

**iuap平台消息组件开发**

**互联网开发培训教程**

2019年

# 版权

©2019用友集团版权所有。

未经用友集团的书面许可，本文档描述任何整体或部分的内容不得被复制、复印、翻译或缩减以用于任何目的。本文档描述的内容在未经通知的情形下可能会发生改变，敬请留意。请注意：本文档描述的内容并不代表用友集团所做的承诺。

目录

[版权 2](#_Toc23072)

[第一章 概述 4](#_Toc2562)

[1.1 业务场景 4](#_Toc3521)

[1.2 技术架构 4](#_Toc23300)

[1.3 应用效果 5](#_Toc29095)

[第二章 应用平台配置 7](#_Toc23210)

[2.1 发送站内消息配置 7](#_Toc24655)

[2.1.1 预置业务对象消息实体 7](#_Toc30467)

[2.1.2 预置业务对象消息实体 7](#_Toc16107)

[2.1.3 预置消息模板数据 8](#_Toc31320)

[2.1.4 创建消息模板 9](#_Toc10516)

[2.2 发送邮件配置 10](#_Toc27269)

[2.2.1 消息通道配置 10](#_Toc11065)

[2.2.2 用户邮箱设置 10](#_Toc20533)

[第三章 开发示例 11](#_Toc2408)

[3.1 后端开发 11](#_Toc3340)

[3.1.1 修改配置文件 11](#_Toc15705)

[3.1.2 Service层编写 11](#_Toc24570)

[3.1.3 Control层编写 13](#_Toc28137)

[第四章 其他参考 14](#_Toc29009)

# 概述

## 业务场景

业务开发过程中经常需要发送站内消息和发送邮件，IUAP的消息组件能满足此类需求，本示例将介绍如何实现。

* 发送站内消息（普通消息、固定模板的消息）
* 发送邮件消息（模拟给U001这用户发送邮件）

## 技术架构



图 1 消息平台架构图

消息平台接收来自业务系统的消息发送请求，并按照相应消息模板和消息通道实现消息的发送。

消息通道支持扩展功能。

消息模板提供默认的消息模版类型：html消息模版、纯文本消息。Html消息模版主要用于web浏览器、邮件等显示支持富文本消息内容。纯文本的，一般为短信消息使用，支持配置短信模版时限制输入长度。

消息平台支持web端消息推送。Web端推送采用socket长链接的方式，使用vert.x构建web端未读消息推送服务，监控在线用户，向在线用户推送相应消息。

消息平台支持短信、邮件、APP等方式推送消息。

## 应用效果



图 2 ：发送站内消息

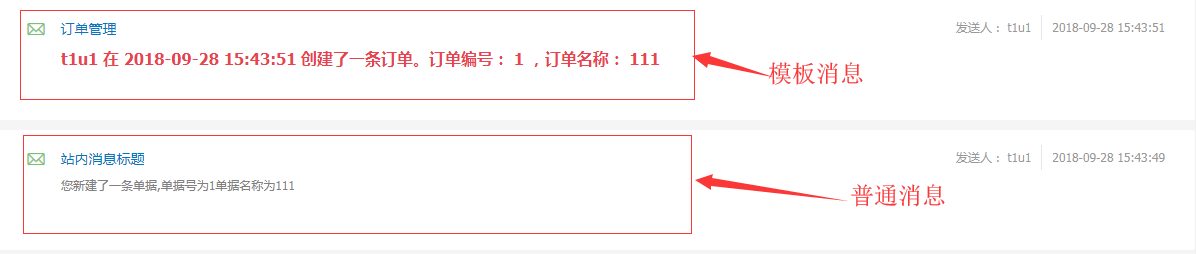


图 3 ：收到站内消息

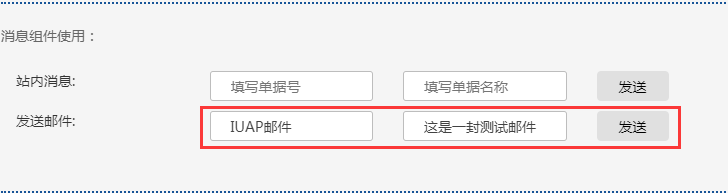


图 4 发送邮件消息



图 5 收到邮件消息

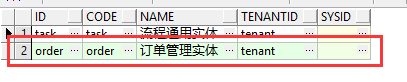
# 应用平台配置

发送站内消息和发送邮件依赖于/iuap-saas-message-center，默认都随应用平台一起部署。

## 发送站内消息配置

### 预置业务对象消息实体

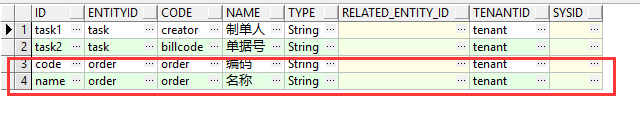
|  |
| --- |
| insert into iuap\_business\_entity (ID, CODE, NAME, TENANTID, SYSID)  values ('order', 'order', '订单管理实体', 'tenant', null); |



