VUE-SearchBar -->后台系统搜索框基础件

# 功能简介:

## 该基础件主要解决后台搜索框部分功能实现代码重复冗余问题, 目的在于提高前端界面开发效率,针对于搜索

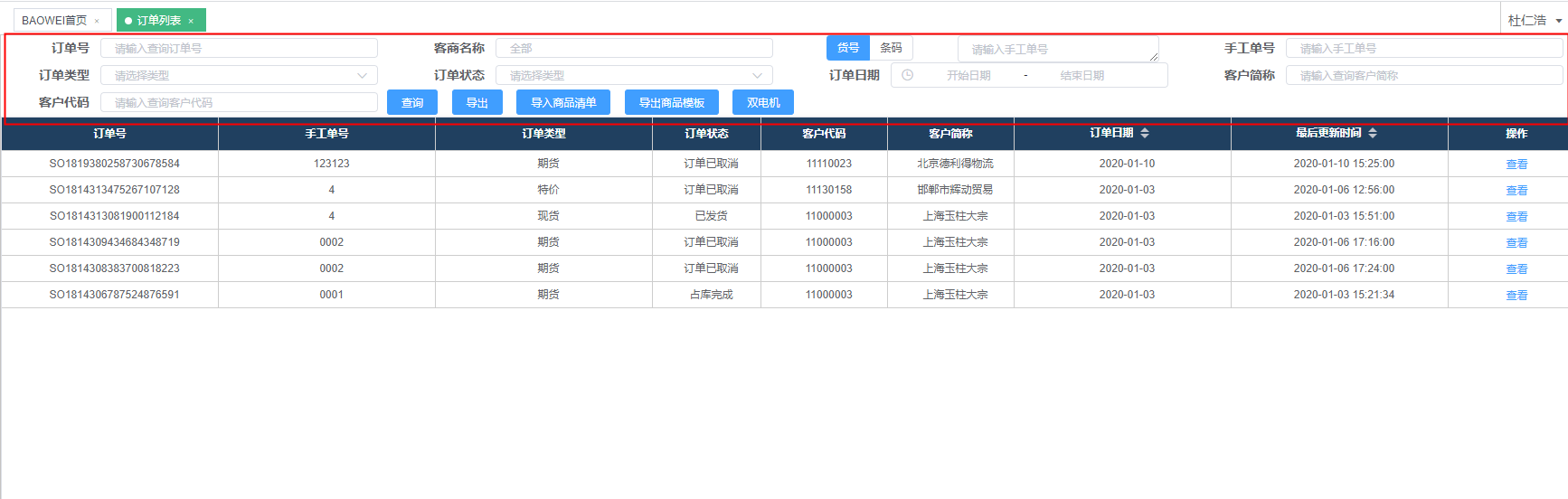
## 框组合,数据传输, 导入, 后台导出/前端模板导出, 操作按钮插入, Select搜索框远程搜索功能实现, 基础下拉选项数据

## 自动匹配赋值, 等等一系列功能. 通过操作文档, 可以在项目中快速,便捷的创建各种搜索类型控件搭配的搜索框, 并且

## 能依赖内部处理的基础件逻辑,  直接获取到按钮动作状态所产生的参数信息. 同时,可以根据个人需求爱好, 通过基础件

## 自身携带的插槽, 设置不同的按钮函数, 来适应多种需求情况.

# 功能效果预览



## git 地址:<https://github.com/hejiyun/vue-SearchBar-common>

## 执行命令 `git clone [git@github.com](mailto:git@github.com):hejiyun/vue-SearchBar-common.git`

## 基础件操作手册

# 基础件的使用

# a. 首先将该基础件放在项目src目录下的component/component→common文件夹中

# 

# b. 在需要使用的组件中, 引入, 并使用

# 引入

# 

# 两种方式是一样的, 不过推荐使用@方式, 这种在引用路径时不容易, 需要在webpack中resolve解析规则中配置指代路径.

# 

# 使用

# 首先注册子组件--也就是这个基础件

# 

# 然后在template结构中直接使用对应标签

# 

# 基础件布局配置

# 基础件的最基本配置, 必须要有options, 基础件会通过这个传值去创建相应的搜索项,

# 可选布局配置:

# a.  width-number:  width-number 对应的是每一个搜索选项占用的空间, 采用vue栅格机制, 一行分为24. 它定义了一行可以占有几个搜索选项,  默认为8,即, 一行占有三个搜索选项, 设置为6,则一行占有4个

