**API校订版**

返回消息格式概述

1、type、action、data.status字段

（1）字段含义——成功与失败的判断标志

1. type字段 - 消息的业务类型

作用：告诉手机端这是什么业务消息

含义：

* 表示这个消息属于哪个业务流程
* 手机端根据type字段知道这是哪个业务操作的结果

2. action字段 - 操作执行结果

作用：告诉手机端这个操作的结果如何

含义：

* 表示这个业务操作执行的结果状态
* 手机端根据action字段知道操作是成功、失败、超时还是取消

3. data.status字段 - 具体业务状态

作用：告诉手机端具体的业务状态细节含义：

* 在action的基础上，提供更具体的业务状态信息
* 手机端可以根据这个字段做更精细的处理

（2）层级关系

type (业务类型) → action (操作结果) → data.status (具体状态)

（3）reference总表

注：手机端可忽略表格第一列

1. type字段 - 消息业务类型

|  |  |  |
| --- | --- | --- |
| 常量名 | 值 | 含义 |
| 系统消息类型 |  |  |
| CONNECTION | 'connection' | 连接相关消息 |
| ERROR | 'error' | 错误消息 |
| 入库流程消息类型 |  |  |
| INBOUND\_FOOD\_NUMBER | 'inbound\_food\_number' | 食品数量设置 |
| INBOUND\_READY | 'inbound\_ready' | 准备完成 |
| INBOUND\_RFID\_READING | 'inbound\_rfid\_reading' | RFID读取 |
| INBOUND\_RFID\_RESULT | 'inbound\_rfid\_result' | RFID读取结果 |
| INBOUND\_OCR\_PROCESS | 'inbound\_info\_process' | 信息处理请求 |
| INBOUND\_UPLOAD\_SUCCESS | 'inbound\_upload\_success' | 上传成功确认 |
| INBOUND\_END | 'inbound\_end' | 入库结束 |
| INBOUND\_CANCEL | 'inbound\_cancel' | 中止入库请求 |
| 出库流程消息类型 |  |  |
| OUTBOUND\_NOTIFICATION | 'outbound\_notification' | 出库通知（树莓派主动推送） |

2. action字段 - 操作执行结果

|  |  |  |
| --- | --- | --- |
| 常量名 | 值 | 含义 |
| 结果类Action |  |  |
| SUCCESS | 'success' | 完全成功 |
| FAILED | 'failed' | 完全失败 |
| TIMEOUT | 'timeout' | 超时 |
| 数据类Action |  |  |
| IMAGES | 'images' | 图片处理完成 |
| DATA | 'data' | 数据通知 |
| 通知类Action |  |  |
| ERROR | 'error' | 错误 |

3. data.status字段 - 具体业务状态

|  |  |  |
| --- | --- | --- |
| 常量名 | 值 | 含义 |
| 系统状态 |  |  |
| CONNECTED | 'connected' | 连接成功 |
| ERROR | 'error' | 错误状态 |
| 入库流程状态 |  |  |
| READY | 'ready' | 准备完成 |
| RFID\_READING\_SUCCESS | 'rfid\_reading\_success' | RFID读取成功 |
| RFID\_READING\_TIMEOUT | 'rfid\_reading\_timeout' | RFID读取超时 |
| DATA\_READY | 'data\_ready' | 数据准备完成 |
| IMAGES\_READY | 'images\_ready' | 图片处理完成 |
| UPLOAD\_SUCCESS | 'upload\_success' | 上传成功 |
| END\_SUCCESS | 'end\_success' | 入库结束成功 |
| CANCELLED | 'cancelled' | 取消成功 |
| 出库流程状态 |  |  |
| DELETE\_FOOD\_ITEM | 'delete\_food\_item' | 删除食品条目 |
| SHOW\_ERROR\_NOTIFICATION | 'show\_error\_notification' | 显示错误通知 |

2、正常返回格式

json

{

    "type": "业务消息类型",

    "action": "操作结果",

    "data": {

        "status": "具体状态",

        "其他业务数据": "具体值"

