

## 字典翻译

字典目前使用mysql+redis 存储，优先使用缓存。

前端应用：前端需要在页面加载的时候请求本页面需要使用的字典。

```
import { initDictFromRedis, transDict } from '@/utils/dict'

data() {
  return {
    dict: {},          # 放置字典返回类容
    dictKeys: 'guest_type, sex, cert_type'  #需要请求的字典，使用逗号分隔
  }
}

mounted() {
  const that = this
  initDictFromRedis(that)
}

#字典翻译
methods: {
  transDict(key, value) {
    const that = this
    return transDict(that, key, value)
  }
}
```

```
#调用以证件类型为例
<el-col :xs="24" :sm="12">
  <div class="view-item">
    <span>{{ '证件类型' + ': ' }}</span> {{
transDict('cert_type',detail.certType) }}
  </div>
</el-col>
```

新增（修改）字典以后，需要将字典刷入缓存，目前是手动操作，简单做了，后面调整。

系统主页 | ● 字典管理 x

类型	描述	搜索	重置	更多操作
<input type="checkbox"/>	类型	描述	备注信息	添加 删除 刷新缓存 创建时间
<input type="checkbox"/>	rangeType	通知公告范围类型	通知公告范围类型	24 15:08:43
<input type="checkbox"/>	publishStatus	发布状态	发布状态	2020-11-24 15:05:09

## 表格--》字典filter

效果

小区门名称	访客姓名	性别	手机号	
南大门	张三丰	<input type="checkbox"/> 男 <input type="checkbox"/> 女 <input type="checkbox"/> 保密	12566667777	20

10条/页 < 1 > 筛选 重置

HTML

```
<el-table-column
  :label="$t('propertyManager.guestManager.sex')"
  :filters="initFilterArray('sex')"
  :filter-method="filterSex"
  class-name="status-col"
>
  <template slot-scope="{row}">
    <el-tag :type="row.sex | sexFilter">
      {{ transDict('sex', row.sex) }}
    </el-tag>
  </template>
</el-table-column>
```

数据源

过滤函数

函数

```
import { initFilterArray } from '@/utils/dict'

methods: {
  initFilterArray(key) {
    const that = this
    return initFilterArray(that, key)
  },
  filterSex(value, row) {
    return row.sex === value //这里的sex,是你的字段,可能叫guest_sex
  }
}
```

dict.js

```
export function initDictFromRedis(that) {
  that.dict = {}
  that.loading = true
  request.get('system/dictItem/getHashByKey', {
    dictKeys: that.dictKeys
  }).then((r) => {
```

```
    const data = r.data.data
    that.dict = data
    that.loading = false
  })
}
export function initFilterArray(that, key) {
  if (Object.keys(that.dict).length !== 0) {
    const arr = []
    Object.keys(that.dict[key]).map((x) => {
      const obj = { text: that.dict[key][x], value: x }
      arr.push(obj)
    })
    return arr
  }
}
export function transDict(that, key, value) {
  if (Object.keys(that.dict).length !== 0) {
    return that.dict[key][value]
  }
}
function handleObjectToArray(dict) {
  if (Object.keys(dict).length !== 0) {
    const arr = []
    Object.keys(dict).map((x) => {
      const obj = { description: dict[x], value: x }
      arr.push(obj)
    })
    return arr
  }
}
```