

[🏠 / 开发指南 / 后端手册](#)[👤 芋道源码](#) [📅 2023-11-23](#)

WebSocket 实时通信

1. 功能简介

项目的 [yudao-spring-boot-starter-websocket](#) 组件，基于 [Spring WebSocket](#) 进行二次封装，实现了更加简单的使用方式。例如说，WebSocket 的认证、Session 的管理、WebSocket 集群的消息广播等等。

疑问：为什么不使用 Netty 实现 WebSocket？

Netty 的学习和使用门槛较高，对大家可能不够友好，而 Spring WebSocket 足够满足 99.99% 的场景。

1.1 Token 身份认证

① 在 WebSocket 连接建立时，通过 QueryString 的 `token` 参数，进行认证。例如说：

`ws://127.0.0.1:48080/ws?token=xxx`。

由于 WebSocket 是基于 HTTP 建立连接，所以它的认证可以复用项目的 [TokenAuthenticationFilter](#) 实现。

为什么 token 不使用 Header 传递？

WebSocket 不支持 Header 传递，所以只能使用 QueryString 传递。

② 认证完成后，会通过 [LoginUserHandshakeInterceptor](#) 拦截器，将用户信息存储到 WebSocket Session 的 `attributes` 中。

这样，后续可以使用 [WebSocketFrameworkUtils](#) 获取用户信息，例如说：

```
// WebSocketFrameworkUtils.java

// ① 获取当前用户
public static LoginUser getLoginUser(WebSocketSession session)

// ② 获得当前用户的类型
public static Integer getLoginUserType(WebSocketSession session)

// ③ 获得当前用户的编号
public static Integer getLoginUserType(WebSocketSession session)
```

```
// ④ 获得当前用户的租户编号
public static Long getTenantId(WebSocketSession session)
```

1.2 Session 会话管理

每个前端和后端建立的 WebSocket 连接，对应后端的一个 WebSocketSession 会话对象。由于后续需要对 WebSocketSession 进行消息的发送，所以需要进行管理。

① WebSocketSession 的管理，由 [WebSocketSessionManager](#) 定义接口，由 [WebSocketSessionManagerImpl](#) 具体实现。

```
// 添加和移除 Session
void addSession(WebSocketSession session);
void removeSession(WebSocketSession session);

// 获得 Session，多种维度
WebSocketSession getSession(String id); // Session 编号
Collection<WebSocketSession> getSessionList(Integer userType); // 用户类型
Collection<WebSocketSession> getSessionList(Integer userType, Long userId); // 用户 ID
```

② WebSocket 建立和关闭连接时，通过 [WebSocketSessionHandlerDecorator](#) 处理器，分别调用 WebSocketSessionManager 进行 Session 的添加和移除。

1.3 Message 消息格式

WebSocket 默认使用“文本”进行通信，而业务需要按照不同类型的消息，进行不同的处理。因此，项目定义了 [JsonWebSocketMessage](#) 消息对象，包含 `type` 消息类型 + `content` 消息内容。

和 Spring MVC 对比，可以理解为：

	标识	方法	参数
Spring MVC	URL + Method 等	Controller 的 Method 方法	QueryString 或 RequestBody 等
项目 WebSocket	<code>type</code> 消息类型	WebSocketMessageListener 实现类	解析 <code>content</code> 消息内容后的 Message 对象

具体 [JsonWebSocketMessage](#) 和 [WebSocketMessageListener](#) 详细说明，参见「1.4 Message 消息接收」小节。

1.4 Message 消息接收

- ① WebSocket 接收到项目后，会先交给 `JsonWebSocketMessageHandler` 消息处理器，将消息解析成 `JsonWebSocketMessage` 对象。
之后，根据 `type` 消息类型，获得到 `WebSocketMessageListener` 实现类，并将 `content` 消息内容进一步解析成 `Message` 对象，交给它进行处理。
- ② 具体案例，可见 `DemoWebSocketMessageListener`、`DemoSendMessage` 类。

1.5 Message 消息推送

- ① 项目的 `WebSocketMessageSender` 接口，定义了给 Session 发送消息的方法。如下所示：

```
// WebSocketMessageSender.java

// ① 发送消息给指定用户
void send(Integer userType, Long userId, String messageType, String messageContent);
default void sendObject(Integer userType, Long userId, String messageType, Object messageContent) {
    send(userType, userId, messageType, JsonUtils.toJsonString(messageContent));
}

// ② 发送消息给指定用户类型
void send(Integer userType, String messageType, String messageContent);
default void sendObject(Integer userType, String messageType, Object messageContent) {
    send(userType, messageType, JsonUtils.toJsonString(messageContent));
}

// ③ 发送消息给指定 Session
void send(String sessionId, String messageType, String messageContent);
default void sendObject(String sessionId, String messageType, Object messageContent) {
    send(sessionId, messageType, JsonUtils.toJsonString(messageContent));
}
```

- ② `WebSocketMessageSender` 有多种实现类，如下：

实现类	是否支持 WebSocket 集群	前置要求
LocalWebSocketMessageSender	✗	无
RedisWebSocketMessageSender	✓	开启 《消息队列（Redis）》
RocketMQWebSocketMessageSender	✓	开启 《消息队列（RocketMQ）》
KafkaWebSocketMessageSender	✓	开启 《消息队列（Kafka）》

实现类	是否支持 WebSocket 集群	前置要求
RabbitMQWebSocketMessageSender		开启 《消息队列（RabbitMQ）》

疑问：什么是 WebSocket 集群？

在后端部署多个 Java 进程时，会形成 WebSocket 集群。此时，就会存在跨进程的消息推送问题。例如说，连接 A 进程的 WebSocket 的用户，想要发送消息给连接 B 进程的 WebSocket 用户。

🤖 如何解决呢？消息不直接发送给用户 WebSocketSession，而是先发给 Redis、RocketMQ 等消息队列，再由每个 Java 进程监听该消息，分别判断判断该用户 WebSocket 是否连接的是自己，如果是，则进行消息推送。

默认配置下，使用 LocalWebSocketMessageSender 本地发送消息，不支持 WebSocket 集群。可通过修改 application.yaml 配置文件的 yudao.websocket.sender-type 来切换，如下：

```
yudao:
  websocket:
    enable: true # websocket的开关
    path: /infra/ws # 路径
    sender-type: redis # 消息发送的类型，可选值为 local、redis、rocketmq、kafka、rabbitmq
    sender-rocketmq:
      topic: ${spring.application.name}-websocket # 消息发送的 RocketMQ Topic
      consumer-group: ${spring.application.name}-websocket-consumer # 消息发送的
    sender-rabbitmq:
      exchange: ${spring.application.name}-websocket-exchange # 消息发送的 RabbitMQ Queue
      queue: ${spring.application.name}-websocket-queue # 消息发送的 RabbitMQ Queue
    sender-kafka:
      topic: ${spring.application.name}-websocket # 消息发送的 Kafka Topic
      consumer-group: ${spring.application.name}-websocket-consumer # 消息发送的
```

另外，默认的 WebSocket 连接地址是 ws://127.0.0.1:48080/infra/ws ，可通过 yudao.websocket.path 配置项进行修改。