    },

    "timestamp": "2025-01-08T12:35:30.456789"

}

3、异常返回格式

（1）消息返回格式：

json

{

    "type": "error",

    "action": "error",

    "data": {

        "status": "error",

        "error\_message": "具体错误信息"

    },

    "timestamp": "2025-01-08T12:35:30.456789"

}

（2）错误判断逻辑

检查 type 字段是否为 "error"

检查 action 字段是否为 "error"

（检查 data.status 字段是否为 "error"；一般只检查前两个即可）

（3）有效数据字段，内容及含意

data.error\_message - 具体错误信息，用于显示给用户

|  |  |  |  |
| --- | --- | --- | --- |
| 错误分类 | 错误信息 | 含义 | 触发场景 |
| 连接相关错误 |  |  |  |
| 连接注册失败 | "连接注册失败" | WebSocket连接建立后，注册到服务器时失败 | 连接建立后注册过程出错 |
| 消息格式错误 |  |  |  |
| 无效的JSON格式 | "无效的JSON格式" | 手机端发送的不是有效JSON格式 | JSON解析失败 |
| 未知消息类型 | "未知消息类型" | 手机端发送了服务器不认识的消息类型 | 消息类型不在支持列表中 |
| 系统处理错误 |  |  |  |
| 通用系统异常 | str(e) | 通用系统异常，包含具体的异常信息 | 第215行，处理消息时出现未预期异常 |
| 业务操作错误 |  |  |  |
| 保存图片失败 | "保存图片失败" | 服务器无法保存上传的图片文件 | 图片保存到磁盘失败 |
| 设置食品数量失败 | "设置食品数量失败" | 服务器无法更新全局配置 | 更新food\_number配置失败 |
| RFID连接失败 | "RFID连接失败" | RFID硬件连接失败 | RFID设备连接失败 |
| RFID读取失败 | "RFID读取失败" | RFID硬件操作失败 | RFID读取过程中出现异常 |
| 信息处理失败 | "信息处理失败" | OCR识别、分类检测、表格日期检测等处理失败 | 信息处理模块执行失败 |
| 上传成功确认处理失败 | "上传成功确认处理失败" | 数据库操作失败 | 标记数据已同步到手机端失败 |
| 入库结束处理失败 | "入库结束处理失败" | 清理操作失败 | 清理临时数据失败 |
| 中止入库失败 | "中止入库失败" | 中止入库操作失败 | 中止入库过程中出现异常 |

手机端API

图示：

【小程序界面】

“按钮”

「文字内容」

手机端小程序界面操作触发

手机端小程序UI界面响应

手机端后台操作

手机端后台操作逻辑

\*回合API

* + **没有注明怎样视为本步骤成功/失败的，默认为收到成功/失败返回消息就算做成功/失败**

**连接与断开连接**

1、连接建立

手机端连接成功后，树莓派会自动发送连接确认消息，手机端无需发送连接请求

连接确认消息

{

"type": "connection",

"action": "success",

"data": {

"status": "connected"

},

"timestamp": "2024-01-01T12:00:00.123456"

}

2、断开连接

手机端直接关闭WebSocket连接即可，树莓派会自动清理连接

注：AI推荐的手机端WebSocket连接与断开代码

const ws = new WebSocket('ws://192.168.1.100:5001'); //此处支持用户输入，不要硬编码

ws.onopen = function(event) {

console.log('WebSocket连接已建立');

};

ws.onmessage = function(event) {

const message = JSON.parse(event.data);

console.log('收到消息:', message);

};

ws.onclose = function(event) {

console.log('WebSocket连接已关闭');

};

ws.onerror = function(error) {

console.log('WebSocket错误:', error);

};

**入库流程**

【数据准备界面】

界面功能：拍摄和输入食品数量

1、界面包含左上角退出入库流程的按钮“🡨”，点击可以中止入库过程

\*中止入库

请求

{

"type": "inbound\_cancel" // 消息类型，表示中止入库请求

}

1. 成功

成功返回消息

成功标志: type="inbound\_cancel" + action="success" + status="cancelled"

