**IUAP平台使用指南**

**开发环境安装**

用友网络科技股份有限公司

二○一八年五月

**此文档为保密文档，未经用友软件股份有限公司书面同意，不得向任何单位或个人提供、转让本文档中的任何内容，用友软件股份有限公司将保留对泄漏文档内容的起诉权利。**

**IUAP平台使用指南**

|  |  |  |
| --- | --- | --- |
| 文件状态：  [ ] 草稿  [**√**]正在修改  [] 正式发布 | 文件标识： |  |
| 当前版本： |  |
| 作者： |  |
| 完成日期： |  |

版本历史

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 版本 | 作者 | 参与者 | 起止日期 | 备注 |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

审核记录

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 版本 | 姓名 | 职位 | 审阅日期 | 备注 |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

目 录

[第一章 概述 1](#_Toc515562421)

[1.1 业务场景 1](#_Toc515562422)

[1.2 开发方案 1](#_Toc515562423)

[第二章 示例配置 2](#_Toc515562424)

[2.1 模板文件配置 2](#_Toc515562425)

[2.2 数据库配置 2](#_Toc515562426)

[2.3 开发环境配置 3](#_Toc515562427)

[2.4 功能验证 3](#_Toc515562428)

[2.5 常见问题 5](#_Toc515562429)

[第三章 代码开发示例 6](#_Toc515562430)

[3.1 后端开发 6](#_Toc515562431)

[3.1.1 编写entity文件 6](#_Toc515562432)

[3.1.2 编写dao文件 7](#_Toc515562433)

[3.1.3 编写service文件 7](#_Toc515562434)

[3.1.4 编写controller文件 10](#_Toc515562435)

[3.1.5 编写xml文件 12](#_Toc515562436)

[3.2 前端开发 12](#_Toc515562437)

[3.2.1 编写meta.js 12](#_Toc515562438)

[3.2.2 编写template.html文件 12](#_Toc515562439)

[3.2.3 编写template.css文件 14](#_Toc515562440)

[3.2.4 编写template.js文件 14](#_Toc515562441)

# 概述

## 业务场景

开发者经常遇到需要批量上传记录，将记录导出成excel的需求， IUAP内置了一些对于excel模板的操作，开发者只需编写少量的代码即可方便快捷地实现这类需求。

* 批量导入
* 批量导出

## 开发方案

* 首先构建一张测试表用来测试利用模板实现批量导入，数据导出的功能。测试表为“example\_template”，表结构如下。

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段 | 名称 | 类型 | 主键 | 备注 |
| Id | 主键 | Varchar | 是 | Uuid |
| Code | 编码 | Varchar |  | 非空，唯一索引 |
| Name | 名称 | Varchar |  | 可空 |
| Remark | 备注 | Varchar |  | 可空 |

* 设计示例功能。
* 模板下载
* 导入数据
* 导出数据

# 示例配置

## 数据库配置

在IUAP的数据库中创建示例表“example\_attachment”。

**CREATE TABLE `example\_template` (**

**`id` varchar(36) COLLATE utf8\_unicode\_ci NOT NULL,**

**`code` varchar(50) COLLATE utf8\_unicode\_ci DEFAULT NULL,**

**`name` varchar(50) COLLATE utf8\_unicode\_ci DEFAULT NULL,**

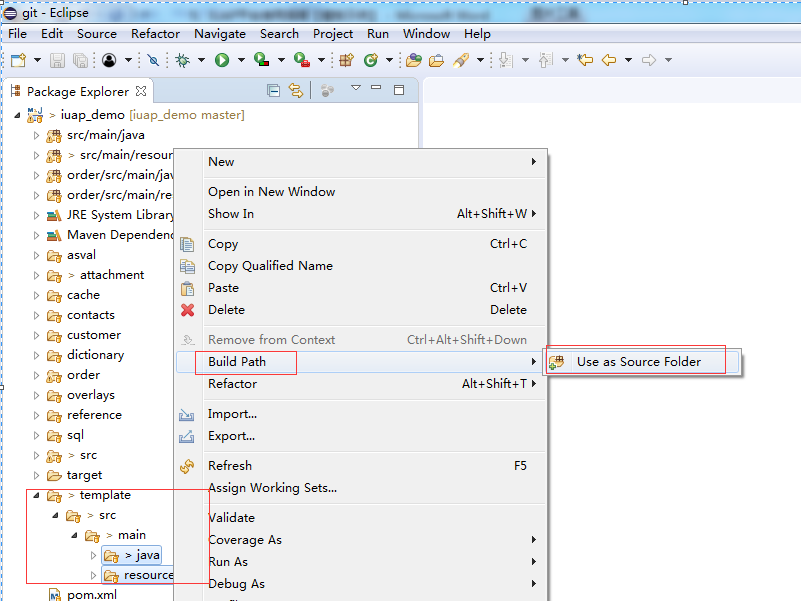
**`remark` varchar(50) COLLATE utf8\_unicode\_ci DEFAULT NULL,**

