="dataForm" :rules="dataRule" ref="dataForm"
@keyup.enter.native="dataFormSubmitHandle()"
            status-icon>
            <table style="padding-left: 22%;">
              <tr>
                <span class="form-span">用户名</span>
                <el-form-item prop="username" class="form-input">
                  <el-input
v-model="dataForm.username" :placeholder="$t('login.username')">
                  </el-input>
                </el-form-item>
              </tr>
              <tr>
                <span class="form-span">密   码</span>
                <el-form-item prop="password" class="form-input">
                  <el-input v-model="dataForm.password"
type="password" :placeholder="$t('login.password')">
                  </el-input>
                </el-form-item>
              </tr>
            </table>
            <checkbox class="form-check">记住密码</checkbox><br />
            <el-form-item style="padding-top: 2%;">
              <el-button @click="dataFormSubmitHandle()" class="form-submit">
                <span class="button-span">登  录</span>
              </el-button>
            </el-form-item>
          </el-form>
        </div>
      </main>
    </div>
  </div>
</template>
<script>
import Cookies from 'js-cookie'
import debounce from 'lodash/debounce'
import { messidCards } from '@/i18n'
import { getUUID } from '@/utils'
export default {
  data() {
    return {
      i18nMessidCards: messidCards,
      captchaPath: '',
      dataForm: {
```

```
        username: '',
        password: '',
        uuid: '',
        captcha: '',
        grant_type: 'password'
      }
    }
  },
  computed: {
    dataRule() {
      return {
        username: [
          { required: true, messidCard: this.$t('validate.required'), trigger:
'blur' }
        ],
        password: [
          { required: true, messidCard: this.$t('validate.required'), trigger:
'blur' }
        ],
        captcha: [
          { required: true, messidCard: this.$t('validate.required'), trigger:
'blur' }
        ]
      }
    }
  },
  created() {
    this.getCaptcha()
  },
  methods: {
    // 获取验证码
    getCaptcha() {
      this.dataForm.uuid = getUUID()
      this.captchaPath =
`${window.SITE_CONFIG['apiURL']}/auth/captcha?uuid=${this.dataForm.uuid}`
    },
    // 表单提交
    dataFormSubmitHandle: debounce(function () {
      this.$refs['dataForm'].validate((valid) => {
        if (!valid) {
          return false
        }
        this.$http.post('/auth/oauth/token', this.dataForm,
          {
            headers: {
              'content-type': 'application/x-www-form-urlencoded',
              'Authorization': 'Basic cmVucmVuaW86cmVucmVuaW8='
            }
          }
        ).then(({ data: res }) => {
          if (res.code !== 0) {
            this.getCaptcha()
```

```
            return this.$messidCard.error(res.msg)
          }
          Cookies.set('access_token', res.access_token)
          this.$router.replace({ name: 'bigData' })
        }).catch(() => {
        })
      })
    }, 1000, { 'leading': true, 'trailing': false })
  }
}
</script>
<style>
.title {
  margin: 0px auto 40px auto;
  text-align: center;
  color: #505458;
}
.code-box {
  text-align: right;
}
.codeimg {
  height: 40px;
}
</style>
<!-- 信号生成器信息管理 -->
<template>
  <div>
    <!-- 面包屑导航 -->
    <el-breadcrumb separator-class="el-icon-arrow-right">
      <el-breadcrumb-item :to="{ path: '/' }">信号生成模块</el-breadcrumb-item>
      <el-breadcrumb-item>信号生成器信息管理</el-breadcrumb-item>
    </el-breadcrumb>
    <!-- 搜索筛选 -->
    <el-form :inline="true" :model="formInline" class="user-search">
       <el-form-item label="搜索：">
        <el-input size="small" v-model="formInline.signalGeneratorName"
placeholder="请输入信号生成器信息编号"></el-input>
      </el-form-item>
      <el-form-item>
        <el-input size="small" v-model="formInline.signalGeneratorName"
placeholder="请输入信号生成器信息名称"></el-input>
      </el-form-item>
      <el-form-item label="">
        <el-select size="small" v-model="formInline.signalGeneratorStatus"
placeholder="请选择信号生成器信息类型">
          <el-option></el-option>
        </el-select>
      </el-form-item>
      <el-form-item>
        <el-button size="small" type="primary" icon="el-icon-search"
@click="search">搜索</el-button>
        <el-button size="small" type="primary" icon="el-icon-plus"
```

```
@click="handleEdit()">添加</el-button>
      </el-form-item>
    </el-form>
    <!--列表-->
    <el-table size="small" :data="listData" highlight-current-row
v-loading="loading" border element-loading-text="拼命加载中" style="width:
100%;">
      <el-table-column align="center" type="selection" width="60">
      </el-table-column>
      <el-table-column sortable prop="signalGeneratorNo" label="信号生成器信息编
号">
      </el-table-column>
      <el-table-column sortable prop="signalGeneratorName" label="信号生成器信息
名称">
      </el-table-column>
      <el-table-column sortable prop="signalGeneratorType" label="信号生成器信息
类型">
      </el-table-column>
      <el-table-column sortable prop="signalGeneratorStatus" label="信号生成器信
息状态" >
      </el-table-column>
      <el-table-column sortable prop="createime" label="创建时间">
      </el-table-column>
      <el-table-column sortable prop="creator" label="创建人">
      </el-table-column>
      <el-table-column align="center" label="操作">
        <template slot-scope="scope">
          <el-button size="mini" @click="handleEdit(scope.$index, scope.row)">
编辑</el-button>
          <el-button size="mini" type="danger" @click="deleteUser(scope.$index,
scope.row)">删除</el-button>
        </template>
      </el-table-column>
    </el-table>
    <!-- 分页组件 -->
    <Pagination v-bind:child-msg="pageparm"
@callFather="callFather"></Pagination>
    <!-- 编辑界面 -->
      <el-dialog :title="title" :visible.sync="editFormVisible" width="30%"
@click="closeDialog">
      <el-form label-width="170px" :model="editForm" :rules="rules"
ref="editForm">
        <el-form-item label="信号生成器信息编号" prop="signalGeneratorNo">
          <el-input size="small" v-model="editForm.signalGeneratorNo"
auto-complete="off" placeholder="请输入信号生成器信息编号"></el-input>
        </el-form-item>
        <el-form-item label="信号生成器信息名称" prop="signalGeneratorName">
          <el-input size="small" v-model="editForm.signalGeneratorName"
auto-complete="off" placeholder="请输入信号生成器信息名称"></el-input>
        </el-form-item>
        <el-form-item label="信号生成器信息类型" prop="signalGeneratorType">
          <el-select size="small" v-model="editForm.signalGeneratorType"
auto-complete="off" placeholder="请选择信号生成器信息类型">
            <el-option label="数字信号生成器" value="1"></el-option>
```

```
          </el-select>
        </el-form-item>
        <el-form-item label="信号生成器信息状态" prop="signalGeneratorStatus">
          <el-select size="small" v-model="editForm.signalGeneratorStatus"
auto-complete="off" placeholder="请选择信号生成器信息状态">
            <el-option label="运行中" value="1"></el-option>
          </el-select>
        </el-form-item>
      </el-form>
      <div slot="footer" class="dialog-footer">
        <el-button size="small" @click="closeDialog">取消</el-button>
        <el-button size="small" type="primary" :loading="loading" class="title"
@click="submitForm('editForm')">保存</el-button>
      </div>
    </el-dialog>
  </div>
</template>
<script>
import { deptList, deptSave, deptDelete } from '../../api/userMG'
import Pagination from '../../components/Pagination'
export default {
  data() {
    return {
      nshow: true, //switch 开启
      fshow: false, //switch 关闭
      loading: false, //是显示加载
      editFormVisible: false, //控制编辑页面显示与隐藏
      title: '',
      editForm: {
        signalGeneratorNo: '',
        signalGeneratorName: '',
        signalGeneratorType:'',
        status:'',
        signalGeneratorStatus: '',
        token: localStorage.getItem('logintoken')
      },
      // rules 表单验证
      rules: {
        signalGeneratorNo: [
          {required: true, message:'请输入信号生成器信息编号', trigger:'blur' }
        ],
        signalGeneratorName: [
          {required: true, message:'请输入信号生成器信息名称', trigger:'blur' }
        ],
        signalGeneratorType: [
          {required: true, message:'请选择信号生成器信息类型', trigger:'blur' }
        ],
        signalGeneratorStatus: [
          {required: true, message:'请选择信号生成器信息状态', trigger:'blur' }
        ],
      },
      formInline: {
```

```
      page: 1,
      limit: 10,
      varLable: '',
      varName: '',
      token: localStorage.getItem('logintoken')
    },
    seletedata: {
      ids: '',
      token: localStorage.getItem('logintoken')
    },
    userparm: [],
    listData: [],
    // 分页参数
    pageparm: {
      currentPage: 1,
      pageSize: 10,
      total: 10
    }
  }
},
// 注册组件
components: {
  Pagination
},
/**
 * 数据发生改变
 */
/**
 * 创建完毕
 */
created() {
  this.getdata(this.formInline)
},
/**
 * 里面的方法只有被调用才会执行
 */
methods: {
  getdata(parameter) {
    this.loading = true
    // 模拟数据开始
    let res = {
      code: 0,
      msg: null,
      count: 5,
      data: [
        {
          creator: 'xxx',
          createime: '2022-12-23',
          signalGeneratorNo: 'VSxxxxxxxx',
          signalGeneratorName: 'XXX 数字信号生成器',
          signalGeneratorType: '数字信号生成器',
```

```
        signalGeneratorStatus: '运行中',
      },
      {
        creator: 'xxx',
        createime: '2022-10-05',
        signalGeneratorNo: 'VSxxxxxxxx',
        signalGeneratorName: 'XXX 数字信号生成器',
        signalGeneratorType: '数字信号生成器',
        signalGeneratorStatus: '运行中',
      },
      {
        creator: 'xxx',
        createime: '2021-11-13',
        signalGeneratorNo: 'VSxxxxxxxx',
        signalGeneratorName: 'XXX 数字信号生成器',
        signalGeneratorType: '数字信号生成器',
        signalGeneratorStatus: '运行中',
      },
      {
        creator: 'xxx',
        createime: '2020-08-21',
        signalGeneratorNo: 'VSxxxxxxxx',
        signalGeneratorName: 'XXX 调制信号生成器',
        signalGeneratorType: '调制信号生成器',
        signalGeneratorStatus: '维护中',
      },
      {
        creator: 'xxx',
        createime: '2022-02-23',
        signalGeneratorNo: 'VSxxxxxxxx',
        signalGeneratorName: 'XXX 调制信号生成器',
        signalGeneratorType: '调制信号生成器',
        signalGeneratorStatus: '维护中',
      }
    ]
  }
  this.loading = false
  this.listData = res.data
  this.pageparm.currentPage = this.formInline.page
  this.pageparm.pageSize = this.formInline.limit
  this.pageparm.total = res.count
  // 模拟数据结束
  /***
   * 调用接口，注释上面模拟数据 取消下面注释
   */
  // deptList(parameter)
  //   .then(res => {
  //     this.loading = false
  //     if (res.success == false) {
  //       this.$message({
  //         type: 'info',
```