# 

# 

# b. label-width: 对应的是每一个搜索选项的标题所占用当前搜索项的宽度,  默认为110px, 作用同element

# 搜索选项的创建与基本配置

# 在基础件标签中, 若需要使用该基础件, option属性为必传属性,它的作用是基础件会通过这个属性去创建相应的搜索项

# 目前支持的搜索项类型,  其中type为必须属性, type定义支持的类型为以下几种

# a.input类型

# 

# 该类型配置很简单, type定义需要生成的搜索项类型, name定义当前需要创建的搜索项对应的参数属性, 也就是当点击查询时,获取搜索参数集合时所对应的属性及值

# placeholder 定义输入框的提示内容, label 定义 搜索项对应的标签

# 

# b. check类型,

# 

# 该类型的配置也很简单, 没有label, placeholder对应多选框后面的文字内容,目前搜索框只支持单个多选框配置(主要考虑多个一般不会有这种场景)

# 

# c. 时间选择器,- 单日期时间类型

# 

# 该类型的配置也比较简单,  defaultTime可以设置默认的时间, 其他的设置属性都是相同定义

# valueFormat属性定义传出参数的格式, 默认为'yyyy-MM-dd HH:mm:ss'

# 

# d. 时间选择器–日期时间类型

# 

# 该类型配置, 默认支持, 近三小时, 今天, 近一周, 近一月, 近三月, 后续会将此处可选化

# defaultValue 定义初始化默认值,  valueFormat定义时间参数传出的格式, 默认为'yyyy-MM-dd HH:mm:ss'

# 

# e. 下拉框- 单选下拉框  ,  由于逻辑及功能效果不同,  此处下拉类型, 由两种类型定义, 一种select, 不做远程搜索功能, 一种lazySelect,做远程搜索功能,

# 

# 该类型配置简单, 用法同element,   options定义接收的数据源信息

# 

# f.下拉框-多选下拉框

# 

# 该类在单选下拉的基础上增加multiple的属性,完成多选类型转换, 自动添加全部选项及对应内容逻辑.

# 

# g. 下拉框–远程搜索单选

# 

# 远程搜索控件中, 包含了单选, 多选, 这些功能

# nocheck是有无多选框,  multiple支持多选,  去掉filterable可单纯使用单选功能,

# noHeader, 支持是否在下拉框第一行显示提示标题文字.

# className: 可以为控件添加类名,

# optionHeader: 下拉框首行提示信息, 不填默认为 值, 标题

# optionHeaderCode: 下拉框每一行显示对应的属性字段, 不填默认为value, label

# optionFilterRole: 下拉框远程搜索时查找匹配的属性, 默认是value,和label其中任意一个包含输入的字符, 都筛选出来

# optionHeaderWidthSpan: 下拉框每一项所占一整行的宽度

# changePlaceholder: 下拉框聚焦时,提示内容替换的字符串内容, 默认为全部

# optionFilterList: 当前下拉框过滤出来的数组

# optionAllList: 下拉框数组的全部信息

# 

# h. 下拉框, 远程搜索多选

# 

# 与单选相比, 只需要增加mutiple属性即可.

# i. 下拉框- 远程搜索首行提示 单选,与多选, 不多细说了, 跟前两个差别不大

# 

# 只需将noHeader属性去掉, 就可以添加对应的首行信息提示

# 

# j. 