**PRIMARY KEY (`id`)**

**) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8\_unicode\_ci;**

## 开发环境配置

用eclipse载入示例工程，选中template示例下的java和resources目录，然后右键—》Build Path—》Use as Source Folder，将示例工程载入编译环境。



检查项目无问题后，右击iuap\_demo –》Run As —》run on server。

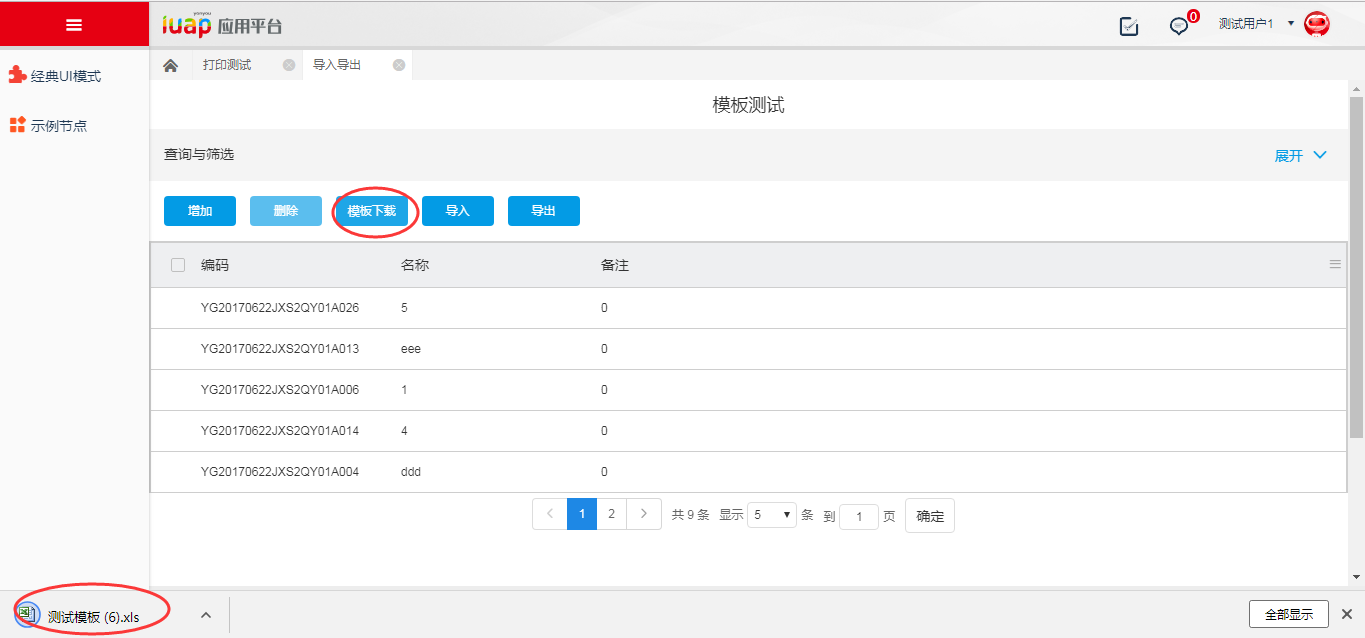
## 功能测试

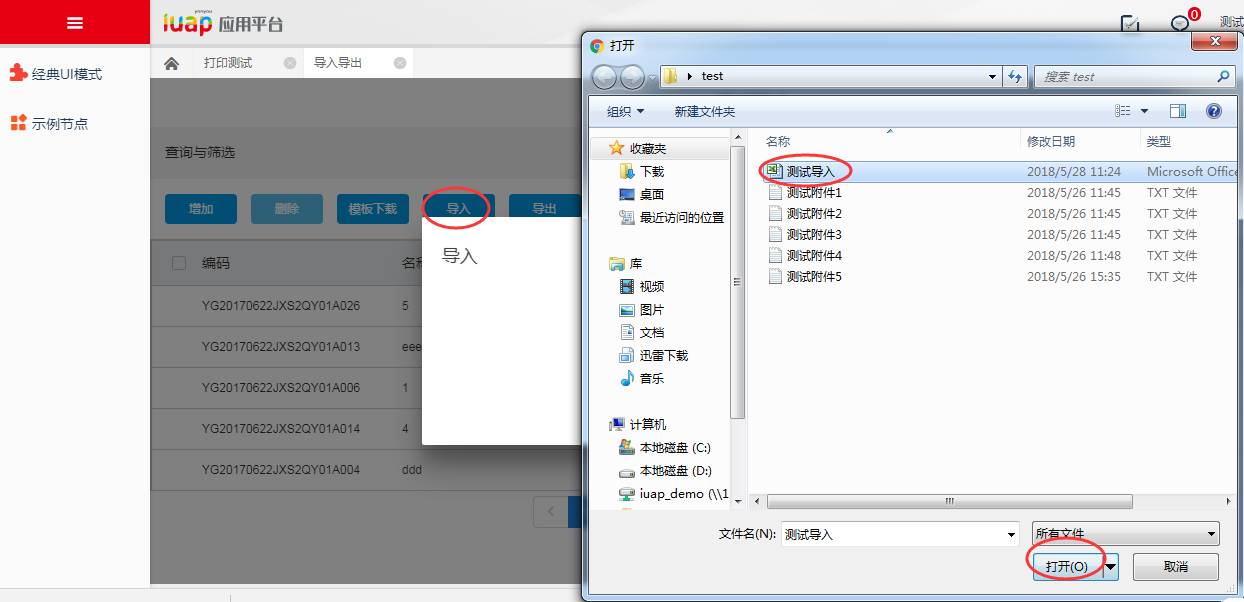
### 菜单注册

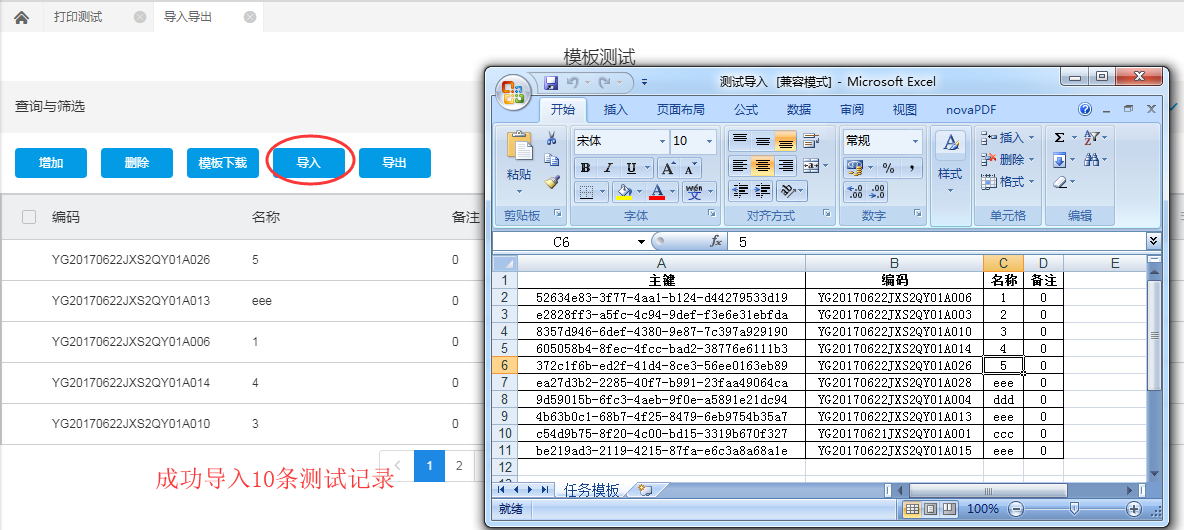
手工进行菜单注册，为标准方法，不重复介绍。

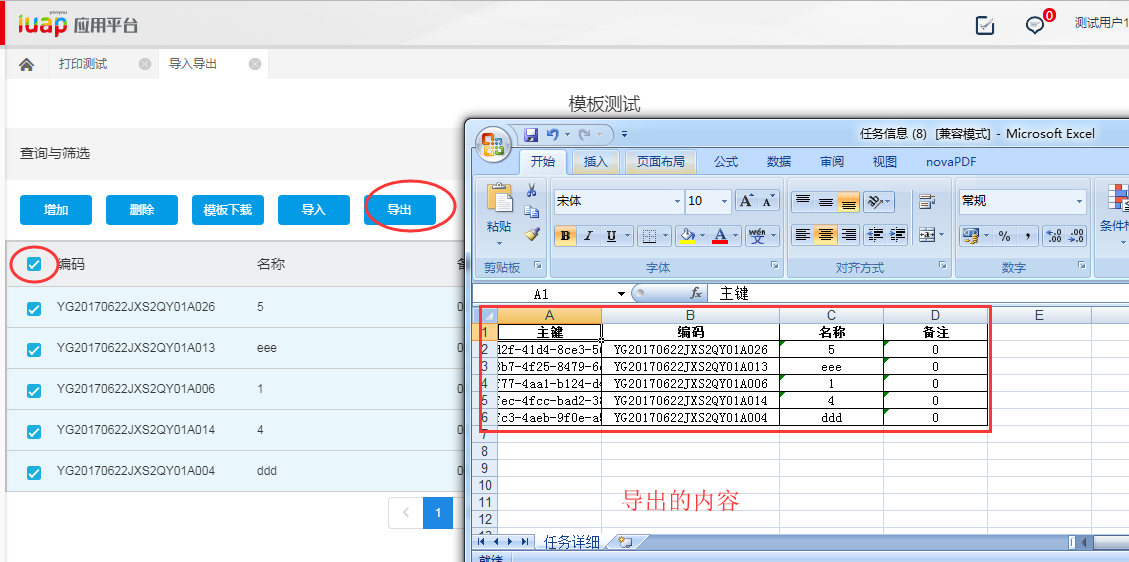
功能节点设置js地址为：/iuap-example/pages/template/template.js。

### 功能验证









# 代码开发示例

## 后端开发

### 编写entity文件

ExampleTemplate需要继承BaseEntity，并重写getMetaDefinedName()和getNamespace()

**package** com.yonyou.iuap.example.template.entity;

**import** com.yonyou.iuap.persistence.vo.BaseEntity;

**public** **class** ExampleTemplate **extends** BaseEntity {

**private** String id;

**private** String