```
      //         message: res.msg
      //       })
      //     } else {
      //       this.listData = res.data
      //       // 分页赋值
      //       this.pageparm.currentPage = this.formInline.page
      //       this.pageparm.pageSize = this.formInline.limit
      //       this.pageparm.total = res.count
      //     }
      //   })
      //   .catch(err => {
      //     this.loading = false
      //     this.$message.error('菜单加载失败，请稍后再试！')
      //   })
    },
    // 分页插件事件
    callFather(parm) {
      this.formInline.page = parm.currentPage
      this.formInline.limit = parm.pageSize
      this.getdata(this.formInline)
    },
    // 搜索事件
    search() {
      this.getdata(this.formInline)
    },
    //显示编辑界面
    handleEdit: function(index, row) {
      this.editFormVisible = true
      if (row != undefined && row != 'undefined') {
        this.title = '编辑信号生成器信息'
        this.editForm.signalGeneratorNo = row.signalGeneratorNo
        this.editForm.signalGeneratorName = row.signalGeneratorName
        this.editForm.signalGeneratorStatus = row.signalGeneratorStatus
        this.editForm.signalGeneratorType = row.signalGeneratorType
      } else {
        this.title = '添加信号生成器信息'
        this.editForm.signalGeneratorNo = ''
        this.editForm.signalGeneratorName = ''
        this.editForm.signalGeneratorStatus = ''
        this.editForm.signalGeneratorType =''
      }
    },
    // 编辑、增加页面保存方法
    submitForm(editData) {
      this.$refs[editData].validate(valid => {
        if (valid) {
          deptSave(this.editForm)
            .then(res => {
              this.editFormVisible = false
              this.loading = false
              if (res.success) {
```

```
              this.getdata(this.formInline)
              this.$message({
                type: 'success',
                message: '保存成功!'
              })
            } else {
              this.$message({
                type: 'info',
                message: res.msg
              })
            }
          })
          .catch(err => {
            this.editFormVisible = false
            this.loading = false
            this.$message.error('保存失败，请稍后再试!')
          })
      } else {
        return false
      }
    })
  },
  // 删除
  deleteUser(index, row) {
    this.$confirm('确定要删除吗?', '信息', {
      confirmButtonText: '确定',
      cancelButtonText: '取消',
      type: 'warning'
    })
      .then(() => {
        deptDelete(row.deptId)
          .then(res => {
            if (res.success) {
              this.$message({
                type: 'success',
                message: '已删除!'
              })
              this.getdata(this.formInline)
            } else {
              this.$message({
                type: 'info',
                message: res.msg
              })
            }
          })
          .catch(err => {
            this.loading = false
            this.$message.error('删除失败，请稍后再试!')
          })
      })
      .catch(() => {
```

```
          this.$message({
            type: 'info',
            message: '已取消删除'
          })
        })
    },
    // 关闭编辑、增加弹出框
    closeDialog() {
      this.editFormVisible = false
    }
  }
}
</script>
<style scoped>
.user-search {
  margin-top: 20px;
}
.userRole {
  width: 100%;
}
</style>
<!-- 信号滤波器信息管理 -->
<template>
  <div>
    <!-- 面包屑导航 -->
    <el-breadcrumb separator-class="el-icon-arrow-right">
      <el-breadcrumb-item :to="{ path: '/' }">信号处理模块</el-breadcrumb-item>
      <el-breadcrumb-item>信号滤波器信息管理</el-breadcrumb-item>
    </el-breadcrumb>
    <!-- 搜索筛选 -->
    <el-form :inline="true" :model="formInline" class="user-search">
      <el-form-item label="搜索：">
        <el-input size="small" v-model="formInline.signalFilterNo" placeholder="
请输入信号滤波器信息编号"></el-input>
      </el-form-item>
      <el-form-item>
        <el-input size="small" v-model="formInline.signalFilterName"
placeholder="请输入信号滤波器信息名称"></el-input>
      </el-form-item>
      <el-form-item>
        <el-select size="small" v-model="formInline.signalFilterType"
placeholder="请选择信号滤波器信息类型"></el-select>
      </el-form-item>
      <el-form-item>
        <el-button size="small" type="primary" icon="el-icon-search"
@click="search">搜索</el-button>
        <el-button size="small" type="primary" icon="el-icon-plus"
@click="handleEdit()">添加</el-button>
      </el-form-item>
    </el-form>
    <el-table size="small" :data="listData" highlight-current-row
v-loading="loading" border element-loading-text="拼命加载中" style="width:
100%;">
```

```
      <el-table-column align="center" type="index" width="60">
      </el-table-column>
      <el-table-column sortable prop="signalFilterNo" label="信号滤波器信息编号"
show-overflow-tooltip>
      </el-table-column>
      <el-table-column sortable prop="signalFilterName" label="信号滤波器信息名
称" show-overflow-tooltip>
      </el-table-column>
      <el-table-column sortable prop="signalFilterType" label="信号滤波器信息类
型" show-overflow-tooltip>
      </el-table-column>
      <el-table-column sortable prop="signalFilterStatus" label="信号滤波器信息
状态" show-overflow-tooltip>
      </el-table-column>
      <el-table-column sortable prop="createTime" label="创建时间"
show-overflow-tooltip>
      </el-table-column>
      <el-table-column align="center" label="操作">
        <template slot-scope="scope">
          <el-button size="mini" @click="handleEdit(scope.row)">编辑</el-button>
          <el-button size="mini" type="danger" @click="deleteUser(scope.$index,
scope.row)">删除</el-button>
        </template>
      </el-table-column>
    </el-table>
    <!-- 分页组件 -->
    <Pagination v-bind:child-msg="pageparm"
@callFather="callFather"></Pagination>
    <!-- 编辑界面 -->
      <el-dialog :title="title" :visible.sync="addVisiable" width="30%"
@click="closeDialog">
      <el-form label-width="180px" :model="addForm" :rules="rules"
ref="editForm">
        <el-form-item label="信号滤波器信息编号" prop="signalFilterNo">
          <el-input size="small" v-model="addForm.signalFilterNo"
auto-complete="off" placeholder="请输入信号滤波器信息编号"></el-input>
        </el-form-item>
        <el-form-item label="信号滤波器信息名称" prop="signalFilterName">
          <el-input size="small" v-model="addForm.signalFilterName"
auto-complete="off" placeholder="请输入信号滤波器信息名称"></el-input>
        </el-form-item>
        <el-form-item label="信号滤波器信息类型" prop="signalFilterType">
          <el-select size="small" v-model="addForm.signalFilterType"
auto-complete="off" placeholder="请选择信号滤波器信息类型">
            <el-option label="数字滤波器" value="1"></el-option>
          </el-select>
        </el-form-item>
        <el-form-item label="信号滤波器信息状态" prop="signalFilterStatus">
          <el-select size="small" v-model="addForm.signalFilterStatus"
auto-complete="off" placeholder="请选择信号滤波器信息状态">
            <el-option label="运行中" value="1"></el-option>
          </el-select>
        </el-form-item>
      </el-form>
```

```
      <div slot="footer" class="dialog-footer">
        <el-button size="small" @click="closeDialog">取消</el-button>
        <el-button size="small" type="primary" :loading="loading" class="title"
@click="submitForm('editForm')">保存</el-button>
      </div>
    </el-dialog>
  </div>
</template>
<script>
import { OrderList, OrderRefund, OrderDelete } from '../../api/payMG'
import Pagination from '../../components/Pagination'
export default {
  data() {
    return {
      title:'',
      addVisiable: false,
       addForm:{
       signalFilterNo:'',
       signalFilterName:'',
       signalFilterType:'',
       address:'',
       signalFilterGender:'',
       age:'',
       signalFilterStatus:''
      },
      rules: {
        signalFilterNo: [
          { required: true, message: '请输入信号滤波器信息编号', trigger: 'blur' }
        ],
        signalFilterName: [
          { required: true, message: '请输入信号滤波器信息名称', trigger: 'blur' }
        ],
        signalFilterType: [
          { required: true, message: '请选择信号滤波器信息类型', trigger: 'blur' }
        ],
        signalFilterStatus: [
          { required: true, message: '请选择信号滤波器信息状态', trigger: 'blur' }
        ],
      },
      loading: false, //是显示加载
      editFormVisible: false, //控制编辑页面显示与隐藏
      title: '预览',
      editForm: {
        id: '',
        signalFilterName: '',
        payType: 1,
        partner: '',
        subMchId: '',
        appid: '',
        notifyUrl: '',
        signType: '',
```

```
      partnerKey: '',
      sellerUserId: '',
      certPath: '',
      certPassword: '',
      rsaKey: '',
      token: localStorage.getItem('logintoken')
    },
    formInline: {
      page: 1,
      limit: 10,
      machineNo: '',
      orderNo: '',
      transId: '',
      payType: 0,
      orderStatus: 0,
      token: localStorage.getItem('logintoken')
    },
    seletedata: {
      ids: '',
      token: localStorage.getItem('logintoken')
    },
    userparm: [],
    listData: [],
    // 分页参数
    pageparm: {
      currentPage: 1,
      pageSize: 10,
      total: 10
    }
  }
},
// 注册组件
components: {
  Pagination
},
/**
 * 数据发生改变
 */
/**
 * 创建完毕
 */
created() {
  this.getdata(this.formInline)
},
/**
 * 里面的方法只有被调用才会执行
 */
methods: {
  getdata(parameter) {
    this.loading = true
    // 模拟数据开始
```

```
let res = {
  code: 0,
  msg: null,
  count: 5,
  data: [
    {
      signalFilterNo:'VCxxxxxx',
      signalFilterName:'xxx 数字滤波器',
      signalFilterType:'数字滤波器',
      signalFilterStatus:"运行中",
      createTime: "2020-08-12"
    },
    {
      signalFilterNo:'VCxxxxxx',
      signalFilterName:'xxx 数字滤波器',
      signalFilterType:'数字滤波器',
      signalFilterStatus:"运行中",
      createTime: "2022-10-24"
    },
    {
      signalFilterNo:'VCxxxxxx',
      signalFilterName:'xxx 数字滤波器',
      signalFilterType:'数字滤波器',
      signalFilterStatus:"运行中",
      createTime: "2022-01-12"
    },
    {
      signalFilterNo:'VCxxxxxx',
      signalFilterName:'xxx 高通滤波器',
      signalFilterType:'高通滤波器',
      signalFilterStatus:"维护中",
      createTime: "2022-05-17"
    },
    {
      signalFilterNo:'VCxxxxxx',
      signalFilterName:'xxx 高通滤波器',
      signalFilterType:'高通滤波器',
      signalFilterStatus:"维护中",
      createTime: "2021-09-12"
    },
  ]
}
this.loading = false
this.listData = res.data
this.pageparm.currentPage = this.formInline.page
this.pageparm.pageSize = this.formInline.limit
this.pageparm.total = res.count
// 模拟数据结束
/***
 * 调用接口，注释上面模拟数据 取消下面注释
 */
```