2. 使用方案

目前有 2 种使用方案，分别是：

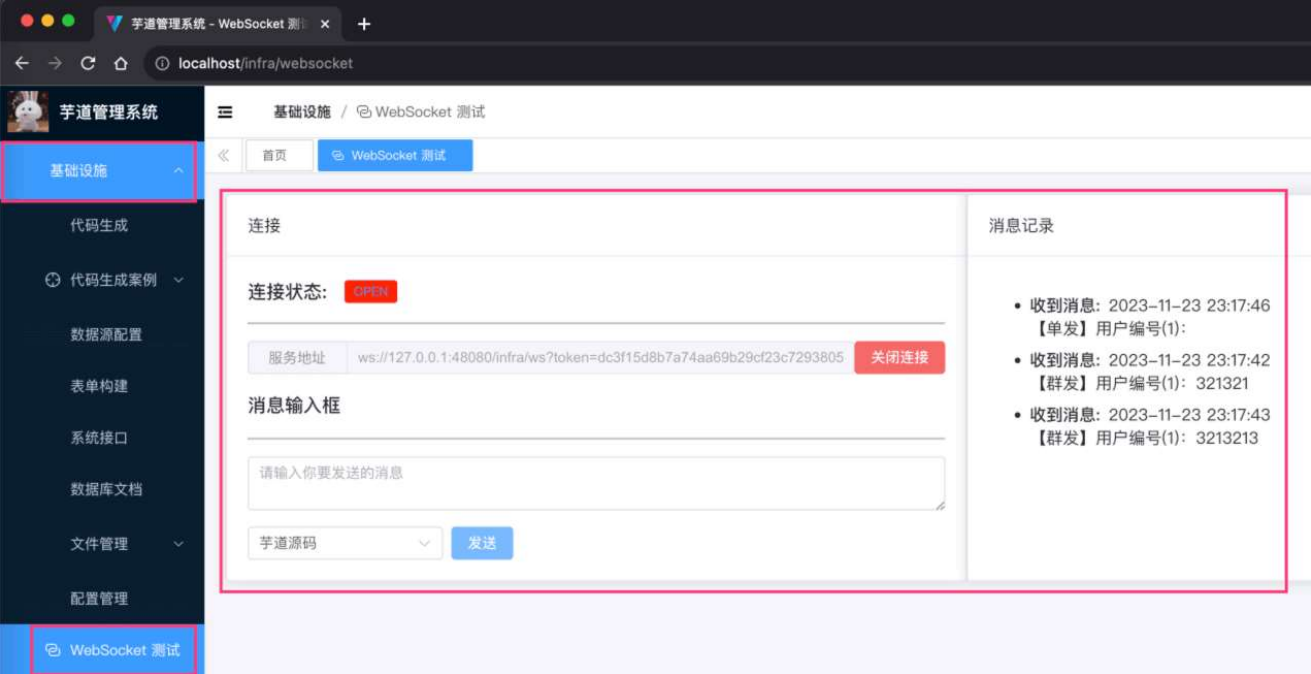
方案名	上行	下行
方案一：纯 WebSocket	WebSocket	WebSocket
方案二：WebSocket + HTTP	HTTP	WebSocket

疑问：什么是上行？什么是下行？

- 上行：指的是“前端”发送消息给“后端”，WebSocket 和 HTTP 都可以。
- 下行：指的是“后端”发送消息给“前端”，只能使用 WebSocket。

友情提示：下文中提到的所有配置，项目都已经配置好。你只需要按照下文的步骤，进行调试即可，了解每个配置的作用即可。

2.1 方案一：纯 WebSocket



- 前端：见 [基础设施 -> WebSocket 测试] 菜单，对应 </views/infra/websocket/index.vue> 界面
- 后端：见 `yudao-module-infra-biz` 模块，对应 `DemoWebSocketMessageListener` 监听器

基于 WebSocket 实现的单聊和群聊，暂时不支持消息的持久化（刷新后，消息会消失）。建议，多多调试，更好的理解 WebSocket 流程。

2.1.1 后端代码

① 在 `yudao-module-infra-biz` 模块的 `pom.xml` 文件中，引入 `yudao-spring-boot-starter-websocket` 依赖。如下所示：

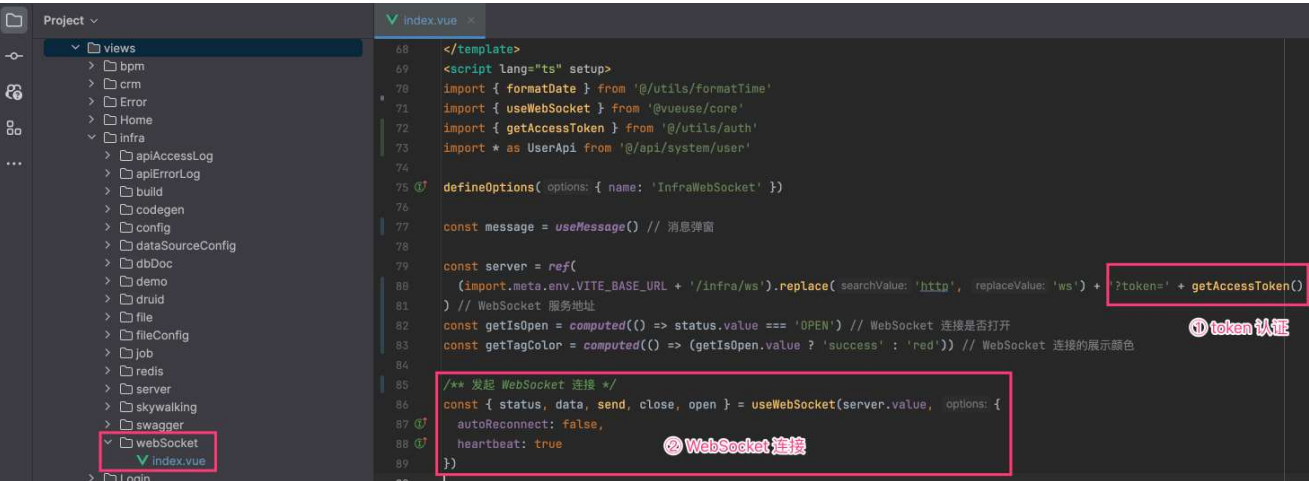
```
<dependency>
  <groupId>cn.iocoder.boot</groupId>
  <artifactId>yudao-spring-boot-starter-websocket</artifactId>
</dependency>
```

