典型场景开发样例

本文档描述了在特殊场景中，后端开发的样例，以及注意事项

Autocomplete控件

后端代码：

```
        /**
     *
     *
     * @param queryVO AutoCompleteQueryVO
     * @return List
     */
    @ApiOperation("cityCode/")
    @ApiImplicitParam(name = "SimpleQueryVO", value = "", required =
false, dataType = "String")
    @GetMapping("/masCitys/simple")
    List<SimpleCityEntity> findSimpleCity(@RequestParam Map<String,
Object> queryVO);
        SimpleCityEntitycityCodecityNameentityName
        Map




@Override
    public List<SimpleEntity> findSimpleCity(Map<String, Object> queryVO) {
        return this.service.findSimpleCity(AutoCompleteQueryVO.by
(queryVO)).stream().map(s -> {
            SimpleEntity e = new SimpleEntity();
            e.setCityId(s.getCityId());
            e.setCityCode(s.getCityCode());
            e.setEntityName(MessagesResourceHelper.isEnLanguage() ? s.
getCityEnglishName()
                    : s.getCityChineseName());
            return e;
        }).collect(Collectors.toList());
    }
controller
serviceAutoCompleteQueryVO.
```

```
/**
     * city.
     *
     * @param queryVO DevQueryVO
     * @return Page<MasCity>
     */
    @SuppressWarnings("unchecked")
    public List<MasCity> findSimpleCity(AutoCompleteQueryVO queryVO) {
        ArafQueryWrapper<MasCity> q = new ArafQueryWrapper<>();
        //
        this.autocompleteQuery(q, queryVO);
        //
        // queryValuequeryFiled
        //
        // autocompleteagentCodeagentName
        //


        q.or();

        // namename
        if (MessagesResourceHelper.isEnLanguage()) {
            q.like(!ArafStringUtils.isEmpty(queryVO.getQueryValue()),
"cityEnglishName",
                    queryVO.getQueryValue());
        } else {
            q.like(!ArafStringUtils.isEmpty(queryVO.getQueryValue()),
"cityChineseName",
                    queryVO.getQueryValue());
        }

        return getDao().selectPage(new Page(0, queryVO.getTop()), q).
getRecords();
    }
serviceAutoCompleteQueryVO
```

DataGrid分组功能

后端代码：

```
voGroupData
public class MasSfcDTO extends GroupData implements Serializable




ctrollerservice
 @Override
    public Page<MasSfcVO> findAll(DevQueryVO vo) {
        return this.service.findAllGroupBy(vo);
    }




service
/**
     * groupby
     *
     * @param vo DevQueryVO
     * @return Page<MasStcVO>
     */
    @SuppressWarnings("unchecked")
    public Page<MasSfcVO> findAllGroupBy(DevQueryVO vo) {
        return PageHelper.convertToPage(
                this.masSfcDao.findAll(PageHelper.generatePage(vo),
generateGroupSelect(vo),
                        CriteriaHelper.parseQuery(vo, new
ArafQueryWrapper(), MasStcVO.class)),
                this.wrapperCount(vo), vo);
    }




daosqlsqlgroupby lolumn
/**
     * groupby
     *
     * @param page page
     * @param selectColumns selectColumns
     * @param queryWrapper queryWrapper
     * @return Page<MasStcVO>
     */
    @Select("select ${selectColumns} from mas_sfc ${ew.customSqlSegment}")
    IPage<MasSfcVO> findAll(IPage<MasSfcVO> page, @Param("selectColumns")
String selectColumns,
                           @Param(Constants.WRAPPER) Wrapper queryWrapper);
```

导出全部数据功能

> 使用基类的findall方法，不需要自定义。
>
> 如果是自定义的联合查询，参考下面的例子。

```
serivce
/**
     *
     *
     * @param vo DevQueryVO
     * @return Page<MasAreaVO>
     */
    @SuppressWarnings({ "rawtypes" })
    public Page<MasAirportVO> findAllVO(DevQueryVO vo) {
        // .
        if (vo.isExportAll()) {
            //
            return PageHelper.convertToPage(getDao().findAllJoin
(PageHelper.generatePage(0, Integer.MAX_VALUE),
                    CriteriaHelper.parseQuery(vo, new ArafQueryWrapper(),
MasAirportVO.class)));
        }
        return PageHelper.convertToPage(getDao().findAllJoin(PageHelper.
generatePage(vo),
                CriteriaHelper.parseQuery(vo, new ArafQueryWrapper(),
MasAirportVO.class)));
    }


dao
/**
     *
     *
     * @param page IPage
     * @param queryWrapper Wrapper
     * @return IPage<MasAreaVO>
     */
    @SuppressWarnings("rawtypes")
    @Select("select ap.*, ct.city_chinese_name,"
            + " ct.city_english_name, ct.city_type,  ct.country_code, "
            + " ct.latitude, ct.longitude " + " from mas_airport ap left
join mas_city ct on "
            + " ap.city_code = ct.city_code" + " ${ew.customSqlSegment}")
    IPage<MasAirportVO> findAllJoin(IPage<MasAirportVO> page,
                                    @Param(Constants.WRAPPER) Wrapper
queryWrapper);
```