```
    // OrderList(parameter)
    //   .then(res => {
    //     this.loading = false
    //     if (res.success == false) {
    //       this.$message({
    //         type: 'info',
    //         message: res.msg
    //       })
    //     } else {
    //       this.listData = res.data
    //       // 分页赋值
    //       this.pageparm.currentPage = this.formInline.page
    //       this.pageparm.pageSize = this.formInline.limit
    //       this.pageparm.total = res.count
    //     }
    //   })
    //   .catch(err => {
    //     this.loading = false
    //     this.$message.error('菜单加载失败，请稍后再试！')
    //   })
  },
  // 分页插件事件
  callFather(parm) {
    this.formInline.page = parm.currentPage
    this.formInline.limit = parm.pageSize
    this.getdata(this.formInline)
  },
  // 搜索事件
  search() {
    this.getdata(this.formInline)
  },
  //显示编辑界面
  handleEdit: function(row) {
    this.addVisiable = true
    if(row!=null){
      this.title='编辑信号滤波器信息'
      this.addForm.signalFilterNo = row.signalFilterNo
      this.addForm.signalFilterName = row.signalFilterName
      this.addForm.signalFilterType = row.signalFilterType
      this.addForm.signalFilterStatus = row.signalFilterStatus
    }else{
      this.title='添加信号滤波器信息'
      this.addForm.signalFilterNo = ''
      this.addForm.signalFilterName = ''
      this.addForm.signalFilterType = ''
      this.addForm.signalFilterStatus = ''
    }
  },
  // 编辑、增加页面保存方法
  submitForm(editData) {
    this.$refs[editData].validate(valid => {
```

```
        if (valid) {
          ConfigSave(this.editForm)
            .then(res => {
              this.editFormVisible = false
              this.loading = false
              if (res.success) {
                this.getdata(this.formInline)
                this.$message({
                  type: 'success',
                  message: '保存成功!'
                })
              } else {
                this.$message({
                  type: 'info',
                  message: res.msg
                })
              }
            })
            .catch(err => {
              this.editFormVisible = false
              this.loading = false
              this.$message.error('保存失败，请稍后再试!')
            })
        } else {
          return false
        }
      })
    },
    // 删除
    deleteUser(index, row) {
      this.$confirm('确定要删除吗?', '信息', {
        confirmButtonText: '确定',
        cancelButtonText: '取消',
        type: 'warning'
      })
        .then(() => {
          ConfigDelete(row.deptId)
            .then(res => {
              if (res.success) {
                this.$message({
                  type: 'success',
                  message: '已删除!'
                })
                this.getdata(this.formInline)
              } else {
                this.$message({
                  type: 'info',
                  message: res.msg
                })
              }
            })
```

```
            .catch(err => {
              this.loading = false
              this.$message.error('删除失败，请稍后再试！')
            })
        })
        .catch(() => {
          this.$message({
            type: 'info',
            message: '维护中删除'
          })
        })
    },
    // 关闭编辑、增加弹出框
    closeDialog(formsignalFilterName) {
      this.editFormVisible = false
      this.$refs[formsignalFilterName].resetFields()
    }
  }
}
</script>
<style scoped>
.user-search {
  margin-top: 20px;
}
.userRole {
  width: 100%;
}
</style>
<!-- 编码与解码数据管理 -->
<template>
  <el-card shadow="never" class="aui-card--fill">
    <div class="mod-worktarget__twgtworktargetcommonarea">
      <el-form :inline="true" :model="dataForm"
@keyup.enter.native="getDataList()">
        <el-form-item label="编码与解码数据类型" v-if="showWorktargetType">
          <worktarget-type-select v-model="dataForm.worktargetTypeId"
         @changeWorktargetType="changeWorktargetType"></worktarget-type-select>
        </el-form-item>
        <el-form-item label="编码与解码数据编号" prop="orgId">
          <dept-tree v-model="dataForm.orgId" @change="changeOrg">
          </dept-tree>
        </el-form-item>
        <el-form-item label="编码与解码数据名称">
          <el-input v-model="dataForm.name"></el-input>
        </el-form-item>
        <el-form-item>
          <el-button @click="getDataList()">{{ $t('query') }}</el-button>
        </el-form-item>
        <el-form-item>
          <el-button type="info"
@click="exportHandle()">{{ $t('export') }}</el-button>
        </el-form-item>
```

```
        <el-form-item>
          <el-button
v-if="$hasPermission('worktarget:twgtworktargetcommonarea:save')"
type="primary"
            @click="addOrUpdateHandle()">{{ $t('add') }}</el-button>
        </el-form-item>
        <el-form-item>
          <el-button
v-if="$hasPermission('worktarget:twgtworktargetcommonarea:delete')"
type="danger"
            @click="deleteHandle()">{{ $t('deleteBatch') }}</el-button>
        </el-form-item>
      </el-form>
      <div class="table-container" :style="[{ width: foldStatus ? '60%' :
'0px' }]">
        <el-table v-loading="dataListLoading" :data="dataList" border
@selection-change="dataListSelectionChangeHandle"
          style="width: 100%;" @row-click="onRowClick">
          <el-table-column type="selection" header-align="center" align="center"
width="50"></el-table-column>
          <el-table-column prop="name"  header-align="center"
align="center"></el-table-column>
          <el-table-column prop="worktargetTypeName" v-if="showWorktargetType"
header-align="center"
            align="center"></el-table-column>
          <el-table-column prop="address" header-align="center"
align="center"></el-table-column>
          <el-table-column prop="area" header-align="center"
align="center"></el-table-column>
          <el-table-column prop="orgName"  header-align="center"
align="center"></el-table-column>
            <template slot-scope="scope">
              <el-button
v-if="$hasPermission('worktarget:twgtworktargetcommonarea:update')" type="text"
size="small"
@click="addOrUpdateHandle(scope.row.id)">{{ $t('update') }}</el-button>
              <el-button
v-if="$hasPermission('worktarget:twgtworktargetcommonarea:delete')" type="text"
size="small"
       @click="deleteHandle(scope.row.id)">{{ $t('delete') }}</el-button>
            </template>
          </el-table-column>
        </el-table>
        <el-pagination
v-show="foldStatus" :current-pidCard="pidCard" :pidCard-sizes="[10, 20, 50,
100]" :pidCard-size="limit"
          :total="total" layout="total, sizes, prev, pidCardr, next, jumper"
@size-change="pidCardSizeChangeHandle"
          @current-change="pidCardCurrentChangeHandle">
        </el-pagination>
        <div class="fold_btn">
          <i v-show="!foldStatus" class="el-icon-d-arrow-right"
@click="clickFold" />
          <i v-show="foldStatus" class="el-icon-d-arrow-left" @click="clickFold"
/>
```

```
        </div>
      </div>
      <div class="map" id="commonarea-container" :style="[{ width: foldStatus ?
'40%' : '100%' }]">
      </div>
      <!-- 弹窗, 新增 / 修改 -->
      <add-or-update v-if="addOrUpdateVisible" ref="addOrUpdate"
@refreshDataList="getDataList"></add-or-update>
    </div>
  </el-card>
</template>
<script>
import AddOrUpdate from './worktarget-add-or-update'
import shopEdited from '@/assets/img/shopEdited.png'
import shopEditing from '@/assets/img/shopEditing.png'
import mixinViewModule from '@/mixins/view-module'
import AddOrUpdate from './worktarget-add-or-update'
import shopEdited from '@/assets/img/shopEdited.png'
import shopEditing from '@/assets/img/shopEditing.png'
import mixinViewModule from '@/mixins/view-module'
import AddOrUpdate from './twgtworktargetcommonarea-add-or-update'
import AMap from 'AMap'
export default {
  mixins: [mixinViewModule],
  data() {
    return {
      foldStatus: true,
      amap: {},
      mapConfig: {
        center: [116.404, 39.915],
        zoom: 6,
        resizeEnable: true,
        plugin: {
        },
        events: {
          init() {
          },
          click(e) {
          }
        }
      },
      polygonList: [],
      activePolygon: '',
      polygonMap: new Map(),
      worktargetTypeId: '',
      mixinViewModuleOptions: {
        getDataListURL: '/worktarget/twgtworktargetcommonarea/listPidCard',
        getDataListIsPidCard: true,
        exportURL: '/worktarget/twgtworktargetcommonarea/export',
        deleteURL: '/worktarget/twgtworktargetcommonarea',
        deleteIsBatch: true
      },
```

```
      dataForm: {
        orgId: '',
        worktargetTypeId: '',
        name: ''
      }
    }
  },
  watch: {
    polygonList(val) {
      this.amap.clearMap()
      if (this.polygonList && this.polygonList.length > 0) {
        for (var i = 0; i < this.polygonList.length; i++) {
          var polygonObj = this.polygonList[i]
          var polygon = new AMap.Polygon({
            map: this.amap,
            path: polygonObj.path,
            extData: polygonObj,
            fillColor: 'blue'
          })
          polygon.on('click', this.polygonClickCallback)
          this.polygonMap.set(polygonObj.id, polygon)
        }
      }
    }
  },
  components: {
    AddOrUpdate
  },
  mounted() {
    this.initMap()
    if (this.$route.meta.cusParams &&
this.$route.meta.cusParams.worktargetTypeId) {
      this.worktargetTypeId = this.$route.meta.cusParams.worktargetTypeId
      this.dataForm.worktargetTypeId = this.worktargetTypeId
      if (this.worktargetTypeId) {
        this.$http.get('/worktarget/worktargettype/' +
this.worktargetTypeId).then(({ data: res }) => {
          if (res.code != 0) {
            return this.$messidCard.error(res.msg)
          }
          this.worktargetType = res.data
          this.worktargetTypeName = this.worktargetType.worktargetTypeName
        }).catch(() => { })
      }
    }
  },
  destroyed() {
    if (this.amap) {
      this.amap.destroy()
    }
  },
  computed: {
```

```
    showWorktargetType() {
      return !this.worktargetTypeId
    }
  },
  methods: {
    initMap() {
      var amap = new AMap.Map('commonarea-container', {
        ...this.mapConfig
      })
      this.amap = amap
    },
    query() {
      this.dataListLoading = true
      this.$http.get(
        this.mixinViewModuleOptions.getDataListURL,
        {
          params: {
            order: this.order,
            orderField: this.orderField,
            pidCard: this.mixinViewModuleOptions.getDataListIsPidCard ?
this.pidCard : null,
            limit: this.mixinViewModuleOptions.getDataListIsPidCard ?
this.limit : null,
            ...this.dataForm
          }
        }
      ).then(({ data: res }) => {
        this.dataListLoading = false
        if (res.code != 0) {
          this.dataList = []
          this.total = 0
          return this.$messidCard.error(res.msg)
        }
        this.dataList = this.mixinViewModuleOptions.getDataListIsPidCard ?
res.data.list : res.data
        this.total = this.mixinViewModuleOptions.getDataListIsPidCard ?
res.data.total : 0
        this.mapConfig.zoom = 5
        var polygonList = []
        for (var i = 0; i < this.dataList.length; i++) {
          var dataInfo = this.dataList[i]
          if (dataInfo.geoinfo) {
            dataInfo.position = [dataInfo.geoCenter.lng, dataInfo.geoCenter.lat]
            dataInfo.path = []
            var geoPointList = dataInfo.geoPointList
            for (var l = 0; l < geoPointList.length; l++) {
              var point = [geoPointList[l].lng, geoPointList[l].lat]
              dataInfo.path.push(point)
            }
            polygonList.push(dataInfo)
          }
        }
```