### 预置业务对象消息实体

iuap\_business\_propertity中的entityid要与iuap\_business\_entity对应

|  |
| --- |
| insert into iuap\_business\_propertity (ID, ENTITYID, CODE, NAME, TYPE, RELATED\_ENTITY\_ID, TENANTID, SYSID)  values ('code', 'order', 'order', '编码', 'String', null, 'tenant', null);  insert into iuap\_business\_propertity (ID, ENTITYID, CODE, NAME, TYPE, RELATED\_ENTITY\_ID, TENANTID, SYSID)  values ('name', 'order', 'order', '名称', 'String', null, 'tenant', null); |

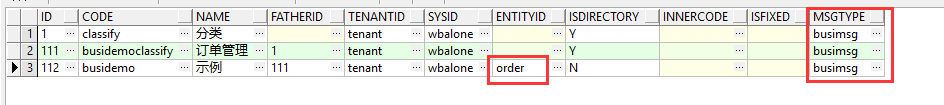


### 预置消息模板数据

entityid要与iuap\_business\_entity对应，msgtype 默认为 “busimsg”

mysql和oracle通用

|  |
| --- |
| insert into iuap\_msgtemplate\_manage (ID, CODE, NAME, FATHERID, TENANTID, SYSID, ENTITYID, ISDIRECTORY, INNERCODE, ISFIXED, MSGTYPE)  values ('1', 'classify', '分类', null, 'tenant', 'wbalone', null, 'Y', null, null, 'busimsg');  insert into iuap\_msgtemplate\_manage (ID, CODE, NAME, FATHERID, TENANTID, SYSID, ENTITYID, ISDIRECTORY, INNERCODE, ISFIXED, MSGTYPE)  values ('111', 'busidemoclassify', '订单管理', '1', 'tenant', 'wbalone', null, 'Y', null, null, 'busimsg');  insert into iuap\_msgtemplate\_manage (ID, CODE, NAME, FATHERID, TENANTID, SYSID, ENTITYID, ISDIRECTORY, INNERCODE, ISFIXED, MSGTYPE)  values ('112', 'busidemo', '示例', '111', 'tenant', 'wbalone', 'order', 'N', null, null, 'busimsg'); |



注意:

1.isdirectory字段表示是否是目录，Y表示是，N表示否；

此表中数据为消息模板管理界面左侧树

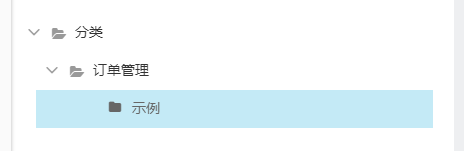


图 6 消息管理左侧树展示

### 创建消息模板

通过点击系统变量和业务对象下的节点可以组织成一个消息模板，如下图所示

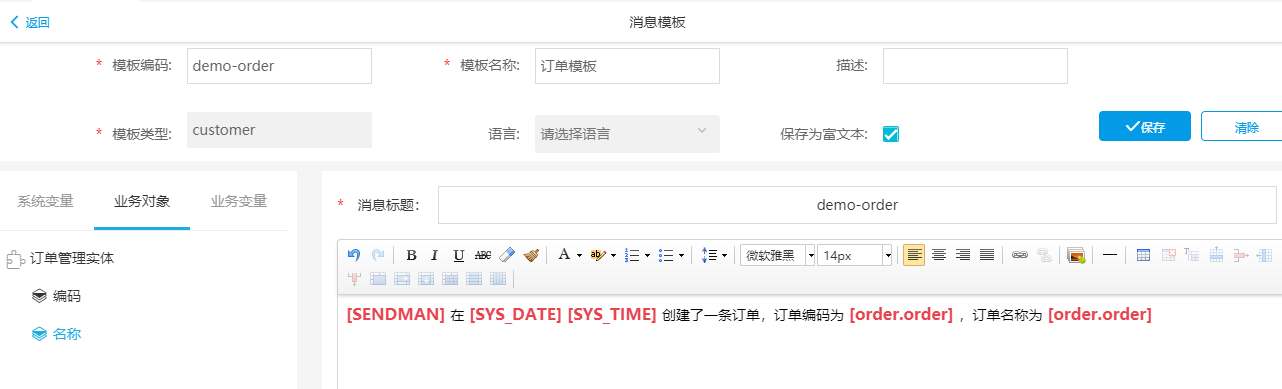


图 7 消息模板配置

点击保存，注意记住该模板的模板编码，稍后在后台配置时需要用到。

## 发送邮件配置

### 消息通道配置

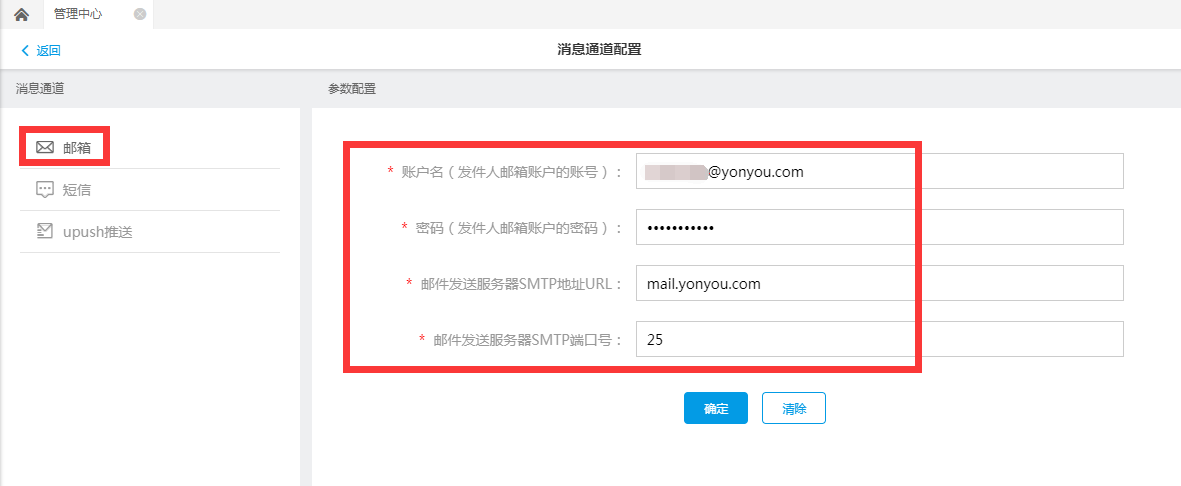


图 8 消息通道配置

### 用户邮箱设置

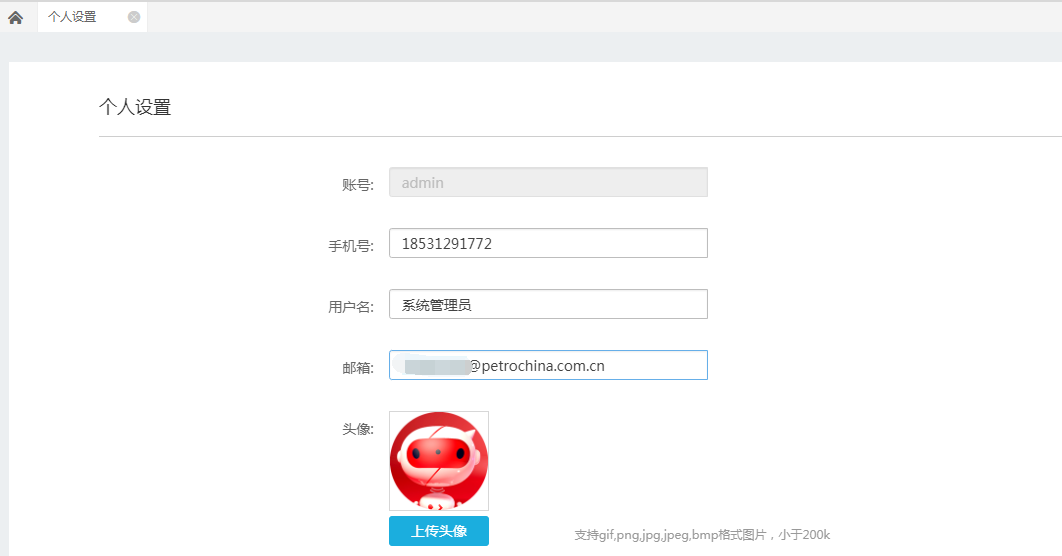


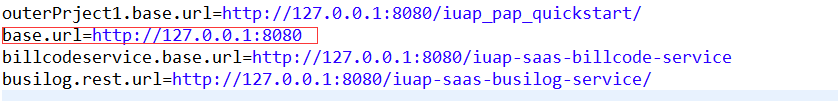
图 9

# 开发示例

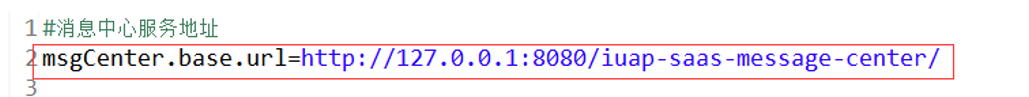
## 后端开发

### 修改配置文件

1. 修改开发环境中src/resource/application.properties文件中的base.url为IUAP的地址。



1. 修改开发项目中src/resources/目录下的msg-sdk.properties文件。



如果应用中心配置在本机，则msgCenter.base.url 配置为本机的IP加对应的端口号即可；如果应用中心配置在远程服务器上，则将msgCenter.base.url配置为远程服务器的IP加对应的端口号；

1. 修改authrbac-sdk.properties文件为应用平台地址

authrbac.server.http.url=http://127.0.0.1:8080/wbalone

### Service层编写

public void sendMessage(String no,String name) {

MessageEntity msg = new MessageEntity();

String userid = InvocationInfoProxy.getUserid();

String userName = InvocationInfoProxy.getUsername();

//发送人,当前设置为当前操作人

msg.setSendman(userid);

//SYS是消息中心

msg.setChannel(new String[]{WebappMessageConst.CHANNEL\_SYS});

//接收者,数组形式,可以根据实际情况配置多人

msg.setRecevier(new String[]{InvocationInfoProxy.getUserid(), "U001"});

//消息模板编码,即在管理中心创建的消息模板编号

msg.setTemplatecode("demo\_order");

//消息ID

msg.setBillid(no);

msg.setTencentid(InvocationInfoProxy.getTenantid());

msg.setMsgtype(WebappMessageConst.MESSAGETYPE\_NOTICE);

msg.setSubject("站内消息标题");

msg.setContent("您新建了一条单据,单据号为" + no + "单据名称为" + name);

JSONObject busiData = new JSONObject();

//entity.code 即为在注册的实体加上实体的属性

busiData.put("busientity.code", no);

//entity.name

busiData.put("busientity.name", name);

//这段方法是根据userid 获取username

String url\_user = "/wbalone/userRest/getByIds";

List<String> listUser\_name = new ArrayList<String>();