关联表查询

使用dev控件，后端是多表关联的情况

```
/**
     * .  API
     *
     * @param vo DevQueryVO
     * @return Page<MasNationalCurrencyVO>
     */
    @ApiOperation("")
    @ApiImplicitParam(name = "DevQueryVO", value = "dev", required =
false, dataType = "String")
    @GetMapping("/masNationalCurrencys")
    Page<MasNationalCurrencyVO> findAll(DevQueryVO vo);


===========================================================================
==========
Controller


@Override
    public Page<MasNationalCurrencyVO> findAll(DevQueryVO vo) {
        return this.getService().findAllVO(vo);
    }


===========================================================================
==========
servicefindallBaseServicefindalldaobasedao


/**
     *
     *
     * @param vo DevQueryVO
     * @return Page<MasNationalCurrencyVO>
     */
    @SuppressWarnings({ "rawtypes" })
    public Page<MasNationalCurrencyVO> findAllVO(DevQueryVO vo) {
        // .
        if (vo.isExportAll()) {
            //
            return PageHelper.convertToPage(getDao().findAllJoinList
(CriteriaHelper.parseQuery(vo,
                    new ArafQueryWrapper(), MasNationalCurrencyVO.class)));
        }
        return PageHelper
                .convertToPage(getDao().findAllJoin(PageHelper.generatePage
(vo), CriteriaHelper
                        .parseQuery(vo, new ArafQueryWrapper(),
MasNationalCurrencyVO.class)));
```

```
    }


===========================================================================
===============
daosqlwhere${ew.customSqlSegment}servicewrapper


 /**
     * List
     *
     * @param queryWrapper Wrapper
     * @return List<MasNationalCurrencyVO>
     */
    @SuppressWarnings("rawtypes")
    @Select("select a.*,b.english_name as country_region_name "
            + "from mas_national_currency a left join mas_country_region
b  "
            + "on a.country_region_code = b.country_region_code " + " ${ew.
customSqlSegment}")
    List<MasNationalCurrencyVO> findAllJoinList(@Param(Constants.WRAPPER)
Wrapper queryWrapper);




===========================================================================
================
vovovodomianmodelvovoQueryAliascolumn


@Data
@Accessors(chain = true)
@ApiModel(value = "MasNationalCurrency", description =
"MasNationalCurrencyVO")
@EqualsAndHashCode(callSuper = false)
public class MasNationalCurrencyVO implements Serializable {

    private static final long serialVersionUID = 1L;

    @ApiModelProperty(value = "uuid")
    private String masNationalCurrencyId;

    @ApiModelProperty(value = "/")
    @NotBlank
    @QueryAlias(value = "a")
    private String countryRegionCode;

    @ApiModelProperty(value = "")
    @NotBlank
    private String currencyCode;
    @ApiModelProperty(value = "")
    @NotNull
    @QueryAlias(value = "a")
    private Date effectiveDate;

    @ApiModelProperty(value = ", S/H")
    @QueryAlias(value = "a")
    private String softHardCurrencyMark;

    @ApiModelProperty(value = ",U:USD,E:EUR")
    private String accountMoneyMark;

    @ApiModelProperty(value = "/")
    @QueryAlias(value = "b", columnAlias = "englishName")
    private String countryRegionName;

    @ApiModelProperty(value = "")
    @QueryAlias(value = "a")
    private String createdBy;

    @ApiModelProperty(value = "")
    @QueryAlias(value = "a")
    private Date createdDate;

    @ApiModelProperty(value = "")
    @QueryAlias(value = "a")
    private String lastModifiedBy;

    @ApiModelProperty(value = "")
    @QueryAlias(value = "a")
    private Date lastModifiedDate;

}
```

异步导出全部数据功能

前台

html

```html
<!-- onRowPrepared -->
<dx-data-grid XXX (onRowPrepared)="onRowPrepared($event)">
</dx-data-grid>
```

```ts
//
gridInfo = new ArafGridExportInfo();


ngOnInit() {

  //exportBtnexportAllAsyncBtngridInfo
  //buttons,null
  this.arafCommonGrid.initToolbar(this.grid, this.getGridBtns(), {
      exportBtn: { visible: false },
      exportAllAsyncBtn: { visible: true }
    }, this.gridInfo);

  //gridurl
  this.gridInfo.requestUrl = this.ipUrl + masUrlConfig.
Mas_Flights_Export_All;

  //gridobj(ArafDataSource)onLoadingProcess
  //bind thisthis.arafBaseGridService.registerDataSource
  this.obj.onLoadingProcess = this.onGridLoadingProcess.bind(this);
}


/**
 * gridloading
 * @param loadOptions
 */
onGridLoadingProcess(loadOptions) {
  //gridloadOptionshttp param
  this.gridInfo.loadOptions =
  this.arafCommonService.getExportAllAsyncLoadOptions(loadOptions, this.
obj.searchFilter);
}


/**
 * gridrow
 * @param e
 */
onRowPrepared(e) {
  //gridgrid/rows
  //grid
  this.gridInfo.rows = this.arafCommonService.getExportAllAsyncRows(e,
this.gridInfo.rows);
}
```