```
        this.polygonList = polygonList
        if (this.map.polygonList.length > 0) {
          this.map.zoom = 14
          this.map.center = [this.map.polygonList[0].geoCenter.lng,
this.map.polygonList[0].geoCenter.lat]
        }
      }).catch(() => {
        this.dataListLoading = false
      })
    },
    clickFold() {
      if (this.foldStatus) {
        this.foldStatus = false
      } else {
        this.foldStatus = true
      }
    },
    onRowClick(row, column, event) {
      var polygon = this.polygonMap.get(row.id)
      this.amap.setFitView([polygon])
      // this.amap.setZoomAndCenter(this.mapConfig.zoom, [row.geoCenter.lng,
row.geoCenter.lat])
      this.activeId = row.id
      this.activeShow(row.id)
    },
    unActiveShow() {
      if (this.activePolygon) {
        var activeOptions = this.activePolygon.getOptions()
        activeOptions.fillColor = 'blue'
        this.activePolygon.setOptions(activeOptions)
      }
    },
    activeShow(polygonId) {
      this.unActiveShow()
      var polygon = this.polygonMap.get(polygonId)
      if (polygon != null) {
        var options = polygon.getOptions()
        options.fillColor = 'red'
        polygon.setOptions(options)
        this.activePolygon = polygon
      }
    },
    polygonClickCallback(e) {
      let polygon = e.target.getExtData()
      this.openWindowInfo(polygon)
    },
    openWindowInfo(polygon) {
      var info = []
      info.push('<div class="info-content"><table style="width: 100%">')
      info.push('<tr><td class="info-content-key"></td><td
class="info-content-value">' + polygon.name + '</td></tr>')
      info.push('<tr><td class="info-content-key"></td><td
```

```
class="info-content-value">' + polygon.orgName + '</td></tr>')
      info.push('<tr><td class="info-content-key"></td><td
class="info-content-value">' + polygon.address + '</td></tr>')
      info.push('</table></div>')
      var infoWindow = new AMap.InfoWindow({
        isCustom: true, // 使用自定义窗体
        offset: new AMap.Pixel(16, -40),
        content: this.createInfoWindow('信息', info.join(''))
      })
      infoWindow.open(this.amap, [polygon.geoCenter.lng, polygon.geoCenter.lat])
    },
    changeOrg(org) {
      this.dataForm.orgId = org.id
    },
    changeWorktargetType(worktargetType) {
      this.dataForm.worktargetTypeId = worktargetType.id
    },
    addOrUpdateHandle(id) {
      this.addOrUpdateVisible = true
      this.$nextTick(() => {
        this.$refs.addOrUpdate.dataForm.id = id
        this.$refs.addOrUpdate.worktargetTypeName = this.worktargetTypeName
        this.$refs.addOrUpdate.worktargetTypeId = this.worktargetTypeId
        this.$refs.addOrUpdate.dataForm.worktargetTypeId = this.worktargetTypeId
        this.$refs.addOrUpdate.init()
      })
    },
    createInfoWindow(title, content) {
      var info = document.createElement('div')
      info.className = 'custom-info input-card content-window-card'
      // 可以通过下面的方式修改自定义窗体的宽高
      info.style.width = '250px'
      // 定义顶部标题
      var top = document.createElement('div')
      var titleD = document.createElement('div')
      var closeX = document.createElement('img')
      top.className = 'info-top'
      titleD.innerHTML = title
      closeX.src = 'https://webapi.amap.com/imidCards/close2.gif'
      closeX.onclick = this.closeInfoWindow
      top.appendChild(titleD)
      top.appendChild(closeX)
      info.appendChild(top)
      // 定义中部内容
      var middle = document.createElement('div')
      middle.className = 'info-middle'
      middle.style.backgroundColor = 'white'
      middle.innerHTML = content
      info.appendChild(middle)
      // 定义底部内容
      var bottom = document.createElement('div')
```

```
      bottom.className = 'info-bottom'
      bottom.style.position = 'relative'
      bottom.style.top = '0px'
      bottom.style.margin = '0 auto'
      var sharp = document.createElement('img')
      sharp.src = 'https://webapi.amap.com/imidCards/sharp.png'
      bottom.appendChild(sharp)
      info.appendChild(bottom)
      return info
    },
    // 关闭信息窗体
    closeInfoWindow() {
      this.amap.clearInfoWindow()
    }
  }
}
</script>
<style lang="scss">
.table-container {
  background-color: white;
  width: 60%;
  height: calc(100vh - 215px);
  float: left;
  position: relative;
  z-index: 999;
  .fold_btn {
    position: absolute;
    right: -40px;
    top: 50%;
    height: 100px;
    width: 40px;
    background-color: rgba(255, 255, 255, 0.9);
    border-top-right-radius: 5px;
    border-bottom-right-radius: 5px;
    cursor: pointer;
    margin-top: -50px;
    text-align: center;
    i {
      font-size: 18px;
      line-height: 100px;
    }
  }
}
.map {
  float: left;
  width: 40%;
  height: calc(100vh - 195px); //
}
.content-window-card {
  position: relative;
  box-shadow: none;
```

```
  bottom: 0;
  left: 0;
  width: auto;
  padding: 0;
}
.content-window-card p {
  height: 2rem;
}
.custom-info {
  border: solid 1px silver;
}
div.info-top {
  position: relative;
  background: none repeat scroll 0 0 #F9F9F9;
  border-bottom: 1px solid #CCC;
  border-radius: 5px 5px 0 0;
}
div.info-top div {
  display: inline-block;
  color: #333333;
  font-size: 14px;
  font-weight: bold;
  line-height: 31px;
  padding: 0 10px;
  background-color: rgba(255, 255, 255, 0.9);
  border-top-right-radius: 5px;
  border-bottom-right-radius: 5px;
  cursor: pointer;
  margin-top: -50px;
  text-align: center;
}
div.info-top img {
  position: absolute;
  top: 10px;
  right: 10px;
  transition-duration: 0.25s;
}
div.info-top img:hover {
  box-shadow: 0px 0px 5px #000;
}
.info-content {
  padding: 5px 5px 5px 5px;
}
.info-content-key {
  width: 30%;
}
.info-content-value {
  width: 65%;
}</style>
<!-- 干扰模拟器信息管理 -->
<template>
```

```
  <el-card shadow="never" class="aui-card--fill">
    <div class="mod-worktarget__worktarget}">
      <el-form :inline="true" :model="dataForm"
@keyup.enter.native="getDataList()">
        <el-form-item label="干扰模拟器信息编号">
          <dept-tree v-model="dataForm.orgId" @change="changeOrg" placeholder="
请选择"></dept-tree>
        </el-form-item>
        <el-form-item label="干扰模拟器信息类型" v-if="showWorktargetType">
          <worktarget-type-select v-model="dataForm.worktargetTypeId"
@changeWorktargetType="changeWorktargetType"></worktarget-type-select>
        </el-form-item>
        <el-form-item label="干扰模拟器信息名称">
          <el-input v-model="dataForm.name"></el-input>
        </el-form-item>
        <el-form-item>
          <el-button @click="getDataList()">{{ $t('query') }}</el-button>
        </el-form-item>
        <el-form-item>
          <el-button @click="getDataList()">{{ $t('query') }}</el-button>
        </el-form-item>
        <el-form-item>
          <el-button type="info"
@click="exportHandle()">{{ $t('export') }}</el-button>
        </el-form-item>
        <el-form-item>
          <el-button v-if="$hasPermission('worktarget:worktarget:save')"
type="primary" @click="addOrUpdateHandle()">{{
            $t('add') }}</el-button>
        </el-form-item>
        <el-form-item>
          <el-button v-if="$hasPermission('worktarget:worktarget:delete')"
type="danger" @click="deleteHandle()">{{
            $t('deleteBatch') }}</el-button>
        </el-form-item>
      </el-form>
      <div class="table-container" :style="[{ width: foldStatus ? '60%' :
'0px' }]">
        <el-table v-loading="dataListLoading" :data="dataList" border
@selection-change="dataListSelectionChangeHandle"
          <el-table-column :label="$t('handle')" fixed="right"
header-align="center" align="center" width="150">
            <template slot-scope="scope">
              <el-button v-if="$hasPermission('worktarget:worktarget:update')"
type="text" size="small"
@click="addOrUpdateHandle(scope.row.id)">{{ $t('update') }}</el-button>
              <el-button v-if="$hasPermission('worktarget:worktarget:delete')"
type="text" size="small"
@click="deleteHandle(scope.row.id)">{{ $t('delete') }}</el-button>
            </template>
          </el-table-column>
        </el-table>
        <el-pagination :style="`${foldStatus ? '' :
```

```
'display:none'}`" :current-pidCard="pidCard" :pidCard-sizes="[10, 20, 50, 100]"
          :pidCard-size="limit" :total="total" layout="total, sizes, prev,
pidCardr, next, jumper"
          @size-change="pidCardSizeChangeHandle"
@current-change="pidCardCurrentChangeHandle">
        </el-pagination>
        <div class="fold_btn">
          <i v-show="!foldStatus" class="el-icon-d-arrow-right"
@click="clickFold" />
          <i v-show="foldStatus" class="el-icon-d-arrow-left" @click="clickFold"
/>
        </div>
      </div>
      <div
class="map" :id="`worktarget-container_${worktargetTypeId}`" :style="[{ width:
foldStatus ? '40%' : '100%' }]">
      </div>
      <!-- 弹窗, 新增 / 修改 -->
      <add-or-update v-if="addOrUpdateVisible" ref="addOrUpdate"
@refreshDataList="getDataList"></add-or-update>
    </div>
  </el-card>
</template>
<script>
import mixinViewModule from '@/mixins/view-module'
import AddOrUpdate from './worktarget-add-or-update'
import shopEdited from '@/assets/img/shopEdited.png'
import shopEditing from '@/assets/img/shopEditing.png'
import AMap from 'AMap'
export default {
  mixins: [mixinViewModule],
  data() {
    return {
      foldStatus: true,
      amap: '',
      mapConfig: {
        center: [116.404, 39.915],
        zoom: 6,
        resizeEnable: true,
        plugin: {
        },
        events: {
          init() {
          },
          click(e) {
          }
        }
      },
      markerList: [],
      activeMarker: '',
      markerMap: new Map(),
      mixinViewModuleOptions: {
        getDataListURL: '/worktarget/worktarget/listPidCard',
```

```
      getDataListIsPidCard: true,
      exportURL: '/worktarget/worktarget/export',
      deleteURL: '/worktarget/worktarget',
      deleteIsBatch: true,
      createdIsNeed: false,
      activatedIsNeed: false
    },
    activeId: '',
    worktargetTypeId: '',
    plftype: '',
    worktargetTypeName: '',
    worktargetTypeList: [],
    worktargetTypeMap: new Map(),
    dataForm: {
      orgId: '',
      worktargetTypeId: '',
      name: ''
    }
  }
},
watch: {
  markerList(val) {
    this.cleanMap()
    if (this.markerList && this.markerList.length > 0) {
      for (var i = 0; i < this.markerList.length; i++) {
        var markerObj = this.markerList[i]
        this.createMarker(markerObj)
      }
    }
    this.amap.setFitView()
  }
},
computed: {
  showWorktargetType() {
    return !this.worktargetTypeId
  },
  showPoint() {
    return this.plftype == 1
  },
  showPolyline() {
    return this.plftype == 2 || this.plftype == 3
  },
  ploygonColor() {
    return (ploygonObj) => {
      return this.activeId == ploygonObj.id ? 'red' : 'blue'
    }
  },
  ploylineColor() {
    return (ploylineObj) => {
      return this.activeId == ploylineObj.id ? 'red' : 'blue'
    }
```