{

"type": "inbound\_cancel", // 消息类型，表示中止入库

"action": "success", // 动作类型，表示成功

"data": {

"status": "cancelled", // 状态类型，表示已取消

"message": "入库流程已中止" // 成功消息

},

"timestamp": "2025-01-08T12:35:30.456789" // 时间戳（手机端可忽略）

}

小程序返回入库界面首页

2. 失败——程序报错，中止程序

失败返回消息

失败标志: type="error" + action="error" + error\_message包含"中止入库失败"

{

"type": "error", // 消息类型

"action": "error", // 动作类型，表示错误

"data": {

"status": "error", // 状态类型，表示错误

"error\_message": "中止入库失败" // 错误消息

},

"timestamp": "2025-01-08T12:35:30.456789" // 时间戳（手机端可忽略）

}

2、拍摄完成、输入食品数量后，点击“下一步”

（1）手机端以二进制形式传送图片组至images\_tempt

注：AI推荐上传顺序：按添加顺序依次上传，每张图片上传成功后继续下一张

\*图片组传送

手机端直接发送二进制数据，无需JSON请求；每张图片单独发送一次WebSocket消息

byte[] imageData = getImageBytes(); // 获取图片的二进制数据

websocketClient.send(imageData); // 直接发送二进制数据

单张图片上传成功返回消息

成功标志: type="inbound\_ready" + action="success" + status="ready"

{

"type": "inbound\_ready", // 消息类型，表示准备完成

"action": "success", // 动作类型，表示成功

"data": {

"status": "ready", // 状态类型，表示准备完成

"message": "图片上传成功", // 成功消息

"filename": "food\_20250108\_123530\_456.jpg" // 服务器生成的文件名

},

"timestamp": "2025-01-08T12:35:30.456789" // 时间戳（手机端可忽略）

}

失败返回消息

失败标志: type="error" + action="error" + error\_message包含"保存图片失败"

{

"type": "error",

"action": "error",

"data": {

"status": "error",

"error\_message": "保存图片失败"

},

"timestamp": "2025-01-08T12:35:30.456789"

}

（2）手机端传送物品数量food\_number

\*food\_number传送

请求

{

"type": "inbound\_food\_number", // 消息类型，表示食品数量设置

"data": {

"food\_number": 3 // 要设置的食品数量（整数）

}

}

成功返回消息

{

"type": "inbound\_ready", // 消息类型，表示准备完成

"action": "success", // 动作类型，表示成功

"data": {

"status": "ready", // 状态类型，表示准备完成

"message": "食品数量设置成功", // 成功消息

"food\_number": 3 // 确认设置的食品数量

},

"timestamp": "2025-01-08T12:35:30.456789" // 时间戳（手机端可忽略）

}

失败返回消息

{

"type": "error", // 消息类型

"action": "error", // 动作类型，表示错误

"data": {

"status": "error", // 状态类型，表示错误

"error\_message": "设置食品数量失败" // 错误消息

},

"timestamp": "2025-01-08T12:35:30.456789" // 时间戳（手机端可忽略）

}

1. 如果（1）（2）中收到任何一个失败消息，认为本步骤失败：

注：AI推荐错误处理: 上传失败时可选择重试或跳过

手机端收集到底是谁失败了，弹出提示消息「xx传送失败」，留在**数据准备界面**

2. 如果（1）（2）都成功，认为本步骤成功：

小程序进入**RFID读取界面**

【RFID读取界面】

提示进度，告知用户应该读取标签以及需要读取的标签数，界面显示「请扫描xx个RFID标签」

手机端进入RFID读取界面后，发送进入RFID读取阶段请求

\*启动RFID读取

请求

{

"type": "inbound\_rfid\_reading" // 消息类型，表示启动RFID读取

}

1. 成功（读取到目标数量）

成功返回消息

成功标志: type="inbound\_rfid\_result" + action="success" + status="rfid\_reading\_success"