② 新建 DemoWebSocketMessageListener 类，实现对应消息的处理。如下图所示：

 DemoWebSocketMessageListener 类

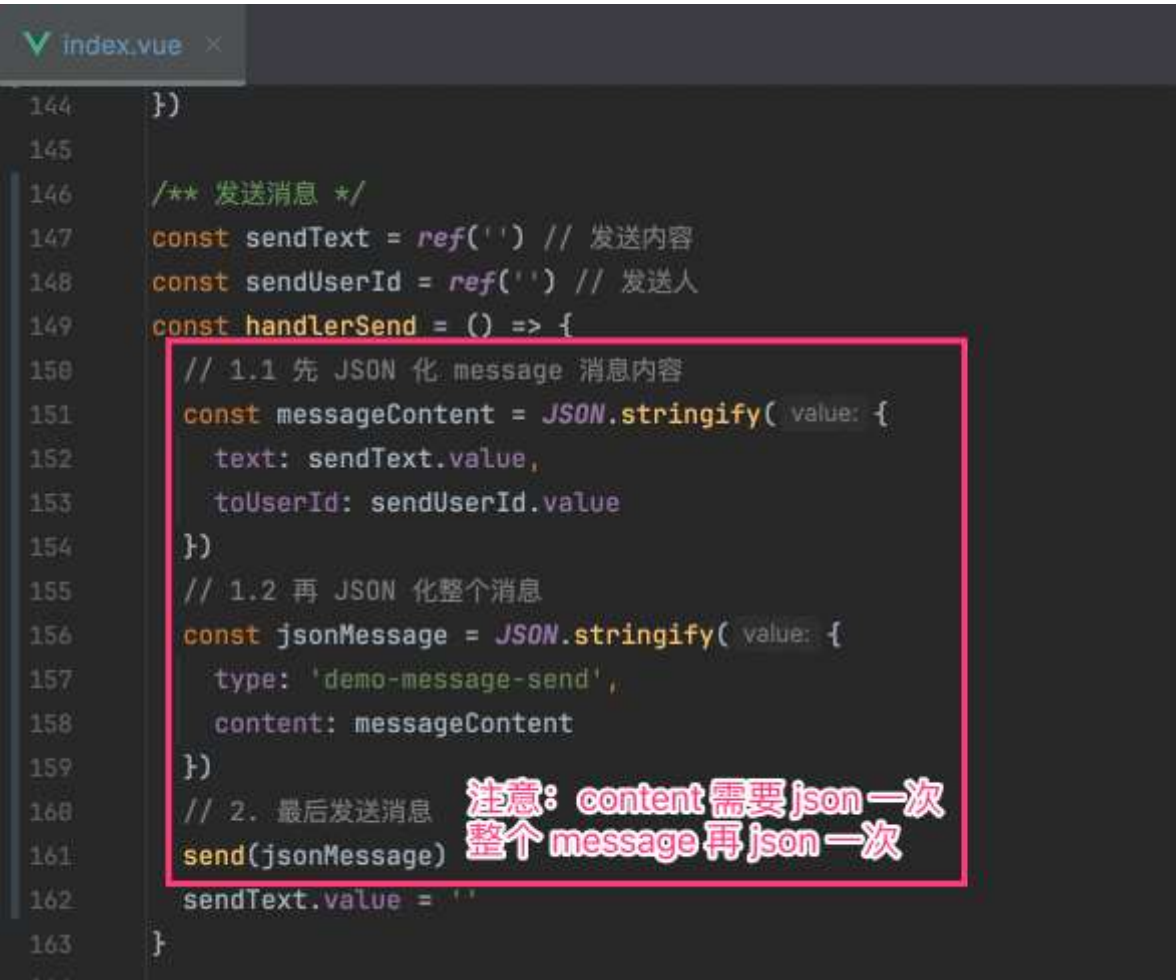
2.1.2 前端代码

① 建立 WebSocket 连接，如下图所示：



```
68 </template>
69 <script lang="ts" setup>
70   import { formatDate } from '@utils/formatTime'
71   import { useWebSocket } from '@vueuse/core'
72   import { getAccessToken } from '@utils/auth'
73   import * as UserApi from '@api/system/user'
74
75   defineOptions({ name: 'InfraWebSocket' })
76
77   const message = useMessage() // 消息弹窗
78
79   const server = ref(
80     (import.meta.env.VITE_BASE_URL + '/infra/ws').replace( searchValue: 'http', replaceValue: 'ws') + '?token=' + getAccessToken()
81   ) // WebSocket 服务地址
82   const getIsOpen = computed(() => status.value === 'OPEN') // WebSocket 连接是否打开
83   const getTagColor = computed(() => (getIsOpen.value ? 'success' : 'red')) // WebSocket 连接的展示颜色
84
85   /** 发起 WebSocket 连接 */
86   const { status, data, send, close, open } = useWebSocket(server.value, {
87     autoReconnect: false,
88     heartbeat: true
89   })
```

② 发送 WebSocket 消息，如下图所示：

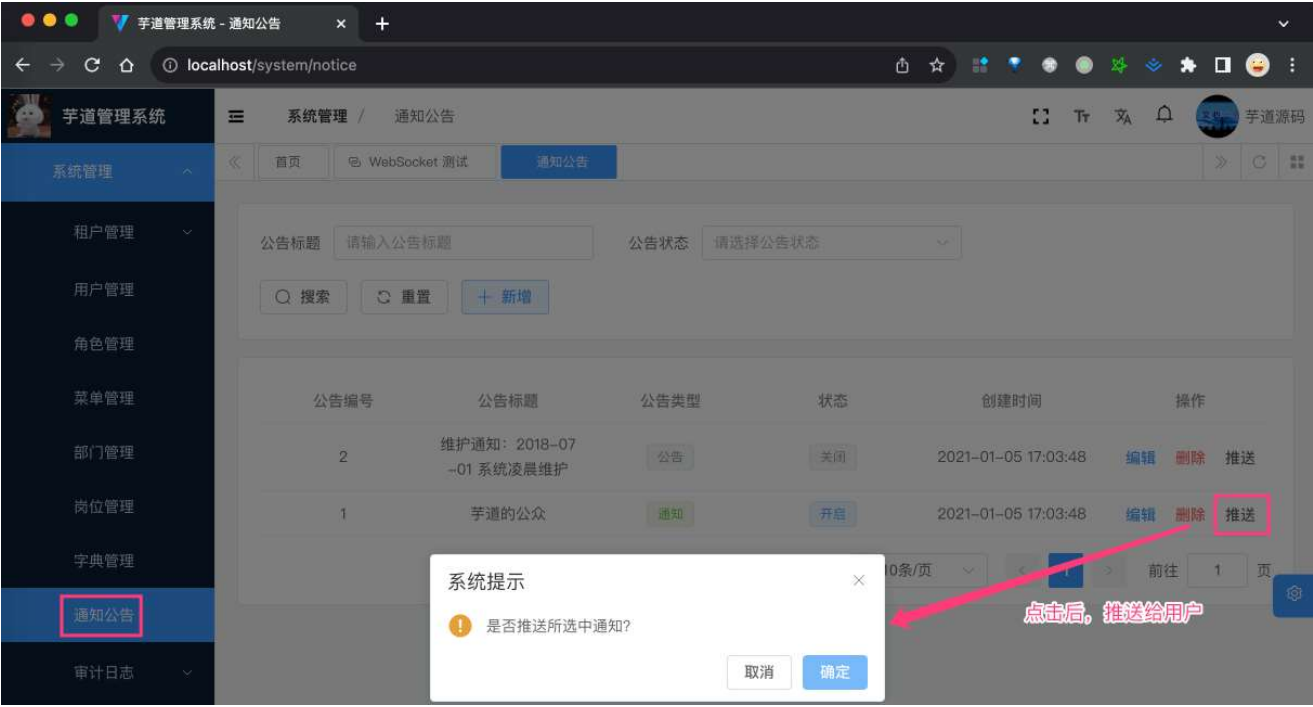


```
144 })
145
146 /** 发送消息 */
147 const sendText = ref('') // 发送内容
148 const sendUserId = ref('') // 发送人
149 const handlerSend = () => {
150   // 1.1 先 JSON 化 message 消息内容
151   const messageContent = JSON.stringify( value: {
152     text: sendText.value,
153     toUserId: sendUserId.value
154   })
155   // 1.2 再 JSON 化整个消息
156   const jsonMessage = JSON.stringify( value: {
157     type: 'demo-message-send',
158     content: messageContent
159   })
160   // 2. 最后发送消息
161   send(jsonMessage)
162   sendText.value = ''
163 }
```