listUser\_name.add(userid);

Map<String, String> mapParameter4 = new ConcurrentHashMap<String, String>();

mapParameter4.put("tenantId", "tenant");

mapParameter4.put("userIds", JSONArray.toJSON(listUser\_name).toString());

Map<String, String> mapUser\_name = convertRefName(url\_user, mapParameter4);

userName = mapUser\_name.get(userid);

//设置发送人,为消息模板中的系统变量

busiData.put("SENDMAN", userName);

SimpleDateFormat df0 = new SimpleDateFormat("HH:mm:ss");

SimpleDateFormat df1 = new SimpleDateFormat("yyyy-MM-dd");

//设置日期,为消息模板中的系统变量

busiData.put("SYS\_DATE", df1.format(System.currentTimeMillis()));

//设置时间,为消息模板中的系统变量

busiData.put("SYS\_TIME", df0.format(System.currentTimeMillis()));

//发送模板消息

commonMessageSendService.sendTemplateMessage(msg, busiData);

//发送站内消息

commonMessageSendService.sendTextMessage(msg, busiData);

}

### Control层编写

public void sendEmail(String title,String content) {

JSONObject msgObj = new JSONObject();

JSONObject jsonObj = new JSONObject();

jsonObj.put("sendman",InvocationInfoProxy.getUserid());

jsonObj.put("recevier",new String[]{"U001"});//这里发给ID为U001的用户

jsonObj.put("subject",title);

jsonObj.put("tenantid",InvocationInfoProxy.getTenantid());

String[] channels = {WebappMessageConst.CHANNEL\_EMAIL};

jsonObj.put("channel",channels);

jsonObj.put("msgtype", WebappMessageConst.MESSAGETYPE\_NOTICE);

jsonObj.put("content",content);

msgObj.put("data",jsonObj);

MessageCenterUtil.pushTextMessage(msgObj.toJSONString());

}

# 其他参考

在WebappMessageConst常量中定义了一些常用常量

通道类型：

* 邮件email(CHANNEL\_EMAIL)、
* 短信note(CHANNEL\_SMS)、
* upush-messagepush(CHANNEL\_UPUSH)、
* 系统消息sys(CHANNEL\_SYS)

消息类型：

* 通知notice（MESSAGETYPE\_NOTICE）、
* 任务task(MESSAGETYPE\_TASK)、
* 文本消息text(MESSAGECONTENTTYPE\_TEXT)、
* 模板消息template(MESCONTENTTYPE\_TEMPLATE)