code;

**private** String name;

**private** String remark;

**public** ExampleTemplate() {}

**public** String getId() {

**return** **this**.id;

}

**public** **void** setId(String id) {

**this**.id = id;

}

**public** String getCode() {

**return** **this**.code;

}

**public** **void** setCode(String code) {

**this**.code = code;

}

**public** String getName() {**return** **this**.name;}

**public** **void** setName(String name) {**this**.name = name;}

**public** String getRemark() {**return** **this**.remark;}

**public** **void** setRemark(String remark) {**this**.remark = remark;}

@Override

**public** String getMetaDefinedName() {

**return** "example\_template";

}

@Override

**public** String getNamespace() {

**return** "iuap\_qy";

}

}

### 编写dao文件

导入文件有可能是新增，有可能是更新，需要先取出所有的id，再判定导入的文件记录是新增还是更新，所以编写了List<String> getIds()方法。

**package** com.yonyou.iuap.example.template.dao;

**import** com.yonyou.iuap.example.template.entity.ExampleTemplate;

**import** com.yonyou.iuap.mvc.type.SearchParams;

**import** com.yonyou.iuap.mybatis.type.PageResult;

**import** com.yonyou.iuap.persistence.mybatis.anotation.MyBatisRepository;

**import** org.apache.ibatis.annotations.Param;

**import** org.springframework.data.domain.PageRequest;

**import** java.util.List;

@MyBatisRepository

**public** **interface** ExampleTemplateMapper {

**int** insert(ExampleTemplate record);

**int** update(ExampleTemplate record);

**int** insertSelective(ExampleTemplate record);

**int** updateByPrimaryKeySelective(ExampleTemplate record);

**int** updateByPrimaryKey(ExampleTemplate record);

**int** deleteByPrimaryKey(String pk);

ExampleTemplate selectByPrimaryKey(String pk);