③ 接收 WebSocket 消息。如下图所示：

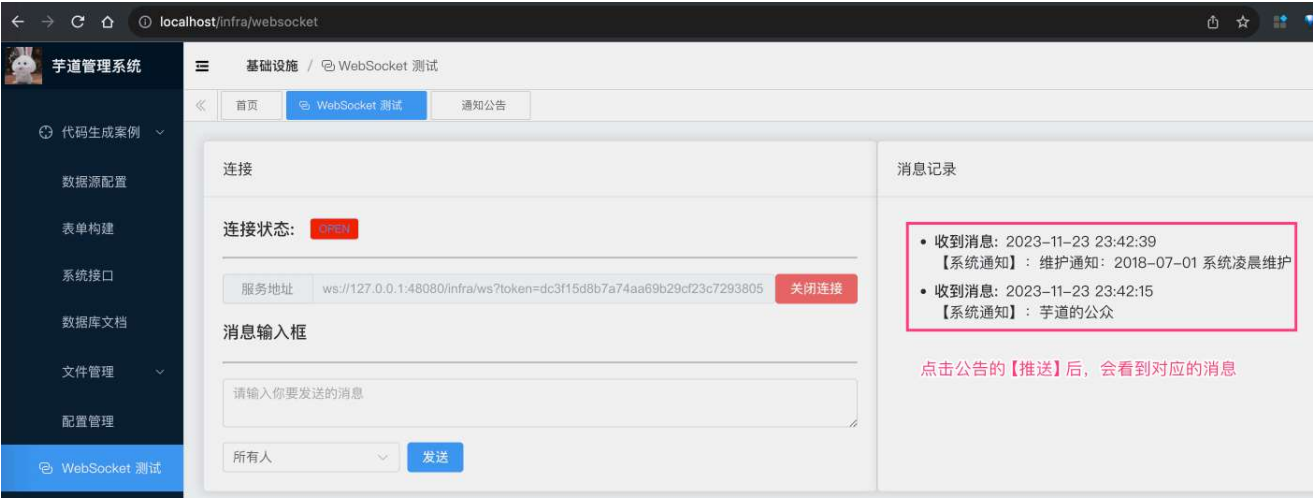
```
84
85  /** 发起 WebSocket 连接 */
86  > const { status, data, send, close, open } = useWebSocket(server.value, {...})
87
88
89
90
91  /** 监听接收到的数据 */
92  const messageList = ref([] as { time: number; text: string }[]) // 消息列表
93  watchEffect(() => {
94    > if (!data.value) {...}
95
96    try {
97      // 1. 收到心跳
98      > if (data.value === 'pong') {...}
99
100
101
102
103
104
105
106
107    // 2.1 解析 type 消息类型
108    const jsonMessage = JSON.parse(data.value)
109    const type = jsonMessage.type
110    const content = JSON.parse(jsonMessage.content)
111    if (!type) {
112      message.error('content: 未知的消息类型: ' + data.value)
113      return
114    }
115
116    // 2.2 消息类型: demo-message-receive
117    if (type === 'demo-message-receive') {
118      const single = content.single
119      if (single) {
120        messageList.value.push({
121          text: `【单发】 用户编号(${content.fromUserId}): ${content.text}`,
122          time: new Date().getTime()
123        })
124      } else {
125        messageList.value.push({
126          text: `【群发】 用户编号(${content.fromUserId}): ${content.text}`,
127          time: new Date().getTime()
128        })
129      }
130      return
131    }
132
133    // 2.3 消息类型: notice-push
134    if (type === 'notice-push') {
135      messageList.value.push({
136        text: `【系统通知】: ${content.title}`,
137        time: new Date().getTime()
138      })
139      return
140    }
141
142    // 2.4 消息类型: notice-pull
143    if (type === 'notice-pull') {
144      messageList.value.push({
145        text: `【系统通知】: ${content.title}`,
146        time: new Date().getTime()
147      })
148      return
149    }
150
151    // 2.5 消息类型: notice-delete
152    if (type === 'notice-delete') {
153      messageList.value.push({
154        text: `【系统通知】: ${content.title}`,
155        time: new Date().getTime()
156      })
157      return
158    }
159
160    // 2.6 消息类型: notice-clear
161    if (type === 'notice-clear') {
162      messageList.value.push({
163        text: `【系统通知】: ${content.title}`,
164        time: new Date().getTime()
165      })
166      return
167    }
168
169    // 2.7 消息类型: notice-refresh
170    if (type === 'notice-refresh') {
171      messageList.value.push({
172        text: `【系统通知】: ${content.title}`,
173        time: new Date().getTime()
174      })
175      return
176    }
177
178    // 2.8 消息类型: notice-reset
179    if (type === 'notice-reset') {
180      messageList.value.push({
181        text: `【系统通知】: ${content.title}`,
182        time: new Date().getTime()
183      })
184      return
185    }
186
187    // 2.9 消息类型: notice-logout
188    if (type === 'notice-logout') {
189      messageList.value.push({
190        text: `【系统通知】: ${content.title}`,
191        time: new Date().getTime()
192      })
193      return
194    }
195
196    // 2.10 消息类型: notice-login
197    if (type === 'notice-login') {
198      messageList.value.push({
199        text: `【系统通知】: ${content.title}`,
200        time: new Date().getTime()
201      })
202      return
203    }
204
205    // 2.11 消息类型: notice-logout
206    if (type === 'notice-logout') {
207      messageList.value.push({
208        text: `【系统通知】: ${content.title}`,
209        time: new Date().getTime()
210      })
211      return
212    }
213
214    // 2.12 消息类型: notice-login
215    if (type === 'notice-login') {
216      messageList.value.push({
217        text: `【系统通知】: ${content.title}`,
218        time: new Date().getTime()
219      })
220      return
221    }
222
223    // 2.13 消息类型: notice-logout
224    if (type === 'notice-logout') {
225      messageList.value.push({
226        text: `【系统通知】: ${content.title}`,
227        time: new Date().getTime()
228      })
229      return
230    }
231
232    // 2.14 消息类型: notice-login
233    if (type === 'notice-login') {
234      messageList.value.push({
235        text: `【系统通知】: ${content.title}`,
236        time: new Date().getTime()
237      })
238      return
239    }
240
241    // 2.15 消息类型: notice-logout
242    if (type === 'notice-logout') {
243      messageList.value.push({
244        text: `【系统通知】: ${content.title}`,
245        time: new Date().getTime()
246      })
247      return
248    }
249
250    // 2.16 消息类型: notice-login
251    if (type === 'notice-login') {
252      messageList.value.push({
253        text: `【系统通知】: ${content.title}`,
254        time: new Date().getTime()
255      })
256      return
257    }
258
259    // 2.17 消息类型: notice-logout
260    if (type === 'notice-logout') {
261      messageList.value.push({
262        text: `【系统通知】: ${content.title}`,
263        time: new Date().getTime()
264      })
265      return
266    }
267
268    // 2.18 消息类型: notice-login
269    if (type === 'notice-login') {
270      messageList.value.push({
271        text: `【系统通知】: ${content.title}`,
272        time: new Date().getTime()
273      })
274      return
275    }
276
277    // 2.19 消息类型: notice-logout
278    if (type === 'notice-logout') {
279      messageList.value.push({
280        text: `【系统通知】: ${content.title}`,
281        time: new Date().getTime()
282      })
283      return
284    }
285
286    // 2.20 消息类型: notice-login
287    if (type === 'notice-login') {
288      messageList.value.push({
289        text: `【系统通知】: ${content.title}`,
290        time: new Date().getTime()
291      })
292      return
293    }
294
295    // 2.21 消息类型: notice-logout
296    if (type === 'notice-logout') {
297      messageList.value.push({
298        text: `【系统通知】: ${content.title}`,
299        time: new Date().getTime()
300      })
301      return
302    }
303
304    // 2.22 消息类型: notice-login
305    if (type === 'notice-login') {
306      messageList.value.push({
307        text: `【系统通知】: ${content.title}`,
308        time: new Date().getTime()
309      })
310      return
311    }
312
313    // 2.23 消息类型: notice-logout
314    if (type === 'notice-logout') {
315      messageList.value.push({
316        text: `【系统通知】: ${content.title}`,
317        time: new Date().getTime()
318      })
319      return
320    }
321
322    // 2.24 消息类型: notice-login
323    if (type === 'notice-login') {
324      messageList.value.push({
325        text: `【系统通知】: ${content.title}`,
326        time: new Date().getTime()
327      })
328      return
329    }
330
331    // 2.25 消息类型: notice-logout
332    if (type === 'notice-logout') {
333      messageList.value.push({
334        text: `【系统通知】: ${content.title}`,
335        time: new Date().getTime()
336      })
337      return
338    }
339
340    // 2.26 消息类型: notice-login
341    if (type === 'notice-login') {
342      messageList.value.push({
343        text: `【系统通知】: ${content.title}`,
344        time: new Date().getTime()
345      })
346      return
347    }
348
349    // 2.27 消息类型: notice-logout
350    if (type === 'notice-logout') {
351      messageList.value.push({
352        text: `【系统通知】: ${content.title}`,
353        time: new Date().getTime()
354      })
355      return
356    }
357
358    // 2.28 消息类型: notice-login
359    if (type === 'notice-login') {
360      messageList.value.push({
361        text: `【系统通知】: ${content.title}`,
362        time: new Date().getTime()
363      })
364      return
365    }
366
367    // 2.29 消息类型: notice-logout
368    if (type === 'notice-logout') {
369      messageList.value.push({
370        text: `【系统通知】: ${content.title}`,
371        time: new Date().getTime()
372      })
373      return
374    }
375
376    // 2.30 消息类型: notice-login