下拉框- 远程搜索首行提示多条目

# 

# 多条目展示,  调整optionHeader和optionHeaderCode即可添加并获取对应值展示,

# 需要注意的是, optionHeaderWidthSpan 需要按照添加的条目添加对应宽度, 会超出省略

# 

# k. 下拉框- 远程搜索多选框组合

# 

# 将noCheck属性去掉, 就可以实现多选框的添加组合, 并且在多选框出现时, 会自动去除选中后, 项尾自带√符号

# 

# L. 文本域-- 按钮文本域组合

# 

# 该控件属性, checkValue为默认选中按钮属性, radioList,为左侧按钮的渲染个数, 建议为两个, 如果过多, 可能会出现样式问题.

# 该控件, 支持在文本域聚焦时,文本域层级变高, 在文本域失焦时, 文本域层级回归.

# 

# 

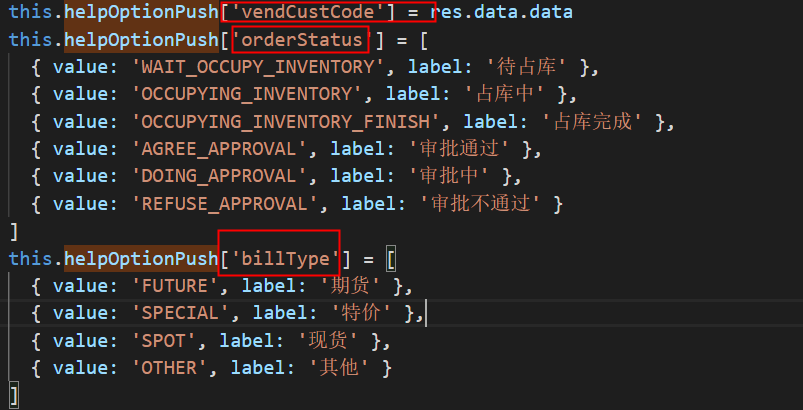
# 

# 4.下拉选项数据源的赋值与传递

## 如果在使用基础件中, 同步出现多个下拉框的使用, 那么, 对于每个下拉框中的数据源,  只需要给基础件传值 helpOptionPush集合对象, 其中每一个属性定义各个属性的下拉框数据源

## 使用, 根据下拉框的多少, 为helpOptionPush对象添加相应个数的属性, 属性名为每个下拉框的name,  值为当前下拉框的数据源. 基础件会根据属性名自动协助添加对应下拉框的数据源

## 远程搜索框会自动初始化同步filter和all的数据



# 5.基础件默认按钮功能



## 基础件默认按钮有五个, 分别是查询, 重置, 导出, 导入, 导出模板

## 其中, 重置, 导出, 和导出模板 无通信函数,  内部逻辑自执行

## 查询与导入按钮 有通信函数, 在基础件标签中, 通过@search和@import 进行操作

## search参数返回为当前选择的params, import无参数返回



## 显示默认按钮:

## is-import  : 显示导入按钮,  import-excel-text 设置当前按钮名称

## is-model : 显示导出模板按钮  model-export-name 设置当前导出模板文件名称,  model-text: 设置当前按钮名称,  model-export-options 设置当前导出模板的首行字段

## is-search:  显示查询按钮, 默认按钮名称

## is-reset: 显示重置按钮, 默认按钮名称

## is-export: 显示导出按钮   export-options设置导出时调用的接口及参数信息

## url为调用地址,   data为当前请求参数,  method为请求方式, mergeParams为合并搜索项参数,  timeout设置超时时间, 默认一分钟.



# 6.按钮的自定义插入

         按钮的自定义插入是使用了vue中的插槽,  具体使用如下

           在searchBar标签内, 使用template进行按钮插入, slot对应名称btn, 定义为按钮类型插入,  slot-scope接收子组件传值,  它的params属性就是基础件内部的当前选择的搜索项参数

            按钮方面可以根据个人喜好增加, 也可以不用默认按钮, 自己设置按钮都可以



# 7.基础件的方法调用

## 1.  getParams() :   获取当前基础件搜索项参数, 返值为params

## 2.  resetForm() :   重置搜索项