5-8 执行系统消息异步收发

一、注册流程中的消息发送

我们封装好了消息模块的代码，接下来我们可以挑选一个功能测试系统消息的收发。比如说用户注册流程中，当用户成功注册的时候，我们就利用 `MessageTask` 异步发送消息到MQ队列中。

在 `UserServiceImpl.java` 中，添加发送系统消息的代码

```
1. public int registerUser(String registerCode, String code, String nickname, String photo) {
2.     ……
3.     MessageEntity entity = new MessageEntity();
4.     entity.setSenderId(0);
5.     entity.setSenderName("系统消息");
6.     entity.setUuid(IdUtil.simpleUUID());
7.     entity.setMsg("欢迎您注册成为超级管理员，请及时更新你的员工个人信息。");
8.     entity.setSendTime(new Date());
9.     messageTask.sendAsync(id + "", entity);
10.    ……
11. }
```

用户注册成功之后，系统消息应该发送到 `message` 集合作为存档，而 `message_ref` 集合中没有记录。

二、登陆流程中的消息接收

在用户离线的过程中，Emos系统发送的消息通知是存放在MQ队列中的，所以用户登陆之后就需要接收这些消息通知。其实就是让 `MessageTask` 异步接受MQ中的消息，然后存储在 `message_ref` 集合中。

在 `UserServiceImpl.java` 中，添加接收系统消息的代码

```
1. public Integer login(String code) {
2.     ……
3.     messageTask.receiveAysnc(id+"");
4.     ……
5. }
```

三、轮询接收系统消息

在 `MessageController.java` 中，添加接收轮询消息的Web方法

```
public class MessageController {
    ……
    @GetMapping("/refreshMessage")
    @ApiOperation("刷新用户的消息")
    public R refreshMessage(@RequestHeader("token") String token) {
        int userId = jwtUtil.getUserId(token);
        //异步接收消息
        messageTask.receiveAysnc(userId + "");
        //查询接收了多少条消息
        long lastRows=messageService.searchLastCount(userId);
        //查询未读数据
        long unreadRows = messageService.searchUnreadCount(userId);
        return R.ok().put("lastRows", lastRows).put("unreadRows", unreadRows);
    }
}
```

在 `main.js` 文件中添加全局路径

```
Vue.prototype.url = {
    ……
    refreshMessage: baseUrl + "/message/refreshMessage"
}
```

在 `index.vue` 文件中引入气泡消息控件

```
import uniPopup from '@/components/uni-popup/uni-popup.vue';
import uniPopupMessage from '@/components/uni-popup/uni-popup-message.vue';
import uniPopupDialog from '@/components/uni-popup/uni-popup-dialog.vue';
export default {
    components: {
        uniPopup,
        uniPopupMessage,
        uniPopupDialog
    },
    data() {
        return {
            timer: null,
            unreadRows: 0,
            lastRows: 0
        };
    }
    ……
}
```

在页面上引入气泡消息标签

```
<uni-popup ref="popupMsg" type="top">
    <uni-popup-message type="success" :message="'接收到' + lastRows + '条消息'"
:duration="2000"/>
</uni-popup>
```

编写JavaScript代码，执行消息轮询

```
onLoad:function() {
    // 监听事件
    let that = this;
    uni.$on('showMessage', function() {
        that.$refs.popupMsg.open();
    });
},
onUnload:function() {
    // 移除监听事件
    uni.$off('showMessage');
},
onShow: function() {
    let that = this;
    that.timer = setInterval(function() {
        that.ajax(that.url.refreshMessage, 'GET', null, function(resp) {
            that.unreadRows = resp.data.unreadRows;
            that.lastRows = resp.data.lastRows;
            if (that.lastRows > 0) {
                uni.$emit('showMessage');
            }
        });
    }, 5 * 1000);
},
onHide:function() {
    clearInterval(this.timer);
}
```