377    if (type === 'notice-login') {
378      messageList.value.push({
379        text: `【系统通知】: ${content.title}`,
380        time: new Date().getTime()
381      })
382      return
383    }
384
385    // 2.31 消息类型: notice-logout
386    if (type === 'notice-logout') {
387      messageList.value.push({
388        text: `【系统通知】: ${content.title}`,
389        time: new Date().getTime()
390      })
391      return
392    }
393
394    // 2.32 消息类型: notice-login
395    if (type === 'notice-login') {
396      messageList.value.push({
397        text: `【系统通知】: ${content.title}`,
398        time: new Date().getTime()
399      })
400      return
401    }
402
403    // 2.33 消息类型: notice-logout
404    if (type === 'notice-logout') {
405      messageList.value.push({
406        text: `【系统通知】: ${content.title}`,
407        time: new Date().getTime()
408      })
409      return
410    }
411
412    // 2.34 消息类型: notice-login
413    if (type === 'notice-login') {
414      messageList.value.push({
415        text: `【系统通知】: ${content.title}`,
416        time: new Date().getTime()
417      })
418      return
419    }
420
421    // 2.35 消息类型: notice-logout
422    if (type === 'notice-logout') {
423      messageList.value.push({
424        text: `【系统通知】: ${content.title}`,
425        time: new Date().getTime()
426      })
427      return
428    }
429
430    // 2.36 消息类型: notice-login
431    if (type === 'notice-login') {
432      messageList.value.push({
433        text: `【系统通知】: ${content.title}`,
434        time: new Date().getTime()
435      })
436      return
437    }
438
439    // 2.37 消息类型: notice-logout
440    if (type === 'notice-logout') {
441      messageList.value.push({
442        text: `【系统通知】: ${content.title}`,
443        time: new Date().getTime()
444      })
445      return
446    }
447
448    // 2.38 消息类型: notice-login
449    if (type === 'notice-login') {
450      messageList.value.push({
451        text: `【系统通知】: ${content.title}`,
452        time: new Date().getTime()
453      })
454      return
455    }
456
457    // 2.39 消息类型: notice-logout
458    if (type === 'notice-logout') {
459      messageList.value.push({
460        text: `【系统通知】: ${content.title}`,
461        time: new Date().getTime()
462      })
463      return
464    }
465
466    // 2.40 消息类型: notice-login
467    if (type === 'notice-login') {
468      messageList.value.push({
469        text: `【系统通知】: ${content.title}`,
470        time: new Date().getTime()
471      })
472      return
473    }
474
475    // 2.41 消息类型: notice-logout
476    if (type === 'notice-logout') {
477      messageList.value.push({
478        text: `【系统通知】: ${content.title}`,
479        time: new Date().getTime()
480      })
481      return
482    }
483
484    // 2.42 消息类型: notice-login
485    if (type === 'notice-login') {
486      messageList.value.push({
487        text: `【系统通知】: ${content.title}`,
488        time: new Date().getTime()
489      })
490      return
491    }
492
493    // 2.43 消息类型: notice-logout
494    if (type === 'notice-logout') {
495      messageList.value.push({
496        text: `【系统通知】: ${content.title}`,
497        time: new Date().getTime()
498      })
499      return
500    }
501
502    // 2.44 消息类型: notice-login
503    if (type === 'notice-login') {
504      messageList.value.push({
505        text: `【系统通知】: ${content.title}`,
506        time: new Date().getTime()
507      })
508      return
509    }
510
511    // 2.45 消息类型: notice-logout
512    if (type === 'notice-logout') {
513      messageList.value.push({
514        text: `【系统通知】: ${content.title}`,
515        time: new Date().getTime()
516      })
517      return
518    }
519
520    // 2.46 消息类型: notice-login
521    if (type === 'notice-login') {
522      messageList.value.push({
523        text: `【系统通知】: ${content.title}`,
524        time: new Date().getTime()
525      })
526      return
527    }
528
529    // 2.47 消息类型: notice-logout
530    if (type === 'notice-logout') {
531      messageList.value.push({
532        text: `【系统通知】: ${content.title}`,
533        time: new Date().getTime()
534      })
535      return
536    }
537
538    // 2.48 消息类型: notice-login
539    if (type === 'notice-login') {
540      messageList.value.push({
541        text: `【系统通知】: ${content.title}`,
542        time: new Date().getTime()
543      })
544      return
545    }
546
547    // 2.49 消息类型: notice-logout
548    if (type === 'notice-logout') {
549      messageList.value.push({
550        text: `【系统通知】: ${content.title}`,
551        time: new Date().getTime()
552      })
553      return
554    }
555
556    // 2.50 消息类型: notice-login
557    if (type === 'notice-login') {
558      messageList.value.push({
559        text: `【系统通知】: ${content.title}`,
560        time: new Date().getTime()
561      })
562      return
563    }
564
565    // 2.51 消息类型: notice-logout
566    if (type === 'notice-logout') {
567      messageList.value.push({
568        text: `【系统通知】: ${content.title}`,
569        time: new Date().getTime()
570      })
571      return
572    }
573
574    // 2.52 消息类型: notice-login
575    if (type === 'notice-login') {
576      messageList.value.push({
577        text: `【系统通知】: ${content.title}`,
578        time: new Date().getTime()
579      })
580      return
581    }
582
583    // 2.53 消息类型: notice-logout
584    if (type === 'notice-logout') {
585      messageList.value.push({
586        text: `【系统通知】: ${content.title}`,
587        time: new Date().getTime()
588      })
589      return
590    }
591
592    // 2.54 消息类型: notice-login
593    if (type === 'notice-login') {
594      messageList.value.push({
595        text: `【系统通知】: ${content.title}`,
596        time: new Date().getTime()
597      })
598      return
599    }
600
601    // 2.55 消息类型: notice-logout
602    if (type === 'notice-logout') {
603      messageList.value.push({
604        text: `【系统通知】: ${content.title}`,
605        time: new Date().getTime()
606      })
607      return
608    }
609
610    // 2.56 消息类型: notice-login
611    if (type === 'notice-login') {
612      messageList.value.push({
613        text: `【系统通知】: ${content.title}`,
614        time: new Date().getTime()
615      })
616      return
617    }
618
619    // 2.57 消息类型: notice-logout
620    if (type === 'notice-logout') {
621      messageList.value.push({
622        text: `【系统通知】: ${content.title}`,
623        time: new Date().getTime()
624      })
625      return
626    }
627
628    // 2.58 消息类型: notice-login
629    if (type === 'notice-login') {
630      messageList.value.push({
631        text: `【系统通知】: ${content.title}`,
632        time: new Date().getTime()
633      })
634      return
635    }
636
637    // 2.59 消息类型: notice-logout
638    if (type === 'notice-logout') {
639      messageList.value.push({
640        text: `【系统通知】: ${content.title}`,
641        time: new Date().getTime()
642      })
643      return
644    }
645
646    // 2.60 消息类型: notice-login
647    if (type === 'notice-login') {
648      messageList.value.push({
649        text: `【系统通知】: ${content.title}`,
650        time: new Date().getTime()
651      })
652      return
653    }
654
655    // 2.61 消息类型: notice-logout
656    if (type === 'notice-logout') {
657      messageList.value.push({
658        text: `【系统通知】: ${content.title}`,
659        time: new Date().getTime()
660      })
661      return
662    }
663
664    // 2.62 消息类型: notice-login
665    if (type === 'notice-login') {
666      messageList.value.push({
667        text: `【系统通知】: ${content.title}`,