PageResult<ExampleTemplate> getByIds(List<String> ids);

List<String> getIds();//或得所有id集合

PageResult<ExampleTemplate> selectAllByPage(@Param("page") PageRequest pageRequest, @Param("search") SearchParams searchParams);

**void** batchInsert(List<ExampleTemplate> addList);

**void** batchUpdate(List<ExampleTemplate> updateList);

**void** batchDeleteByPrimaryKey(List<ExampleTemplate> list);

}

### 编写service文件

* 定义存储模板头对象，做字段与名称的映射。

**private** **static** String *values* = "{'id':'主键','code':'编码','name':'名称','remark':'备注'}";

**private** Map<String, String> headInfo;

* 下载模板

ExcelExportImportor.downloadExcelTemplate(response, getHeadInfo(), "任务模板", "测试模板"); 注意传入的参数要与模板定义的一致。

* 导入数据

先要查询出所有的id，List<String> ids = mapper.getIds();然后判定是新增还是修改。

* 导出数据

先根据选中的id查询实体数据，然后导出。

@Service

**public** **class** ExampleTemplateService **extends** ParentService<ExampleTemplate> {

@Autowired

ExampleTemplateMapper mapper;

**public** Page<ExampleTemplate> selectAllByPage(PageRequest pageRequest, SearchParams searchParams) {

**return** mapper.selectAllByPage(pageRequest, searchParams).getPage();

}

**public** **void** save(List<ExampleTemplate> recordList) {

List<ExampleTemplate> addList = **new** ArrayList<>(recordList.size());

List<ExampleTemplate> updateList = **new** ArrayList<>(recordList.size());

**for** (ExampleTemplate cardTable : recordList) {

**if** (cardTable.getId() == **null**) {

cardTable.setId(UUID.*randomUUID*().toString());

addList.add(cardTable);

} **else** {

updateList.add(cardTable);

}

}

**if** (CollectionUtils.*isNotEmpty*(addList)) {

mapper.batchInsert(addList);

}

**if** (CollectionUtils.*isNotEmpty*(updateList)) {

mapper.batchUpdate(updateList);

}

}

**public** **void** batchDeleteByPrimaryKey(List<ExampleTemplate> list) {

mapper.batchDeleteByPrimaryKey(list);

}

**private** **static** String *values* = "{'id':'主键','code':'编码','name':'名称','remark':'备注'}";

**private** Map<String, String> headInfo;

**public** Map<String, String> getHeadInfo() {

**if** (headInfo == **null**) {

headInfo = **new** HashMap<String, String>();

net.sf.json.JSONObject json = net.sf.json.JSONObject.*fromObject*(*values*);

headInfo = (Map<String, String>) json;

}

**return** headInfo;

}

/\*\*

\* 下载excel模版

\*/

**public** **void** downloadExcelTemplate(HttpServletResponse response) **throws** Exception {

ExcelExportImportor.*downloadExcelTemplate*(response, getHeadInfo(), "任务模板", "测试模板");

}

**public** **void** importExcelData(InputStream excelStream) **throws** Exception {

List<ExampleTemplate> list = ExcelExportImportor.*loadExcel*(excelStream, getHeadInfo(), ExampleTemplate.**class**);

List<String> ids = mapper.getIds();

**for** (ExampleTemplate entity : list) {

System.***out***.println("Entityid:" + entity.getId());

**if** (ids.contains(entity.getId().trim())) {

entity.setStatus(VOStatus.***UPDATED***); // 编辑

} **else** {

entity.setStatus(VOStatus.***NEW***); // 新增

}

saveEntity(entity);

}

}

**private** ExampleTemplate saveEntity(ExampleTemplate entity) {

**if** (entity.getStatus() == VOStatus.***NEW***) {

mapper.insert(entity);