```
    },
    markerImg() {
      return (marker) => {
        return this.activeId == marker.id ? 'img/start.png' :
'img/villidCard48.png'
      }
    }
  },
  components: {
    AddOrUpdate
  },
  mounted() {
    if (this.$route.meta.cusParams &&
this.$route.meta.cusParams.worktargetTypeId) {
      this.worktargetTypeId = this.$route.meta.cusParams.worktargetTypeId
      this.dataForm.worktargetTypeId = this.worktargetTypeId
      if (this.worktargetTypeId) {
        this.$http.get('/worktarget/worktargettype/' +
this.worktargetTypeId).then(({ data: res }) => {
          if (res.code != 0) {
            return this.$messidCard.error(res.msg)
          }
          this.worktargetType = res.data
          this.plftype = this.worktargetType.plftype
          this.worktargetTypeName = this.worktargetType.worktargetTypeName
        }).catch(() => { })
      }
    }
    this.initMap()
    this.query()
  },
  methods: {
    initMap() {
      var divId = 'worktarget-container_' + this.worktargetTypeId
      var amap = new AMap.Map(divId, {
        ...this.mapConfig
      })
      this.amap = amap
    },
    query() {
      this.dataListLoading = true
      this.$http.get(
        this.mixinViewModuleOptions.getDataListURL,
        {
          params: {
            order: this.order,
            orderField: this.orderField,
            pidCard: this.mixinViewModuleOptions.getDataListIsPidCard ?
this.pidCard : null,
            limit: this.mixinViewModuleOptions.getDataListIsPidCard ?
this.limit : null,
            ...this.dataForm
```

```
          }
        }
      ).then(({ data: res }) => {
        this.dataListLoading = false
        if (res.code != 0) {
          this.dataList = []
          this.total = 0
          return this.$messidCard.error(res.msg)
        }
        this.dataList = this.mixinViewModuleOptions.getDataListIsPidCard ?
res.data.list : res.data
        this.total = this.mixinViewModuleOptions.getDataListIsPidCard ?
res.data.total : 0
        this.mapConfig.zoom = 5
        var markerList = []
        for (var i = 0; i < this.dataList.length; i++) {
          var dataInfo = this.dataList[i]
          if (dataInfo.plftype == 2 || dataInfo.plftype == 3) {
            if (dataInfo.geoinfo) {
              dataInfo.position = [dataInfo.geoCenter.lng,
dataInfo.geoCenter.lat]
              dataInfo.path = []
              var geoPointList = dataInfo.geoPointList
              for (var l = 0; l < geoPointList.length; l++) {
                var point = [geoPointList[l].lng, geoPointList[l].lat]
                dataInfo.path.push(point)
              }
              markerList.push(dataInfo)
            }
          } else if (dataInfo.plftype == 1) {
            if (dataInfo.lat && dataInfo.lng) {
              markerList.push(dataInfo)
            }
          }
        }
        this.markerList = markerList
      }).catch(() => {
        this.dataListLoading = false
      })
    },
    getWorktargetTypeList() {
      return this.$http
        .get('/worktarget/worktargettype/list', { params: {} })
        .then(({ data: res }) => {
          if (res.code != 0) {
            return this.$messidCard.error(res.msg)
          }
          this.worktargetTypeList = res.data
          if (this.worktargetTypeList && this.worktargetTypeList.length > 0) {
            for (var i = 0; i < this.worktargetTypeList.length; i++) {
              var worktargetType = this.worktargetTypeList[i]
              this.worktargetTypeMap.set(worktargetType.id, worktargetType)
```

```
          }
        }
      })
      .catch(() => { })
  },
      clickFold() {
    if (this.foldStatus) {
      this.foldStatus = false
    } else {
      this.foldStatus = true
    }
  },
  createMarker(markerObj) {
    if (markerObj.plftype == 1) {
      if (markerObj.lat && markerObj.lng) {
        var marker = new AMap.Marker({
          map: this.amap,
          position: [markerObj.lng, markerObj.lat],
          extData: markerObj,
          icon: shopEdited
        })
        marker.on('click', this.markerClickCallback)
        this.markerMap.set(markerObj.id, marker)
      }
    } else if (markerObj.plftype == 2) {
      var polyline = new AMap.Polyline({
        map: this.amap,
        path: markerObj.path,
        extData: markerObj,
        fillColor: 'blue'
      })
      polyline.on('click', this.markerClickCallback)
      this.markerMap.set(markerObj.id, polyline)
    } else if (markerObj.plftype == 3) {
      var polygon = new AMap.Polygon({
        map: this.amap,
        path: markerObj.path,
        extData: markerObj,
        fillColor: 'blue'
      })
      polygon.on('click', this.markerClickCallback)
      this.markerMap.set(markerObj.id, polygon)
    }
  },
  markerClickCallback(e) {
    let marker = e.target.getExtData()
    this.openWindowInfo(marker)
  },
  openWindowInfo(marker) {
    var info = []
    info.push('<p>: ' + marker.name + '</p>')
```

```
    info.push('<p>: ' + marker.orgName + '</p>')
    if (!this.worktargetTypeId) {
      info.push('<p> + marker.worktargetTypeName + '</p>')
    info.push('<p>: ' + marker.address + '</p>')
    var infoWindow = new AMap.InfoWindow({
      content: info.join('') // 使用默认信息窗体框样式，显示信息内容
    })
    infoWindow.open(this.amap, [marker.lng, marker.lat])
  },
  unActiveShow() {
    if (this.activeMarker) {
      var extData = this.activeMarker.getExtData()
      if (extData) {
        if (extData.plftype == 1) {
          this.activeMarker.setIcon(shopEdited)
        } else if (extData.plftype == 2) {
          var polylineOptions = this.activeMarker.getOptions()
          polylineOptions.strokeColor = 'blue'
          this.activeMarker.setOptions(polylineOptions)
        } else if (extData.plftype == 3) {
          var polygonOptions = this.activeMarker.getOptions()
          polygonOptions.strokeColor = 'blue'
          this.activeMarker.setOptions(polygonOptions)
        }
      }
    }
  },
  activeShow(markerId) {
    this.unActiveShow()
    var marker = this.markerMap.get(markerId)
    if (marker != null) {
      var extData = marker.getExtData()
      if (extData) {
        if (extData.plftype == 1) {
          marker.setIcon(shopEditing)
        } else if (extData.plftype == 2) {
          var polylineOptions = marker.getOptions()
          polylineOptions.strokeColor = 'red'
          marker.setOptions(polylineOptions)
        } else if (extData.plftype == 3) {
          var polygonOptions = marker.getOptions()
          polygonOptions.strokeColor = 'red'
          marker.setOptions(polygonOptions)
        }
      }
      this.amap.setFitView(marker)
    }
    this.activeMarker = marker
  },
  cleanMap() {
    if (this.amap) {
```

```
        this.amap.clearMap()
      }
    },
    zoomEnd(e) {
      this.map.zoom = e.target.getZoom()
    },
    onInput(value) {
      this.getDataList()
    },
    onRowClick(row, column, event) {
      this.activeShow(row.id)
    },
    onRowSelect(selection, row) {
      var isExist = selection.find(x => x.id == row.id)
      if (isExist) {
        this.map.zoom = 14
        this.map.center = {
          lng: row.geoCenter.lng,
          lat: row.geoCenter.lat
        }
      }
    },
    addOrUpdateHandle(id) {
      this.addOrUpdateVisible = true
      this.$nextTick(() => {
        this.$refs.addOrUpdate.dataForm.id = id
        if (this.worktargetTypeId) {
          this.$refs.addOrUpdate.worktargetTypeName = this.worktargetTypeName
          this.$refs.addOrUpdate.worktargetTypeId = this.worktargetTypeId
          this.$refs.addOrUpdate.dataForm.worktargetTypeId =
          this.worktargetTypeId
          this.$refs.addOrUpdate.plftype = this.plftype
          this.$refs.addOrUpdate.dataForm.plftype = this.plftype
        }
        this.$refs.addOrUpdate.init()
      })
    },
    changeWorktargetType(worktargetType) {
      this.dataForm.worktargetTypeId = worktargetType.id
    },
    changeOrg(org) {
      this.dataForm.orgId = org.id
    }
  }
}
</script>
<style lang="scss">
.table-container {
  background-color: white;
  width: 60%;
  height: calc(100vh - 215px);
```

```
  float: left;
  position: relative;
  z-index: 999;
  .fold_btn {
    position: absolute;
    right: -40px;
    top: 50%;
    height: 100px;
    width: 40px;
    background-color: rgba(255, 255, 255, 0.9);
    border-top-right-radius: 5px;
    border-bottom-right-radius: 5px;
    cursor: pointer;
    margin-top: -50px;
    text-align: center;
    i {
      font-size: 18px;
      line-height: 100px;
    }
  }
}
.map {
  float: left;
  width: 40%;
  height: calc(100vh - 215px);
}
</style>
<!-- 系统用户管理 -->
<template>
  <div>
    <el-form :inline="true" :model="formInline" class="user-search">
      <el-form-item label="搜索：">
        <el-select size="small" v-model="formInline.isLock" placeholder="请选择
">
          <el-option label="全部" value=""></el-option>
          <el-option label="正常" value="N"></el-option>
          <el-option label="已锁定" value="Y"></el-option>
        </el-select>
      </el-form-item>
      <el-form-item label="">
        <el-input size="small" v-model="formInline.userName" placeholder="输入用
户名"></el-input>
      </el-form-item>
      <el-form-item label="">
        <el-input size="small" v-model="formInline.userMobile" placeholder="输入
手机号"></el-input>
      </el-form-item>
      <el-form-item>
        <el-button size="small" type="primary" icon="el-icon-search"
@click="search">搜索</el-button>
        <el-button size="small" type="primary" icon="el-icon-plus"
@click="handleEdit()">添加</el-button>
```

```
      </el-form-item>
    </el-form>
    <!--列表-->
    <el-table size="small" @selection-change="selectChange" :data="userData"
highlight-current-row v-loading="loading" border element-loading-text="拼命加载
中" style="width: 100%;">
      <el-table-column align="center" type="selection" width="50">
      </el-table-column>
      <el-table-column align="center" sortable prop="userName" label="用户名"
width="120">
      </el-table-column>
      <el-table-column align="center" sortable prop="userRealName" label="姓名"
width="120">
      </el-table-column>
      <el-table-column align="center" sortable prop="userMobile" label="手机号"
width="120">
      </el-table-column>
      <el-table-column align="center" sortable prop="userSex" label="性别"
min-width="50">
      </el-table-column>
      <el-table-column align="center" sortable prop="isLock" label="状态"
min-width="50">
        <template slot-scope="scope">
          <el-switch active-color="#13ce66" inactive-color="#ff4949"
@change="editType(scope.$index, scope.row)">
          </el-switch>
        </template>
      </el-table-column>
      <el-table-column label="操作" min-width="300">
        <template slot-scope="scope">
          <el-button size="mini" @click="handleEdit(scope.$index, scope.row)">
编辑</el-button>
          <el-button size="mini" type="danger" @click="deleteUser(scope.$index,
scope.row)">删除</el-button>
          <el-button size="mini" type="success" @click="resetpwd(scope.$index,
scope.row)">重置密码</el-button>
          <el-button size="mini" type="success" @click="dataAccess(scope.$index,
scope.row)">数据权限</el-button>
          <el-button size="mini" type="success"
@click="refreshCache(scope.$index, scope.row)">刷新缓存</el-button>
        </template>
      </el-table-column>
    </el-table>
    <!-- 分页组件 -->
    <Pagination v-bind:child-msg="pageparm"
@callFather="callFather"></Pagination>
    <!-- 编辑界面 -->
    <el-dialog :title="title" :visible.sync="editFormVisible" width="30%"
@click='closeDialog("edit")'>
      <el-form label-width="80px"
ref="editForm" :model="editForm" :rules="rules">
        <el-form-item label="用户名" prop="userName">
          <el-input size="small" v-model="editForm.userName" auto-complete="off"
placeholder="请输入用户名"></el-input>
```