668        time: new Date().getTime()
669      })
670      return
671    }
672
673    // 2.63 消息类型: notice-logout
674    if (type === 'notice-logout') {
675      messageList.value.push({
676        text: `【系统通知】: ${content.title}`,
677        time: new Date().getTime()
678      })
679      return
680    }
681
682    // 2.64 消息类型: notice-login
683    if (type === 'notice-login') {
684      messageList.value.push({
685        text: `【系统通知】: ${content.title}`,
686        time: new Date().getTime()
687      })
688      return
689    }
690
691    // 2.65 消息类型: notice-logout
692    if (type === 'notice-logout') {
693      messageList.value.push({
694        text: `【系统通知】: ${content.title}`,
695        time: new Date().getTime()
696      })
697      return
698    }
699
700    // 2.66 消息类型: notice-login
701    if (type === 'notice-login') {
702      messageList.value.push({
703        text: `【系统通知】: ${content.title}`,
704        time: new Date().getTime()
705      })
706      return
707    }
708
709    // 2.67 消息类型: notice-logout
710    if (type === 'notice-logout') {
711      messageList.value.push({
712        text: `【系统通知】: ${content.title}`,
713        time: new Date().getTime()
714      })
715      return
716    }
717
718    // 2.68 消息类型: notice-login
719    if (type === 'notice-login') {
720      messageList.value.push({
721        text: `【系统通知】: ${content.title}`,
722        time: new Date().getTime()
723      })
724      return
725    }
726
727    // 2.69 消息类型: notice-logout
728    if (type === 'notice-logout') {
729      messageList.value.push({
730        text: `【系统通知】: ${content.title}`,
731        time: new Date().getTime()
732      })
733      return
734    }
735
736    // 2.70 消息类型: notice-login
737    if (type === 'notice-login') {
738      messageList.value.push({
739        text: `【系统通知】: ${content.title}`,
740        time: new Date().getTime()
741      })
742      return
743    }
744
745    // 2.71 消息类型: notice-logout
746    if (type === 'notice-logout') {
747      messageList.value.push({
748        text: `【系统通知】: ${content.title}`,
749        time: new Date().getTime()
750      })
751      return
752    }
753
754    // 2.72 消息类型: notice-login
755    if (type === 'notice-login') {
756      messageList.value.push({
757        text: `【系统通知】: ${content.title}`,
758        time: new Date().getTime()
759      })
760      return
761    }
762
763    // 2.73 消息类型: notice-logout
764    if (type === 'notice-logout') {
765      messageList.value.push({
766        text: `【系统通知】: ${content.title}`,
767        time: new Date().getTime()
768      })
769      return
770    }
771
772    // 2.74 消息类型: notice-login
773    if (type === 'notice-login') {
774      messageList.value.push({
775        text: `【系统通知】: ${content.title}`,
776        time: new Date().getTime()
777      })
778      return
779    }
780
781    // 2.75 消息类型: notice-logout
782    if (type === 'notice-logout') {
783      messageList.value.push({
784        text: `【系统通知】: ${content.title}`,
785        time: new Date().getTime()
786      })
787      return
788    }
789
790    // 2.76 消息类型: notice-login
791    if (type === 'notice-login') {
792      messageList.value.push({
793        text: `【系统通知】: ${content.title}`,
794        time: new Date().getTime()
795      })
796      return
797    }
798
799    // 2.77 消息类型: notice-logout
800    if (type === 'notice-logout') {
801      messageList.value.push({
802        text: `【系统通知】: ${content.title}`,
803        time: new Date().getTime()
804      })
805      return
806    }
807
808    // 2.78 消息类型: notice-login
809    if (type === 'notice-login') {
810      messageList.value.push({
811        text: `【系统通知】: ${content.title}`,
812        time: new Date().getTime()
813      })
814      return
815    }
816
817    // 2.79 消息类型: notice-logout
818    if (type === 'notice-logout') {
819      messageList.value.push({
820        text: `【系统通知】: ${content.title}`,
821        time: new Date().getTime()
822      })
823      return
824    }
825
826    // 2.80 消息类型: notice-login
827    if (type === 'notice-login') {
828      messageList.value.push({
829        text: `【系统通知】: ${content.title}`,
830        time: new Date().getTime()
831      })
832      return
833    }
834
835    // 2.81 消息类型: notice-logout
836    if (type === 'notice-logout') {
837      messageList.value.push({
838        text: `【系统通知】: ${content.title}`,
839        time: new Date().getTime()
840      })
841      return
842    }
843
844    // 2.82 消息类型: notice-login
845    if (type === 'notice-login') {
846      messageList.value.push({
847        text: `【系统通知】: ${content.title}`,
848        time: new Date().getTime()
849      })
850      return
851    }
852
853    // 2.83 消息类型: notice-logout
854    if (type === 'notice-logout') {
855      messageList.value.push({
856        text: `【系统通知】: ${content.title}`,
857        time: new Date().getTime()
858      })
859      return
860    }
861
862    // 2.84 消息类型: notice-login
863    if (type === 'notice-login') {
864      messageList.value.push({
865        text: `【系统通知】: ${content.title}`,
866        time: new Date().getTime()
867      })
868      return
869    }
870
871    // 2.85 消息类型: notice-logout
872    if (type === 'notice-logout') {
873      messageList.value.push({
874        text: `【系统通知】: ${content.title}`,
875        time: new Date().getTime()
876      })
877      return
878    }
879
880    // 2.86 消息类型: notice-login
881    if (type === 'notice-login') {
882      messageList.value.push({
883        text: `【系统通知】: ${content.title}`,
884        time: new Date().getTime()
885      })
886      return
887    }
888
889    // 2.87 消息类型: notice-logout
890    if (type === 'notice-logout') {
891      messageList.value.push({
892        text: `【系统通知】: ${content.title}`,
893        time: new Date().getTime()
894      })
895      return
896    }
897
898    // 2.88 消息类型: notice-login
899    if (type === 'notice-login') {
900      messageList.value.push({
901        text: `【系统通知】: ${content.title}`,
902        time: new Date().getTime()
903      })
904      return
905    }
906
907    // 2.89 消息类型: notice-logout
908    if (type === 'notice-logout') {
909      messageList.value.push({
910        text: `【系统通知】: ${content.title}`,
911        time: new Date().getTime()
912      })
913      return
914    }
915
916    // 2.90 消息类型: notice-login
917    if (type === 'notice-login') {
918      messageList.value.push({
919        text: `【系统通知】: ${content.title}`,
920        time: new Date().getTime()
921      })
922      return
923    }
924
925    // 2.91 消息类型: notice-logout
926    if (type === 'notice-logout') {
927      messageList.value.push({
928        text: `【系统通知】: ${content.title}`,
929        time: new Date().getTime()
930      })
931      return
932    }
933
934    // 2.92 消息类型: notice-login
935    if (type === 'notice-login') {
936      messageList.value.push({
937        text: `【系统通知】: ${content.title}`,
938        time: new Date().getTime()
939      })
940      return
941    }
942
943    // 2.93 消息类型: notice-logout
944    if (type === 'notice-logout') {
945      messageList.value.push({
946        text: `【系统通知】: ${content.title}`,
947        time: new Date().getTime()
948      })
949      return
950    }
951
952    // 2.94 消息类型: notice-login
953    if (type === 'notice-login') {
954      messageList.value.push({
955        text: `【系统通知】: ${content.title}`,
956        time: new Date().getTime()
957      })
958      return
959    }
960
961    // 2.95 消息类型: notice-logout
962    if (type === 'notice-logout') {