} **else** {

mapper.update(entity);

}

**return** entity;

}

**public** List<ExampleTemplate> getByIds(List<String> ids) {

List<ExampleTemplate> list = mapper.getByIds(ids);

**return** list;

}

/\*\*

\* 导出excel数据

\*/

**public** **void** exportExcelData(PageRequest pageRequest, HttpServletResponse response, String ids) **throws** Exception {

List<ExampleTemplate> list = **null**;

**if** (**null** != ids && ids.length() > 0) {

String[] pks = ids.split(",");

list = **this**.getByIds(Arrays.*asList*(pks));

} **else** {

SearchParams searchParams = **new** SearchParams();

Page<ExampleTemplate> page = selectAllByPage(pageRequest, searchParams, ExampleTemplate.**class**);

list = page.getContent();

}

ExcelExportImportor.*writeExcel*(response, list, getHeadInfo(), "任务详细", "任务信息");

}

**public** ExampleTemplate queryByPK(String id) {

List<String> ids = **new** ArrayList<String>();

ids.add(id);

List<ExampleTemplate> list = mapper.getByIds(ids);

**if** (list != **null** && list.size() > 0) {

ExampleTemplate entity = list.get(0);

**return** entity;

} **else** {

**return** **null**;

}

}

}

### 编写controller文件

@RequestMapping(value = "/excelTemplateDownload", method = { RequestMethod.***GET***, RequestMethod.***POST*** })

**public** @ResponseBody Map<String, String> excelTemplateDownload(HttpServletRequest request,

HttpServletResponse response) **throws** BusinessException {

Map<String, String> result = **new** HashMap<String, String>();

**try** {

service.downloadExcelTemplate(response);

result.put("status", "success");

result.put("msg", "督办任务信息Excel模版下载成功");

} **catch** (Exception e) {

logger.error("督办任务信息Excel模版下载失败", e);

result.put("status", "failed");

result.put("msg", "督办任务信息Excel模版下载失败");

}

**return** result;

}

@RequestMapping(value = "/excelDataImport", method = RequestMethod.***POST***)

**public** @ResponseBody Map<String, String> excelDataImport(

@RequestParam(value = "fileName", required = **false**) MultipartFile fileName, ModelMap model,

HttpServletRequest request) **throws** BusinessException {

Map<String, String> result = **new** HashMap<String, String>();

**try** {

service.importExcelData(fileName.getInputStream());

result.put("status", "success");

result.put("msg", "Excel导入成功");

} **catch** (Exception e) {

logger.error("Excel导入失败", e);

result.put("status", "failed");

result.put("msg", "Excel导入失败");

}

**return** result;

}

@RequestMapping(value = "/excelDataExport", method = { RequestMethod.***GET***, RequestMethod.***POST*** })

**public** @ResponseBody Map<String, String> excelDataExport(PageRequest pageRequest, HttpServletRequest request,

@RequestParam String ids, HttpServletResponse response) **throws** BusinessException {

Map<String, String> result = **new** HashMap<String, String>();

**try** {

service.exportExcelData(pageRequest, response, ids);

result.put("status", "success");

result.put("msg", "信息导出成功");

} **catch** (Exception e) {

logger.error("Excel下载失败", e);

result.put("status", "failed");

result.put("msg", "Excel下载失败");

}

**return** result;

}

### 编写xml文件

根据实际业务需求编写相关sql文件。

## 前端开发

### 编写viewModel.js

viewModel.js中定义数据模型，可根据实际业务需求修改。

var meta = {

meta: {

id: { type: "string" },

code: {

type: "string",

required: true,

nullMsg: "不能为空!"

},

name: {

type: "string",

required: true,

nullMsg: "不能为空!"