```
        </el-form-item>
        <el-form-item label="姓名" prop="userRealName">
          <el-input size="small" v-model="editForm.userRealName"
auto-complete="off" placeholder="请输入姓名"></el-input>
        </el-form-item>
        <el-form-item label="角色" prop="roleId">
          <el-select size="small" v-model="editForm.roleId" placeholder="请选择"
class="userRole">
            <el-option label="管理员" value="1"></el-option>
            <el-option label="普通用户" value="2"></el-option>
          </el-select>
        </el-form-item>
        <el-form-item label="手机号" prop="userMobile">
          <el-input size="small" v-model="editForm.userMobile" placeholder="请输
入手机号"></el-input>
        </el-form-item>
        <el-form-item label="性别" prop="userSex">
          <el-radio v-model="editForm.userSex" label="男">男</el-radio>
          <el-radio v-model="editForm.userSex" label="女">女</el-radio>
        </el-form-item>
      </el-form>
      <div slot="footer" class="dialog-footer">
        <el-button size="small" @click='closeDialog("edit")'>取消</el-button>
        <el-button size="small" type="primary" :loading="loading" class="title"
@click="submitForm('editForm')">保存</el-button>
      </div>
    </el-dialog>
    <!-- 数据权限 -->
    <el-dialog title="数据权限" :visible.sync="dataAccessshow" width="30%"
@click='closeDialog("perm")'>
      <el-tree ref="tree"
default-expand-all="" :data="UserDept" :props="defaultProps" :default-checked-ke
ys="checkmenu" node-key="id" show-checkbox>
      </el-tree>
      <div slot="footer" class="dialog-footer">
        <el-button size="small" @click='closeDialog("perm")'>取消</el-button>
        <el-button size="small" type="primary" :loading="loading" class="title"
@click="menuPermSave">保存</el-button>
      </div>
    </el-dialog>
    <!-- 所属单位 -->
    <el-dialog title="所属单位" :visible.sync="unitAccessshow" width="30%"
@click='closeDialog("unit")'>
      <el-tree ref="tree"
default-expand-all="" :data="UserDept" :props="defaultProps"
@check-change="handleClick" :default-checked-keys="checkmenu" node-key="id"
show-checkbox check-strictly>
      </el-tree>
      <div slot="footer" class="dialog-footer">
        <el-button size="small" @click='closeDialog("unit")'>取消</el-button>
        <el-button size="small" type="primary" :loading="loading" class="title"
@click="unitPermSave">保存</el-button>
      </div>
    </el-dialog>
```

```
  </div>
</template>
<script>
// 导入请求方法
import {
  userList,
  userSave,
  userDelete,
  userPwd,
  userExpireToken,
  userFlashCache,
  userLock,
  UserDeptTree,
  UserDeptSave,
  UserDeptdeptTree,
  UserChangeDept
} from '../../api/userMG'
import Pagination from '../../components/Pagination'
export default {
  data() {
    return {
      nshow: true, //switch 开启
      fshow: false, //switch 关闭
      loading: false, //是显示加载
      title: '添加用户',
      editFormVisible: false, //控制编辑页面显示与隐藏
      dataAccessshow: false, //控制数据权限显示与隐藏
      unitAccessshow: false, //控制所属单位隐藏与显示
      // 编辑与添加
      editForm: {
        userId: '',
        userName: '',
        userRealName: '',
        roleId: '',
        userMobile: '',
        userEmail: '',
        userSex: '',
        token: localStorage.getItem('logintoken')
      },
      unitparm: {
        userIds: '',
        deptId: '',
        deptName: ''
      },
      // 选择数据
      selectdata: [],
      // rules 表单验证
      rules: {
        userName: [
          { required: true, message: '请输入用户名', trigger: 'blur' }
        ],
```

```
      userRealName: [
        { required: true, message: '请输入姓名', trigger: 'blur' }
      ],
      roleId: [{ required: true, message: '请选择角色', trigger: 'blur' }],
      userMobile: [
        { required: true, message: '请输入手机号', trigger: 'blur' },
        {
          pattern: /^1(3\d|47|5((?!4)\d)|7(0|1|[6-8])|8\d)\d{8,8}$/,
          required: true,
          message: '请输入正确的手机号',
          trigger: 'blur'
        }
      ],
      userSex: [{ required: true, message: '请选择性别', trigger: 'blur' }]
    },
    // 删除用户
    seletedata: {
      ids: '',
      token: localStorage.getItem('logintoken')
    },
    // 重置密码
    resetpsd: {
      userId: '',
      token: localStorage.getItem('logintoken')
    },
    offline: {
      token: localStorage.getItem('logintoken')
    },
    // 请求数据参数
    formInline: {
      page: 1,
      limit: 10,
      deptId: '',
      userName: '',
      userMobile: '',
      isLock: ''
    },
    //用户数据
    userData: [],
    // 数据权限
    UserDept: [],
    defaultProps: {
      children: 'children',
      label: 'name'
    },
    // 选中
    checkmenu: [],
    //参数 role
    saveroleId: '',
    // 分页参数
    pageparm: {
```

```
        currentPage: 1,
        pageSize: 10,
        total: 10
      }
    }
  },
  // 注册组件
  components: {
    Pagination
  },
  /**
   * 数据发生改变
   */
  watch: {},
  /**
   * 创建完毕
   */
  created() {
    this.getdata(this.formInline)
  },
  /**
   * 里面的方法只有被调用才会执行
   */
  methods: {
    // 分页插件事件
    callFather(parm) {
      this.formInline.page = parm.currentPage
      this.formInline.limit = parm.pageSize
      this.getdata(this.formInline)
    },
    //搜索事件
    search() {
      this.getdata(this.formInline)
    },
    // 修改 type
    editType: function(index, row) {
      this.loading = true
      let parm = {
        lock: '',
        userId: '',
        token: localStorage.getItem('logintoken')
      }
      parm.userId = row.userId
      let lock = row.isLock
      if (lock == 'N') {
        parm.lock = 'Y'
      } else {
        parm.lock = 'N'
      }
      // 修改状态
      userLock(parm).then(res => {
```

```
        this.loading = false
        if (res.success == false) {
          this.$message({
            type: 'info',
            message: res.msg
          })
        } else {
          this.$message({
            type: 'success',
            message: '状态修改成功'
          })
          this.getdata(this.formInline)
        }
      })
    },
    //显示编辑界面
    handleEdit: function(index, row) {
      this.editFormVisible = true
      if (row != undefined && row != 'undefined') {
        this.title = '修改用户'
        this.editForm.userId = row.userId
        this.editForm.userName = row.userName
        this.editForm.userRealName = row.userRealName
        this.editForm.roleId = row.roleId
        this.editForm.userMobile = row.userMobile
        this.editForm.userEmail = row.userEmail
        this.editForm.userSex = row.userSex
      } else {
        this.title = '添加用户'
        this.editForm.userId = ''
        this.editForm.userName = ''
        this.editForm.userRealName = ''
        this.editForm.roleId = ''
        this.editForm.userMobile = ''
        this.editForm.userEmail = ''
        this.editForm.userSex = ''
      }
    },
    // 编辑、添加提交方法
    submitForm(editData) {
      this.$refs[editData].validate(valid => {
        if (valid) {
          // 请求方法
          userSave(this.editForm)
            .then(res => {
              this.editFormVisible = false
              this.loading = false
              if (res.success) {
                this.getdata(this.formInline)
                this.$message({
                  type: 'success',
```

```
              message: '数据保存成功！'
            })
          } else {
            this.$message({
              type: 'info',
              message: res.msg
            })
          }
        })
        .catch(err => {
          this.editFormVisible = false
          this.loading = false
          this.$message.error('保存失败，请稍后再试！')
        })
    } else {
      return false
    }
  })
},
// 显示部门设置
handleunit: function(index, row) {
  this.unitAccessshow = true
  let parms = 0
  UserDeptdeptTree(parms)
    .then(res => {
      if (res.data.success) {
        this.UserDept = this.changeArr(res.data.data)
      } else {
        this.$message({
          type: 'info',
          message: res.msg
        })
      }
    })
    .catch(err => {
      this.loading = false
      this.$message.error('配置权限失败,请稍后再试！')
    })
},
handleClick(data, checked, node) {
  if (checked) {
    this.$refs.tree.setCheckedNodes([])
    this.$refs.tree.setCheckedNodes([data])
    this.unitparm.deptId = data.id
    this.unitparm.deptName = data.name
    //交叉点击节点
  } else {
  }
},
// 保存部门
unitPermSave() {
```

```
      let len = this.selectdata
      let ids = []
      if (len != 0) {
        for (let i = 0; i < len.length; i++) {
          ids.push(len[i].userId)
        }
      }
      this.unitparm.userIds = ids.join(',')
      UserChangeDept(this.unitparm)
        .then(res => {
          this.unitAccessshow = false
          if (res.success) {
            this.$message({
              type: 'success',
              message: '设置成功!'
            })
            this.getdata(this.formInline)
          } else {
            this.$message({
              type: 'info',
              message: res.msg
            })
          }
        })
        .catch(err => {
          this.loading = false
          this.$message.error('设置失败,请稍后再试!')
        })
    },
    // 选择复选框事件
    selectChange(val) {
      this.selectdata = val
    },
    // 关闭编辑、增加弹出框
    closeDialog(dialog) {
      if (dialog == 'edit') {
        this.editFormVisible = false
      } else if (dialog == 'perm') {
        this.dataAccessshow = false
      } else if (dialog == 'unit') {
        this.unitAccessshow = false
      }
    },
    // 删除用户
    deleteUser(index, row) {
      this.$confirm('确定要删除吗?', '信息', {
        confirmButtonText: '确定',
        cancelButtonText: '取消',
        type: 'warning'
      })
        .then(() => {
```

```
        // 删除
        userDelete(row.id)
          .then(res => {
            if (res.success) {
              this.$message({
                type: 'success',
                message: '已删除!'
              })
              this.getdata(this.formInline)
            } else {
              this.$message({
                type: 'info',
                message: res.msg
              })
            }
          })
          .catch(err => {
            this.loading = false
            this.$message.error('删除失败，请稍后再试！')
          })
      })
      .catch(() => {
        this.$message({
          type: 'info',
          message: '已取消删除！'
        })
      })
  },
  // 重置密码
  resetpwd(index, row) {
    this.resetpsd.userId = row.userId
    this.$confirm('确定要重置密码吗?', '信息', {
      confirmButtonText: '确定',
      cancelButtonText: '取消',
      type: 'warning'
    })
      .then(() => {
        userPwd(this.resetpsd)
          .then(res => {
            if (res.success) {
              this.$message({
                type: 'success',
                message: '重置密码成功！'
              })
              this.getdata(this.formInline)
            } else {
              this.$message({
                type: 'info',
                message: res.msg
              })
            }
```