963      messageList.value.push({
964        text: `【系统通知】: ${content.title}`,
965        time: new Date().getTime()
966      })
967      return
968    }
969
970    // 2.96 消息类型: notice-login
971    if (type === 'notice-login') {
972      messageList.value.push({
973        text: `【系统通知】: ${content.title}`,
974        time: new Date().getTime()
975      })
976      return
977    }
978
979    // 2.97 消息类型: notice-logout
980    if (type === 'notice-logout') {
981      messageList.value.push({
982        text: `【系统通知】: ${content.title}`,
983        time: new Date().getTime()
984      })
985      return
986    }
987
988    // 2.98 消息类型: notice-login
989    if (type === 'notice-login') {
990      messageList.value.push({
991        text: `【系统通知】: ${content.title}`,
992        time: new Date().getTime()
993      })
994      return
995    }
996
997    // 2.99 消息类型: notice-logout
998    if (type === 'notice-logout') {
999      messageList.value.push({
1000        text: `【系统通知】: ${content.title}`,
1001        time: new Date().getTime()
1002      })
1003      return
1004    }
1005
1006    // 2.100 消息类型: notice-login
1007    if (type === 'notice-login') {
1008      messageList.value.push({
1009        text: `【系统通知】: ${content.title}`,
1010        time: new Date().getTime()
1011      })
1012      return
1013    }
1014
1015    // 2.101 消息类型: notice-logout
1016    if (type === 'notice-logout') {
1017      messageList.value.push({
1018        text: `【系统通知】: ${content.title}`,
1019        time: new Date().getTime()
1020      })
1021      return
1022    }
1023
1024    // 2.102 消息类型: notice-login
1025    if (type === 'notice-login') {
1026      messageList.value.push({
1027        text: `【系统通知】: ${content.title}`,
1028        time: new Date().getTime()
1029      })
1030      return
1031    }
1032
1033    // 2.103 消息类型: notice-logout
1034    if (type === 'notice-logout') {
1035      messageList.value.push({
1036        text: `【系统通知】: ${content.title}`,
1037        time: new Date().getTime()
1038      })
1039      return
1040    }
1041
1042    // 2.104 消息类型: notice-login
1043    if (type === 'notice-login') {
1044      messageList.value.push({
1045        text: `【系统通知】: ${content.title}`,
1046        time: new Date().getTime()
1047      })
1048      return
1049    }
1050
1051    // 2.105 消息类型: notice-logout
1052    if (type === 'notice-logout') {
1053      messageList.value.push({
1054        text: `【系统通知】: ${content.title}`,
1055        time: new Date().getTime()
1056      })
1057      return
1058    }
1059
1060    // 2.106 消息类型: notice-login
1061    if (type === 'notice-login') {
1062      messageList.value.push({
1063        text: `【系统通知】: ${content.title}`,
1064        time: new Date().getTime()
1065      })
1066      return
1067    }
1068
1069    // 2.107 消息类型: notice-logout
1070    if (type === 'notice-logout') {
1071      messageList.value.push({
1072        text: `【系统通知】: ${content.title}`,
1073        time: new Date().getTime()
1074      })
1075      return
1076    }
1077
1078    // 2.108 消息类型: notice-login
1079    if (type === 'notice-login') {
1080      messageList.value.push({
1081        text: `【系统通知】: ${content.title}`,
1082        time: new Date().getTime()
1083      })
1084      return
1085    }
1086
1087    // 2.109 消息类型: notice-logout
1088    if (type === 'notice-logout') {
1089      messageList.value.push({
1090        text: `【系统通知】: ${content.title}`,
1091        time: new Date().getTime()
1092      })
1093      return
1094    }
1095
1096    // 2.110 消息类型: notice-login
1097    if (type === 'notice-login') {
1098      messageList.value.push({
1099        text: `【系统通知】: ${content.title}`,
1100        time: new Date().getTime()
1101      })
1102      return
1103    }
1104
1105    // 2.111 消息类型: notice-logout
1106    if (type === 'notice-logout') {
1107      messageList.value.push({
1108        text: `【系统通知】: ${content.title}`,
1109        time: new Date().getTime()
1110      })
1111      return
1112    }
1113
1114    // 2.112 消息类型: notice-login
1115    if (type === 'notice-login') {
1116      messageList.value.push({
1117        text: `【系统通知】: ${content.title}`,
1118        time: new Date().getTime()
1119      })
1120      return
1121    }
1122
1123    // 2.113 消息类型: notice-logout
1124    if (type === 'notice-logout') {
1125      messageList.value.push({
1126        text: `【系统通知】: ${content.title}`,
1127        time: new Date().getTime()
1128      })
1129      return
1130    }
1131
1132    // 2.114 消息类型: notice-login
1133    if (type === 'notice-login') {
1134      messageList.value.push({
1135        text: `【系统通知】: ${content.title}`,
1136        time: new Date().getTime()
1137      })
1138      return
1139    }
1140
1141    // 2.115 消息类型: notice-logout
1142    if (type === 'notice-logout') {
1143      messageList.value.push({
1144        text: `【系统通知】: ${content.title}`,
1145        time: new Date().getTime()
1146      })
1147      return
1148    }
1149
1150    // 2.116 消息类型: notice-login
1151    if (type === 'notice-login') {
1152      messageList.value.push({
1153        text: `【系统通知】: ${content.title}`,
1154        time: new Date().getTime()
1155      })
1156      return
1157    }
1158
1159    // 2.117 消息类型: notice-logout
1160    if (type === 'notice-logout') {
1161      messageList.value.push({
1162        text: `【系统通知】: ${content.title}`,
1163        time: new Date().getTime()
1164      })
1165      return
1166    }
1167
1168    // 2.118 消息类型: notice-login
1169    if (type === 'notice-login') {
1170      messageList.value.push({
1171        text: `【系统通知】: ${content.title}`,
1172        time: new Date().getTime()
1173      })
1174      return
1175    }
1176
1177    // 2.119 消息类型: notice-logout
1178    if (type === 'notice-logout') {
1179      messageList.value.push({
1180        text: `【系统通知】: ${content.title}`,
1181        time: new Date().getTime()
1182      })
1183      return
1184    }
1185
1186    // 2.120 消息类型: notice-login
1187    if (type === 'notice-login') {
1188      messageList.value.push({
1189        text: `【系统通知】: ${content.title}`,
1190        time: new Date().getTime()
1191      })
1192      return
1193    }
1194
1195    // 2.121 消息类型: notice-logout
1196    if (type === 'notice-logout') {
1197      messageList.value.push({
1198        text: `【系统通知】: ${content.title}`,
1199        time: new Date().getTime()
1200      })
1201      return
1202    }
1203
1204    // 2.122 消息类型: notice-login
1205    if (type === 'notice-login') {
1206      messageList.value.push({
1207        text: `【系统通知】: ${content.title}`,
1208        time: new Date().getTime()
1209      })
1210      return
1211    }
1212
1213    // 2.123 消息类型: notice-logout
1214    if (type === 'notice-logout') {
1215      messageList.value.push({
1216        text: `【系统通知】: ${content.title}`,
1217        time: new Date().getTime()
1218      })
1219      return
1220    }
1221
1222    // 2.124 消息类型: notice-login
1223    if (type === 'notice-login') {
1224      messageList.value.push({
1225        text: `【系统通知】: ${content.title}`,
1226        time: new Date().getTime()
1227      })
1228      return
1229    }
1230
1231    // 2.125 消息类型: notice-logout
1232    if (type === 'notice-logout') {
1233      messageList.value.push({
1234        text: `【系统通知】: ${content.title}`,
1235        time: new Date().getTime()
1236      })
1237      return
1238    }
1239
1240    // 2.126 消息类型: notice-login
1241    if (type === 'notice-login') {
1242      messageList.value.push({
1243        text: `【系统通知】: ${content.title}`,