},

remark: {

type: "string"

}

}

};

var viewModel = {

draw: 1,

//页数(第几页)

pageSize: 5,

gridData: new u.DataTable(meta),

formData: new u.DataTable(meta),

};

### 编写template.html文件

新增三个按钮（下载模板、导入、导出）绑定三个方法。

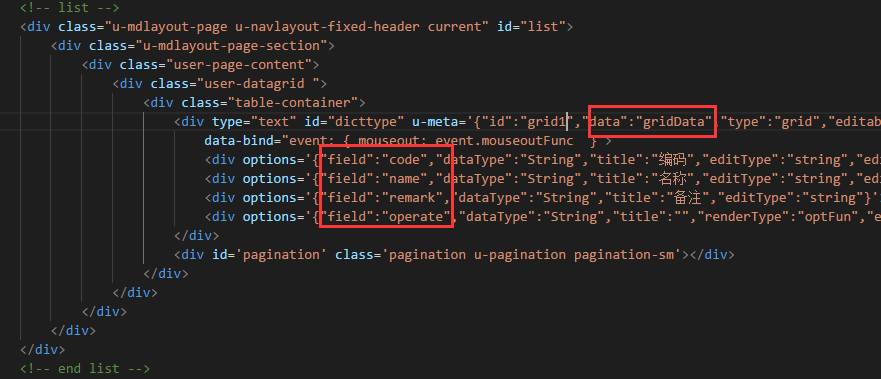
data-bind="click: event.onTempletExcel"

data-bind="click: event.onUploadFile"

data-bind="click: event.onDownloadExcel

添加表格对象

绑定表格字段到meta中定义的filed字段。



<div class="u-mdlayout-page u-navlayout-fixed-header current" id="list">

<div class="u-mdlayout-page-section">

<div class="user-page-content">

<div class="user-datagrid ">

<div class="table-container">

<div type="text" id="dicttype" u-meta='{"id":"grid1","data":"gridData","type":"grid","editable":false,"editType":"default","rowHeight":"40px","multiSelect":true,"canDrag":false,"canSwap":false,"onRowSelected":"event.selectRow","onRowUnSelected":"event.selectRow","onRowHover":"event.rowHoverHandel"}'

data-bind="event: { mouseout: event.mouseoutFunc }">

<div options='{"field":"code","dataType":"String","title":"编码","editType":"string","editOptions":{"validType":"string","required":true}}'></div>

<div options='{"field":"name","dataType":"String","title":"名称","editType":"string","editOptions":{"validType":"string","required":true},"renderType":"event.renderName"}'></div>

<div options='{"field":"remark","dataType":"String","title":"备注","editType":"string"}'></div>

<div options='{"field":"operate","dataType":"String","title":"","renderType":"optFun","editable":true}'></div>

</div>

<div id='pagination' class='pagination u-pagination pagination-sm'></div>

</div>

</div>

</div>

</div>

</div>

### 编写template.css文件

文件中定义的样式可以根据实际情况修改。

### 编写template.js文件

* 添加ajax对上传文件的封装，

'/iuap-saas-filesystem-service/resources/js/ajaxfileupload.js',

'/iuap-saas-filesystem-service/resources/js/ossupload.js',

'interfaceFileImpl'

define(["text!./template.html",

"./meta.js",

"css!./template.css",

"../../config/sys\_const.js",

'/iuap-saas-filesystem-service/resources/js/ajaxfileupload.js',

'/iuap-saas-filesystem-service/resources/js/ossupload.js',

'interfaceFileImpl'],

* 定义请求地址和模型对象。

var ctx = cookie.appCtx + "/exampleTemplate";

var viewModel = {

draw: 1,

pageSize: 5,

listRowUrl: ctx + "/list",                  //列表查询URL

saveRowUrl: ctx + "/save",                  //新增和修改UR

delRowUrl: ctx + "/delete",                 //刪除URL

gridData: new u.DataTable(meta),

formData: new u.DataTable(meta),

* 模板下载

注意构造了form对象，请求地址为：ctx + "/excelTemplateDownload"。

//模板下载

onTempletExcel: function () {

var form = $("<form>"); //定义一个form表单

form.attr('style', 'display:none'); //在form表单中添加查询参数

form.attr('target', '');

form.attr('method', 'post');

form.attr('action', ctx + "/excelTemplateDownload");

$('#user-mdlayout').append(form); //将表单放置在web中

var input2 = $('<input>');

input2.attr('type', 'hidden');

input2.attr('name', 'x-xsrf-token');

input2.attr('value', window.x\_xsrf\_token);

form.append(input2);

form.submit();

},

* 导入数据

导入主要注意请求地址：var urlInfo = "/excelDataImport";