```
          })
          .catch(err => {
            this.loading = false
          })
      })
  },
  // 数据权限
  dataAccess: function(index, row) {
    this.dataAccessshow = true
    this.saveroleId = row.userId
    UserDeptTree(row.userId)
      .then(res => {
        if (res.data.success) {
          this.checkmenu = this.changemenu(res.data.data)
          this.UserDept = this.changeArr(res.data.data)
        } else {
          this.$message({
            type: 'info',
            message: res.data.msg
          })
        }
      })
      .catch(err => {
        this.loading = false
        this.$message.error('获取权限失败，请稍后再试！')
      })
  },
  //数据格式化
  changeArr(data) {
    var pos = {}
    var tree = []
    var i = 0
    while (data.length != 0) {
      if (data[i].pId == 0) {
        tree.push({
          id: data[i].id,
          name: data[i].name,
          pId: data[i].pId,
          open: data[i].open,
          checked: data[i].checked,
          children: []
        })
        pos[data[i].id] = [tree.length - 1]
        data.splice(i, 1)
        i--
      } else {
        var posArr = pos[data[i].pId]
        if (posArr != undefined) {
          var obj = tree[posArr[0]]
          for (var j = 1; j < posArr.length; j++) {
            obj = obj.children[posArr[j]]
```

```
          }
          obj.children.push({
            id: data[i].id,
            name: data[i].name,
            pId: data[i].pId,
            open: data[i].open,
            checked: data[i].checked,
            children: []
          })
          pos[data[i].id] = posArr.concat([obj.children.length - 1])
          data.splice(i, 1)
          i--
        }
      }
      i++
      if (i > data.length - 1) {
        i = 0
      }
    }
    return tree
  },
  // 选中菜单
  changemenu(arr) {
    let change = []
    for (let i = 0; i < arr.length; i++) {
      if (arr[i].checked) {
        change.push(arr[i].id)
      }
    }
    return change
  },
  // 菜单权限-保存
  menuPermSave() {
    let parm = {
      userId: this.saveroleId,
      deptIds: ''
    }
    let node = this.$refs.tree.getCheckedNodes()
    let moduleIds = []
    if (node.length != 0) {
      for (let i = 0; i < node.length; i++) {
        moduleIds.push(node[i].id)
      }
      parm.deptIds = JSON.stringify(moduleIds)
    }
    UserDeptSave(parm)
      .then(res => {
        if (res.success) {
          this.$message({
            type: 'success',
            message: '权限保存成功'
```

```
            })
            this.dataAccessshow = false
            this.getdata(this.formInline)
          } else {
            this.$message({
              type: 'info',
              message: res.msg
            })
          }
        })
        .catch(err => {
          this.loading = false
          this.$message.error('权限保存失败，请稍后再试！')
        })
    },
    // 刷新缓存
    refreshCache(index, row) {
      userFlashCache(row.userName)
        .then(res => {
          if (res.success) {
            this.$message({
              type: 'success',
              message: '刷新成功！'
            })
            this.getdata(this.formInline)
          } else {
            this.$message({
              type: 'info',
              message: res.msg
            })
          }
        })
        .catch(err => {
          this.loading = false
          this.$message.error('刷新失败，请稍后再试！')
        })
    }
  }
}
</script>
<style scoped>
.user-search {
  margin-top: 20px;
}
.userRole {
  width: 100%;
}
</style>
<!-- 信号序列生成信息管理 -->
<template>
  <el-dialog width="80%" :visible.sync="visible" :title="worktargetTypeName +
```

```
(!dataForm.id ? $t('add') : $t('update'))"
    :close-on-click-modal="false" :close-on-press-escape="false">
    <div class="commonarea-edit-mian">
      <div class="commonarea-form-container">
        <el-form :model="dataForm" :rules="dataRule" ref="dataForm"
@keyup.enter.native="dataFormSubmitHandle()"
          :label-width="$i18n.locale === 'en-US' ? '120px' : '80px'">
          <el-row>
            <el-col :span="12">
              <el-form-item label="信号序列生成信息名称" prop="name">
                <el-input v-model="dataForm.name"></el-input>
              </el-form-item>
            </el-col>
            <el-col :span="12">
              <el-form-item label=信号序列生成信息类型" prop="area">
                <el-input :readonly="true" v-model="dataForm.area"
                placeholder="">
                  <template slot="append" v-if="areaUnit !=
''">{{ areaUnit }}</template>
                </el-input>
              </el-form-item>
            </el-col>
          </el-row>
          <el-row>
            <el-col :span="12">
              <el-form-item label="信号序列生成信息编号" prop="orgId">
                <dept-tree v-model="dataForm.orgId"
@change="changeOrg" ></dept-tree>
              </el-form-item>
            </el-col>
            <el-col :span="12">
              <el-form-item label="信号序列生成信息状态" prop="contactTel">
                <el-input v-model="dataForm.contactTel"></el-input>
              </el-form-item>
            </el-col>
          </el-row>
          <el-row>
            <el-col :span="24"></el-col>
          </el-row>
          <el-row>
            <el-col :span="24"></el-col>
            </el-row>
          <el-row>
            <el-col :span="24" style="text-align:center;">
              <el-button @click="visible = false">{{ $t('cancel') }}</el-button>
              <el-button type="primary"
@click="dataFormSubmitHandle()">{{ $t('confirm') }}</el-button>
            </el-col>
          </el-row>
        </el-form>
      </div>
      <div class="commonarea-form-map">
```

```
        <div id="commonarea-edit-container" class="commonarea-edit-map">
          <div class="text-btn">
            <el-button v-show="!dragging" type="primary" size="small"
@click="openDraw">
            </el-button>
            <el-button v-show="dragging" type="primary" size="small"
@click="closeDraw">
            </el-button>
          </div>
        </div>
      </div>
    </div>
  </el-dialog>
</template>
<script>
import debounce from 'lodash/debounce'
import AMap from 'AMap'
export default {
  data() {
    let self = this
    return {
      visible: false,
      worktargetTypeName: '',
      worktargetTypeId: '',
      dataForm: {
        id: '',
        name: '',
        worktargetTypeId: '',
        address: '',
        lng: '',
        lat: '',
        area: '',
        geoinfo: '',
        plftype: '3',
        orgId: '',
        districtCode: '',
        buildUnit: '',
        contacts: '',
        contactTel: '',
        creater: '',
        createDate: '',
        updater: '',
        updateDate: ''
      },
      dragging: false,
      editAmap: {},
      polyEditor: {},
      polygonObj: {},
      polygon: '',
      areaUnit: '',
      mapConfig: {
        center: [116.404, 39.915],
```

```
        zoom: 6,
        resizeEnable: true,
        plugin: {
        },
        events: {
          init() {
          },
          click(e) {
          }
        }
      }
    }
  },
  computed: {
    dataRule() {
      return {
        worktargetTypeId: [{ required: true, messidCard:
this.$t('validate.required'), trigger: 'blur,change' }],
        name: [{required: true, messidCard: this.$t('validate.required'), trigger:
'blur' }],
        address: [{ required: true, messidCard: this.$t('validate.required'),
trigger: 'blur' }],
        orgId: [{ required: true, messidCard: this.$t('validate.required'),
trigger: 'blur,change' }]
      }
    }
  },
  methods: {
    initMap() {
      var editAmap = new AMap.Map('commonarea-edit-container', {
        ...this.mapConfig
      })
      let selft = this
      this.editAmap = editAmap
      this.createPolyEditor(editAmap)
    },
    init() {
      this.visible = true
      this.$nextTick(() => {
        this.initMap()
        this.$refs['dataForm'].resetFields()
        if (this.dataForm.id) {
          this.getInfo()
        }
      })
    },
    createPolyEditor(editAmap) {
      this.polyEditor = new AMap.PolygonEditor(editAmap)
      let self = this
      this.polyEditor.on('add', function (data) {
        var polygon = data.target
        self.polyEditor.addAdsorbPolygons(polygon)
```

```
        polygon.on('dblclick', () => {
          self.polyEditor.setTarget(polygon)
          self.polyEditor.open()
        })
      })
      this.polyEditor.on('end', function (event) {
        console.log(event)
      })
    },
    // 获取信息
    getInfo() {
      this.$http.get('/worktarget/twgtworktargetcommonarea/' +
this.dataForm.id).then(({ data: res }) => {
        if (res.code !== 0) {
          return this.$messidCard.error(res.msg)
        }
        this.dataForm = {
          ...this.dataForm,
          ...res.data
        }
        this.polygonObj = {
          ...this.dataForm
        }
        this.polygonObj.position = [this.polygonObj.geoCenter.lng,
this.polygonObj.geoCenter.lat]
        this.polygonObj.path = []
        var geoPointList = this.polygonObj.geoPointList
        for (var l = 0; l < geoPointList.length; l++) {
          var point = [geoPointList[l].lng, geoPointList[l].lat]
          this.polygonObj.path.push(point)
        }
        this.polygon = this.createPolygon(this.polygonObj)
        this.editAmap.setFitView([this.polygon])
      }).catch(() => { })
    },
    changeOrg(org) {
      this.dataForm.orgId = org.id
    },
    getGeoInfo(polygonObj) {
      var path = polygonObj.path
      var geoinfo = ''
      for (var i = 0; i < path.length; i++) {
        var pathPoint = path[i]
        geoinfo += pathPoint.lat + '_' + pathPoint.lng + 'P'
      }
      return geoinfo
    },
    createPolygon(polygonObj) {
      var polygon = new AMap.Polygon({
        map: this.editAmap,
        path: polygonObj.path,
        extData: polygonObj,
```

```
      fillColor: 'blue'
    })
    return polygon
  },
  openDraw() {
    this.dragging = !this.dragging
    this.polyEditor.close()
    if (this.polygonObj.path && this.polygonObj.path.length >= 3) {
      this.polyEditor.setTarget(this.polygon)
    } else {
      this.polyEditor.setTarget()
    }
    this.polyEditor.open()
  },
  closeDraw() {
    this.dragging = false
    var polygon = this.polyEditor.getTarget()
    this.polygonObj.path = polygon.getPath()
    this.dataForm.area = AMap.GeometryUtil.ringArea(this.polygonObj.path)
    console.log(this.dataForm.area)
    if (this.dataForm.area > 1000) {
      this.dataForm.area = this.dataForm.area / 1000000
          } else {

    }
    this.dataForm.area = this.dataForm.area.toFixed(2)
    var geoinfo = this.getGeoInfo(this.polygonObj)
    this.dataForm.geoinfo = geoinfo
    polygon.remove()
    this.drawPolygon(this.polygonObj)
    this.polyEditor.close()
  },
  drawPolygon(polygonObj) {
    this.polygon = this.createPolygon(polygonObj)
    this.editAmap.add(this.polygon)
  },
  onSearchResult(pois) {
    if (pois.length > 0) {
      this.$nextTick(() => {
        this.$refs.searchBox.keyword = pois[0].name
      })
    }
    // 这边类似模糊查询 会返回一个数组，我就直接取第一个值了。
    this.map.center = [pois[0].lng, pois[0].lat]
  },
  onDialogClose() {
    if (this.editAmap) {
      this.editAmap.destroy()
      this.editAmap = ''
    }
  },
```