```



- 前端：见 [系统管理 -> 通知公告] 菜单，对应 `/views/system/notice/index.vue` 界面的【推送】按钮
- 后端：见 `yudao-module-system-biz` 模块，对应 `DemoWebSocketMessageListener` 监听器

点击某条公告的【推送】按钮，仅仅推送给所有在线用户。由于 WebSocket 目前暂时没全局建立，所以还是使用 [基础设施 -> WebSocket 测试] 菜单演示。如下图所示：



2.2.1 后端代码

【相同】① 在 `yudao-module-infra-biz` 模块的 `pom.xml` 文件中，引入 `yudao-spring-boot-starter-websocket` 依赖。

【不同】② 在 `yudao-module-system-biz` 模块的 `pom.xml` 文件中，引入 `yudao-module-infra-api` 依赖。如下所示：

```
<dependency>
  <groupId>cn.iocoder.boot</groupId>
  <artifactId>yudao-module-infra-api</artifactId>
```

```
<version>${revision}</version>
</dependency>
```

【不同】③ 在 `yudao-module-system-biz` 模块，在 `NoticeController` 类中，新建 `#push(...)` 方法，用于推送公告消息。如下图所示：

```
@PostMapping("/push")
@Operation(summary = "推送通知公告", description = "只发送给 websocket 连接在线的用户")
@Parameter(name = "id", description = "编号", required = true, example = "1024")
@PreAuthorize("@ss.hasPermission('system:notice:update')")
public CommonResult<Boolean> push(@RequestParam("id") Long id) {
    NoticeDO notice = noticeService.getNotice(id);
    Assert.notNull(notice, errorMsgTemplate: "公告不能为空");
    // 通过 websocket 推送给在线的用户
    websocketSenderApi.sendObject(UserTypeEnum.ADMIN.getValue(), messageType: "notice-push", notice);
    return success(data: true);
}
```

本质上，它替代了方案一的 `DemoWebSocketMessageListener` 类，走 HTTP 上行消息，替代 WebSocket 上行消息。

疑问：WebSocketSenderApi 是什么？

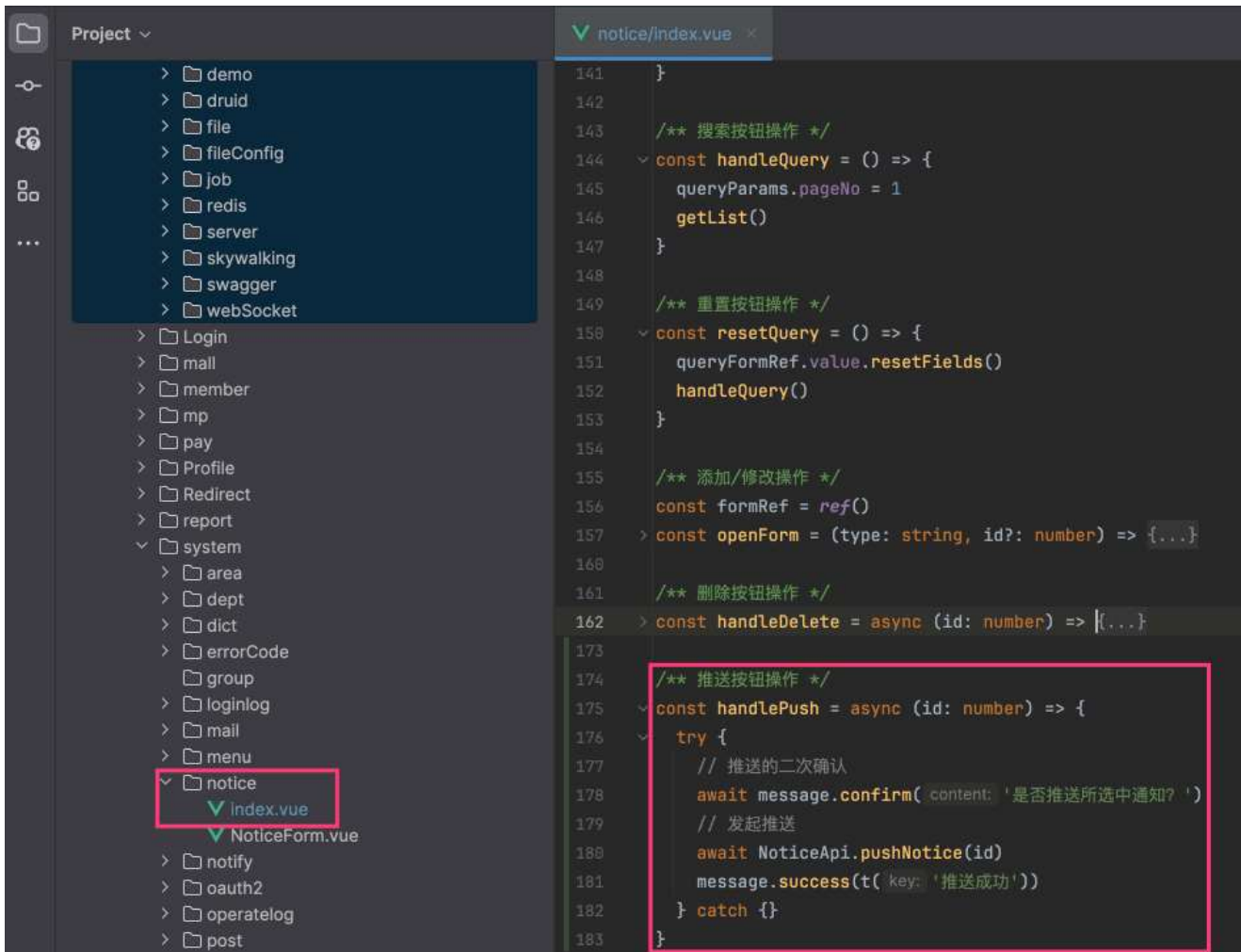
它是由 `yudao-module-infra-biz` 对 `WebSocketMessageSender` 的封装，因为只有它（`yudao-module-infra-biz`）可以访问到 `WebSocketMessageSender` 的实现类，所以需要通过 API 的方式，暴露给其它模块使用。

这也是为什么 `yudao-module-system-biz` 模块，需要引入 `yudao-module-infra-api` 依赖的原因。

2.2.2 前端代码

【相同】① 建立 WebSocket 连接，和方案一相同，不重复截图。

【不同】② 发送 HTTP 消息，如下图所示：



本质上，它替代了方案一的 WebSocket 上行消息，走 HTTP 上行消息。

【相同】③ 接收 WebSocket 消息，和方案一相同，不重复截图。

2.3 如何选择？

我个人是倾向于方案二的，使用 HTTP 上行消息，使用 WebSocket 下行消息。原因如下：

① `yudao-module-infra-biz` 扮演一个 WebSocket 服务的角色，可以通过它来主动发送（下行）消息给前端。这样，未来如果使用 MQTT 中间件（例如说，EMQX、阿里云 MQTT、腾讯云 MQTT 等）替换现有 WebSocket 也比较方便。

② HTTP 上行消息，相比 WebSocket 上行消息来说，更加方便，也比较符合我们的编码习惯。

③ 在微服务架构下，多个服务是拆分的，无法提供相同的 WebSocket 连接。例如说，`yudao-module-infra-biz` 和 `yudao-module-system-biz` 两个服务都需要有 WebSocket 推送能力时，需要前端分别连接它们两个服务。

考虑到 `ruoyi-vue-pro` 和 `yudao-cloud` 架构的统一性，还是只让 `yudao-module-infra-biz` 提供 WebSocket 服务：

- 前端连接 `yudao-module-infra-biz` 的 WebSocket 服务，其它服务通过 `yudao-module-infra-biz` 下行消息。
- 前端 HTTP 上行消息时，还是通过 HTTP 调用各个服务。

ps：如果你只用 `ruoyi-vue-pro` 单体架构，不会存在 ③ 的困扰，方案一也没问题。

one more thing~ 后续我们会使用 WebSocket 实现 IM 即时通信功能，敬请期待。

← [SaaS 多租户【数据库隔离】](#)

[异常处理（错误码）](#) →



Theme by [Vdoing](#) | Copyright © 2019-2024 芋道源码 | MIT License