后台

定义API接口POST请求

查询数据vo 可以是DevQueryVO类型也可以是BaseQueryVO类型

ArafExportInfo export 是POST 参数类型，前端传递导出标题信息。

```
/**
*
* @param vo DevQueryVO
* @param export ArafExportInfo
* @return boolean
*/
@ApiOperation(".")
@PostMapping("/userFeaturesTrack/exportAll")
Result<String> exportAll(DevQueryVO vo, @RequestBody ArafExportInfo
export);
```

Controller实现API接口

方式1、可以直接调用 AbstractRestController的asyncExport方法进行全部导出。

```
/** */
@Autowired
private SysConfigApi sysConfigApi;
@Override
public Result<String> exportAll(DevQueryVO vo, ArafExportInfo export) {
    return asyncExport(vo, export, c -> getService().countAll(vo), x ->
getService().findAll((DevQueryVO) x), sysConfigApi.getTempFilePathPrefix().
getPayload());
}
```

方法2、通过继承或实现

```
private exportFileService


long count =  getService().countAll(vo)
if(count > MAX_EXCEL_SIZE) {
    throw new BusinessException("excel");
}
exportFileService.exportFile(vo, info, PrincipalContext.getCurrentUser(), 
count, f, tempFilePath);
return Result.of("");
```

扩展点

1、数据的自定义显示。

导出的数据需要根据数据字典进行格式化显示。可以在

`x -> getService().findAll((DevQueryVO) x).map(o -> )`

方法中进行处理。

2、导出的数据需要联合查询。

`x -> getService().findAll((DevQueryVO) x)`

自定义查询即可，返回List<?>，自动根据前台传入的展示字段匹配查询结果中的属性值

接口调用方式

POST → {{PATH}}/async/export?sort=%5B%7B%22selector%22:%22sfcCode%22,%22desc%22:false%7D,%7B%22selector%22:%22subSfcCode%22,%22desc%22:false%7D,%7B%22selector%22:%22effectiveDate%22,%22desc%22:false%7D,%7B%22selector%22:%22expireDate%22,%22desc%22:false%7D%5D

body 参数

{"rows":[{"columns":[{"alignment":"left","caption":"航班号","dataField":"flightNumeric","dataType":"string"},{"alignment":"left","caption":"飞行日期","dataField":"flightDate","dataType":"date","format":"yyyy-MM-dd"},{"alignment":"left","caption":"计划起飞日期","dataField":"plannedDepartureDate","dataType":"date","format":"yyyy-MM-dd"},{"alignment":"left","caption":"计划起飞时间","dataField":"plannedDepartureTime","dataType":"datetime","format":"HH:mm"},{"alignment":"left","caption":"计划到达日期","dataField":"plannedArrivalDate","dataType":"date","format":"yyyy-MM-dd"},{"alignment":"left","caption":"计划到达时间","dataField":"plannedArrivalTime","dataType":"datetime","format":"HH:mm"},{"alignment":"left","caption":"航线代号","dataField":"routeCode","dataType":"string"},{"alignment":"left","caption":"去程/回程","dataField":"goBackFlag","dataType":"string"},{"alignment":"left","caption":"航站1","dataField":"airport1","dataType":"string"},{"alignment":"left","caption":"航站2","dataField":"airport2","dataType":"string"},{"alignment":"left","caption":"航班全程","dataField":"route","dataType":"string"},{"alignment":"left","caption":"承运人","dataField":"carrier","dataType":"string"},{"alignment":"left","caption":"飞机注册号","dataField":"aircraftRegister","dataType":"string"},{"alignment":"left","caption":"机型","dataField":"aircraftType","dataType":"string"},{"alignment":"left","caption":"飞机所属子公司","dataField":"aircraftSubCompany","dataType":"string"},{"alignment":"left","caption":"执行任务分公司","dataField":"flightCompany","dataType":"string"},{"alignment":"left","caption":"执行任务飞行队","dataField":"flightTeam","dataType":"string"},{"alignment":"left","caption":"航线全程","dataField":"route","dataType":"string"},{"alignment":"left","caption":"飞行任务（客）","dataField":"flightTaskP","dataType":"string"},{"alignment":"left","caption":"飞行任务（货）","dataField":"flightTaskC","dataType":"string"},{"alignment":"right","caption":"旅客人数","dataField":"passengerCount","dataType":"number"},{"alignment":"right","caption":"F舱人数","dataField":"fClassPax","dataType":"number"},{"alignment":"right","caption":"C舱人数","dataField":"cClassPax","dataType":"number"},{"alignment":"right","caption":"Y舱人数","dataField":"yClassPax","dataType":"number"},{"alignment":"right","caption":"其他主舱人数","dataField":"otherClassPax","dataType":"number"},{"alignment":"left","caption":"原始航班性质","dataField":"originaliFlightType","dataType":"string"}]}],"fileName":"航班信息"}

默认会通知当前用户，在线消息及email消息。