```
    // 表单提交
    dataFormSubmitHandle: debounce(function () {
      this.$refs['dataForm'].validate((valid) => {
        if (!valid) {
          return false
        }
        if (this.dataForm.geoinfo == '') {
          this.$messidCard({
            messidCard: '            type: 'error',
            duration: 1000
          })
          return false
        }
        this.$http[!this.dataForm.id ? 'post' :
'put']('/worktarget/twgtworktargetcommonarea/', this.dataForm).then(({ data:
res }) => {
          if (res.code !== 0) {
            return this.$messidCard.error(res.msg)
          }
          this.$messidCard({
            messidCard: this.$t('prompt.success'),
            type: 'success',
            duration: 500,
            onClose: () => {
              this.visible = false
              this.$emit('refreshDataList')
            }
          })
        }).catch(() => { })
      })
    }, 1000, { 'leading': true, 'trailing': false })
  }
}
</script>
<style lang="scss">
.commonarea-edit-mian {
  height: 100%;
  width: 100%;
  display: inline-flex;
}
.commonarea-form-container {
  background-color: white;
  width: 60%;
  display: table-cell;
  padding: 0px 30px 0px 0px;
  .position {
    .el-input {
      width: 80%;
    }
    @media screen and (max-width: 1680px) {
      .el-input {
        width: 70%;
```

```
      }
    }
    @media screen and (max-width: 1366px) {
      .el-input {
        width: 52%;
      }
    }
  }
  .manidCardr {
    .el-input {
      width: 88%;
    }
    @media screen and (max-width: 1680px) {
      .el-input {
        width: 80%;
        right: 0;
        left: 10px;
        float: left;
      }
    }
    .el-button {
      position: absolute;
      right: 0;
    }
  }
}
.commonarea-form-map {
  width: 40%;
  height: 100%;
  display: table-cell;
  .text-btn {
    position: absolute;
    top: 10px;
    right: 0;
    z-index: 10;
  }
}
.dept-list {
  .el-input__inner,
  .el-input__suffix {
    cursor: pointer;
  }
}
.commonarea-edit-map {
  width: 100%;
  height: 65vh;
  .text-btn {
    position: absolute;
    top: 10px;
    right: 0;
  }
```

```
}</style>
        id: 'containers',
        isReal: true,
        isRecord: false,
        userId: 'apidemo',
        userkey,
        userLevel,
        baseURL,
        isSwitchCodetypeOnFullscreen: false      });
      console.log('byskplayer.js version: ' + player.version);
      console.log(player);
      player.setPlayerNum(4);     },
    clickVideo(device, channel) {
      real.open(parseInt(channel), device);
},
    clickAllVideo(device) {
      let videoDeviceIds = device.videodeviceids
      if (videoDeviceIds.length > 0) {
        real.open(2, videoDeviceIds);
      }
      this.openOnClose = videoDeviceIds;
    },
    clickCloseVideo(device) {
      let videoDeviceIds = device.videodeviceids
      if (videoDeviceIds.length > 0) {
        real.close(videoDeviceIds, '1,2,3,4');
        this.openOnClose = videoDeviceIds;
      }
      this.openOnClose = '';
    },
    filterNode(value, data) {
      if (!value) return true;
      return data.label.indexOf(value) !== -1;
    },
    playerNum(i) {
      player.setPlayerNum(i);
    }
  }
}
</script>
<!-- 信号数据可视化管理  -->
<template>
  <div class="stbox">
    <el-row :gutter="23">
      <el-col :span="18">
        <div class="stbgc">
          <el-row :gutter="23">
            <el-col :span="7">
              <el-input size="small" v-model="machineNo" placeholder="请输入信号
数据名称"></el-input>
            </el-col>
```

```
            <el-col :span="7">
              <el-input size="small" v-model="machineNo" placeholder="请输入信号
数据编号"></el-input>
            </el-col>
            <el-col :span="7">
              <el-select size="small" v-model="machineNo" placeholder="请选择信
号数据类型"></el-select>
            </el-col>
            <el-col :span="3" class="stsearch">
              <el-button size="small" type="primary">搜索</el-button>
            </el-col>
          </el-row>
        </div>
      </el-col>
      <el-col :span="6">
        <div class="stbgc">
          <el-row>
            <el-col :span="8" class="text-c">
              <el-radio v-model="type" label="day">日</el-radio>
            </el-col>
            <el-col :span="8" class="text-c">
              <el-radio v-model="type" label="month">月</el-radio>
            </el-col>
            <el-col :span="8" class="text-c">
              <el-radio v-model="type" label="years">年</el-radio>
            </el-col>
          </el-row>
        </div>
      </el-col>
    </el-row>
    <el-row :gutter="23">
      <el-col :span="8" class="text-c">
        <div class="st-gbox">
          <div class="cavasbox" ref="SCEchart"></div>
        </div>
      </el-col>
      <el-col :span="8" class="text-c">
        <div class="st-gbox">
          <div class="cavasbox" ref="SUMEchart"></div>
        </div>
      </el-col>
      <el-col :span="8" class="text-c">
        <div class="st-gbox">
          <div class="cavasbox" ref="ClickEchart"></div>
        </div>
      </el-col>
    </el-row>
        <el-row :gutter="23">
      <el-col :span="8" class="text-c">
        <div class="st-gbox">
          <div class="cavasbox" ref="SCEchart"></div>
        </div>
```

```
      </el-col>
      <el-col :span="8" class="text-c">
        <div class="st-gbox">
          <div class="cavasbox" ref="SUMEchart"></div>
        </div>
      </el-col>
      <el-col :span="8" class="text-c">
        <div class="st-gbox">
          <div class="cavasbox" ref="ClickEchart"></div>
        </div>
      </el-col>
    </el-row>
    <!-- 统计图 -->
    <div>
      <el-row :gutter="23">
        <el-col :span="12" class="text-c">
          <div class="paybox">
            <div class="cavasbox" ref="payEchart"></div>
          </div>
        </el-col>
        <el-col :span="12" class="text-c">
          <div class="paybox">
            <div class="cavasbox" ref="payNumEchart"></div>
          </div>
        </el-col>
      </el-row>
    </div>
  </div>
</template>
<script type="text/ecmascript-6">
import Chart from 'echarts'
export default {
  name: "welcome",
  data() {
    return {
      machineNo: '',
      type: 'day',
      SCEoption: {
        tooltip: {
          trigger: 'item',
          formatter: "{a} <br/>{b}月 : {c}"
        },
        legend: {
          data: [{
            icon: 'rect'
          }],
          top: 1,
          left: 1,
          itemGap: 10,
          itemWidth: 12,
          itemHeight: 12,
```

```
      textStyle: {
        fontSize: 12,
        color: "#323232"
      }
    },
    grid: {
      left: 50,
      right: 10,
      top: 30,
      bottom: 30,
      borderWidth: 1
    },
    xAxis: {
      type: 'category',
      data: ['1', '2', '3', '4', '5', '6', '7', '8', '9', '10', '11', '12'],
      axisLine: {
        lineStyle: {
          color: "#999999",
          width: 1
        }
      },
      axisLabel: {
        margin: 14,
        height: 70,
        interval: 0,
        textStyle: {
          fontSize: 10,
          color: "#999999"
        }
      }
    },
    yAxis: {
      type: 'value',
      axisLine: {
        lineStyle: {
          color: "#999999",
          width: 1
        }
      },
      axisLabel: {
        margin: 14,
        textStyle: {
          fontSize: 10,
          color: "#999999"
        }
      }
    },
        emphasis: {
          color: new Chart.graphic.LinearGradient(
            0, 0, 0, 1,
            [
```

```
                { offset: 0, color: '#2378f7' },
                { offset: 0.7, color: '#2378f7' },
                { offset: 1, color: '#83bff6' }
              ]
            )
          }
        }
      }]
    },
      yAxis: {
        type: 'value',
        axisLine: {
          lineStyle: {
            color: "#999999",
            width: 1
          }
        },
        axisLabel: {
          margin: 14,
          textStyle: {
            fontSize: 10,
            color: "#999999"
          }
        }
      },
      series: [{
        type: 'line',
        barGap: 0,
        data: [50000, 70000, 80000, 40000, 50000, 30000, 40000, 60000, 50000,
40000, 60000, 40000],
        barWidth: 10,
        itemStyle: {
          color: "#108ff9"
        }
      }]
    },
    Clickoption: {
      tooltip: {
        trigger: 'item',
        formatter: "{a} <br/>{b}月 : {c}"
      },
      legend: {
        data: [{
          icon: 'rect'
        }],
        top: 1,
        left: 1,
        itemGap: 10,
        itemWidth: 12,
        itemHeight: 12,
        textStyle: {
```

```
      fontSize: 12,
      color: "#323232"
    }
  },
  grid: {
    left: 50,
    right: 10,
    top: 30,
    bottom: 30,
    borderWidth: 1
  },
  xAxis: {
    type: 'category',
    data: ['1', '2', '3', '4', '5', '6', '7', '8', '9', '10', '11', '12'],
    axisLine: {
      lineStyle: {
        color: "#999999",
        width: 1
      }
    },
    axisLabel: {
      margin: 14,
      height: 70,
      interval: 0,
      textStyle: {
        fontSize: 10,
        color: "#999999"
      }
    }
  },
  yAxis: {
    type: 'value',
    axisLine: {
      lineStyle: {
        color: "#999999",
        width: 1
      }
    },
    axisLabel: {
      margin: 14,
      textStyle: {
        fontSize: 10,
        color: "#999999"
      }
    }
  },
  series: [
    {
      type: 'pie',
      radius: '55%',
      center: ['50%', '50%'],
```

```
        roseType: 'radius',
        label: {
          normal: {
            textStyle: {
              color: 'rgba(255, 255, 255, 0.3)'
            }
          }
        },
        labelLine: {
          normal: {
            lineStyle: {
              color: 'rgba(255, 255, 255, 0.3)'
            },
            smooth: 0.2,
          }
        },
        itemStyle: {
          normal: {
            color: '#c23531',
            shadowBlur: 200,
            shadowColor: 'rgba(0, 0, 0, 0.5)'
          }
        },
        animationType: 'scale',
        animationEasing: 'elasticOut',
        animationDelay: function (idx) {
          return Math.random() * 200;
        }
      }
    ]
  },
  payNumoption: {
    backgroundColor: '#2c343c',
    title: {
      left: 10,
      top: 5,
      textStyle: {
        fontSize: 12,
        color: '#ccc'
      }
    },
    tooltip: {
      trigger: 'item',
      formatter: "{a} <br/>{b} : {c} ({d}%)"
    },
    visualMap: {
      show: false,
      min: 80,
      max: 600,
      inRange: {
        colorLightness: [0, 1]
```

```
          }
        },
        series: [
          {
            type: 'pie',
            radius: '55%',
            center: ['50%', '50%'],
            roseType: 'radius',
            label: {
              normal: {
                textStyle: {
                  color: 'rgba(255, 255, 255, 0.3)'
                }
              }
            },
            labelLine: {
              normal: {
                lineStyle: {
                  color: 'rgba(255, 255, 255, 0.3)'
                },
                smooth: 0.2,
              }
            },
            animationDelay: function (idx) {
              return Math.random() * 200;
            }
          }
        ]
      },
    }
  },
  mounted() {
    this.getSCE()
    this.getSUM()
  },
  methods: {
    getClick() {
      this.chart = Chart.init(this.$refs.ClickEchart)
      this.chart.setOption(this.Clickoption)
    },
    getpay() {
      this.chart = Chart.init(this.$refs.payEchart)
      this.chart.setOption(this.payoption)
    },
    getpayNum() {
      this.chart = Chart.init(this.$refs.payNumEchart)
      this.chart.setOption(this.payNumoption)
    }
  }
};
</script>
```