//任务信息导入

onUploadFile: function () {

window.md = u.dialog({ id: 'testDialg2', content: "#dialog\_content", hasCloseMenu: true });

$('.sub-list1-new').css('display', 'inline-block');

$("#filenamediv2").html("");

$("#uploadingIMG2").attr("src", "../example/static/beforeUpload.svg"); $("#uploadingMsg2").html("").addClass("uploading").removeClass("fail").removeClass("success");

$("#fileName").remove(); $("#dialog\_content").find(".choosefile").append('<input class="u-input" type="file" name="fileName" id="fileName"/>');

var demoInput = document.getElementById('fileName');

demoInput.addEventListener('change', viewModel.event.onUploadExcel);

},

onUploadExcel: function () {

var filevalue = document.getElementById("fileName").files;

console.log("filevalue:", filevalue);

f (!filevalue || filevalue.length < 1) {

var demoInput = document.getElementById('fileName');

demoInput.removeEventListener('change', viewModel.event.onUploadExcel);

demoInput.addEventListener('change', viewModel.event.onUploadExcel);

return;

} else {

$("#filenamediv2").html(filevalue[0].name);

}

var urlInfo = "/excelDataImport";

viewModel.event.uploadingshow();

$.ajaxFileUpload({

url: ctx + urlInfo,

secureuri: false,

fileElementId: 'fileName',

dataType: 'JSON',//返回值

success: function (data) {

$("#fileName").remove(); $("#dialog\_content").find(".choosefile").append('<input class="u-input" type="file" name="fileName" id="fileName"/>');

var demoInput = document.getElementById('fileName');

demoInput.removeEventListener('change', viewModel.event.onUploadExcel);

demoInput.addEventListener('change', viewModel.event.onUploadExcel);

window.clearInterval(viewModel.loadingTimer);

$('.file-lodedPart').width(200);

viewModel.event.uploadingHide();

$("#uploadingIMG2").attr("src", "../example/static/success.svg");

$("#uploadingMsg2").html("导入成功").addClass("success").removeClass("uploading").removeClass("fail");

viewModel.event.initGridDataList();

md.close();

},

error: function (XMLHttpRequest, textStatus, errorThrown) {

$("#fileName").remove(); $("#dialog\_content").find(".choosefile").append('<input class="u-input" type="file" name="fileName" id="fileName"/>');

var demoInput = document.getElementById('fileName');

demoInput.removeEventListener('change', viewModel.event.onUploadExcel);

demoInput.addEventListener('change', viewModel.event.onUploadExcel);

$("#uploadingIMG2").attr("src", "../example/static/fail.svg");

$("#uploadingMsg2").html("导入失败" + data.message).addClass("fail").removeClass("uploading").removeClass("success"); window.clearInterval(viewModel.loadingTimer);}

});

},

* 导出数据

先选中需要导出的数据行，然后点击导出。构造了form对象，采用post方法，请求地址为：ctx + "/excelDataExport"。

//导出

onDownloadExcel: function () {

var dats = [];

var pks = ""

var row = viewModel.gridData.getSelectedRows();

if (row == null || row.length == 0) {

u.messageDialog({ msg: "请选择要导出的数据", title: "提示", btnText: "确定" });

return;

}

if (row != null && row.length != 0) {

for (var i = 0; i < row.length; i++) {

var pkItem = row[i].getValue("id");

dats.push(row[i].getSimpleData());

if (pks.length == 0) {

pks = pkItem;

} else {

pks = pks + "," + pkItem;

}

}

}

var form = $("<form>"); //定义一个form表单

form.attr('style', 'display:none'); //在form表单中添加查询参数

form.attr('target', '');

form.attr('method', 'post');

form.attr('action', ctx + "/excelDataExport");

$('#user-mdlayout').append(form); //将表单放置在web中

var input1 = $('<input>');

input1.attr('type', 'hidden');

input1.attr('name', 'ids');

input1.attr('value', pks);

form.append(input1); //将查询参数控件提交到表单上

var input2 = $('<input>');

input2.attr('type', 'hidden');

input2.attr('name', 'x-xsrf-token');

input2.attr('value', window.x\_xsrf\_token);

form.append(input2);

form.submit();

},