{

"type": "inbound\_rfid\_result", // 消息类型，表示RFID读取结果

"action": "success", // 动作类型，表示成功

"data": {

"status": "rfid\_reading\_success", // 状态类型，表示RFID读取成功

"tag\_count": 3, // 读取到的标签数量

"rfid\_list": ["tag001", "tag002", "tag003"] // 读取到的标签ID列表

},

"timestamp": "2025-01-08T12:35:30.456789" // 时间戳（手机端可忽略）

}

小程序进入**信息处理界面**。

2. 超时（未读取到目标数量）

超时返回消息

超时标志: type="inbound\_rfid\_result" + action="timeout" + status="rfid\_reading\_timeout"

{

"type": "inbound\_rfid\_result", // 消息类型，表示RFID读取结果

"action": "timeout", // 动作类型，表示超时

"data": {

"status": "rfid\_reading\_timeout", // 状态类型，表示RFID读取超时

"tag\_count": 2, // 已识别的标签数量

"rfid\_list": ["tag001", "tag002"] // 已识别的标签ID列表

},

"timestamp": "2025-01-08T12:35:30.456789" // 时间戳（手机端可忽略）

}

小程序弹出提示消息「读取超时，仅识别get\_number个标签」，然后进入**信息处理界面**。

3.失败

失败返回消息

失败标志: type="error" + action="error" + error\_message包含"RFID连接失败"或"RFID读取失败"

{

"type": "error", // 消息类型

"action": "error", // 动作类型，表示错误

"data": {

"status": "error", // 状态类型，表示错误

"error\_message": "RFID连接失败" // 错误消息

},

"timestamp": "2025-01-08T12:35:30.456789" // 时间戳（手机端可忽略）

}

【信息处理界面】

提示进度，告知用户数据正在处理，界面文字显示“数据处理中，请稍候”

1、手机进入数据处理页面后，发送信息处理请求

\*信息处理

请求

{

"type": "inbound\_info\_process" // 消息类型，表示启动信息处理

}

（1）若数据和图片的JSON返回消息均成功

数据处理完成返回信息

数据完成标志: type="inbound\_info\_process" + action="success" + status="data\_ready"

{

"type": "inbound\_info\_process", // 消息类型，表示信息处理

"action": "success", // 动作类型，表示成功

"data": {

"status": "data\_ready", // 状态类型，表示数据准备完成

"db\_data\_list": { // 识别到的所有数据

"food\_number": 3, // 食品数量

"rfid\_list": ["tag001", "tag002", "tag003"], // RFID标签列表

"name": "苹果", // 食品名称

"category": "水果", // 食品类别

"production\_date": "2025-01-01", // 生产日期

"expiry\_date": "2025-01-15" // 保质期

}

},

"timestamp": "2025-01-08T12:35:30.456789" // 时间戳（手机端可忽略）

}

图片处理完成，准备发送返回信息

图片完成标志: type="inbound\_info\_process" + action="images" + status="images\_ready"

{

"type": "inbound\_info\_process", // 消息类型，表示信息处理

"action": "images", // 动作类型，表示图片数据

"data": {

"status": "images\_ready", // 状态类型，表示图片处理完成

"image\_count": 2, // 处理结果图片数量

"message": "开始发送2张处理结果图片" // 提示消息

},

"timestamp": "2025-01-08T12:35:30.456789" // 时间戳（手机端可忽略）

}

// 该信息后，树莓派将依次发送处理结果图片的二进制数据，每张图片单独发送一次WebSocket消息，图片数量等于上面的image\_count

手机端本地保存返回信息中的食品信息和传回图片

手机端检查保存好刚才上传的两个数据后，发送上传成功确认请求

\*上传成功确认

请求

{

"type": "inbound\_upload\_success" // 消息类型，表示上传成功确认

}

1.成功

成功返回消息

成功标志: type="inbound\_upload\_success" + action="success" + status="upload\_success"

{

"type": "inbound\_upload\_success", // 消息类型，表示上传成功确认

"action": "success", // 动作类型，表示成功

"data": {

"status": "upload\_success" // 状态类型，表示上传成功

},

"timestamp": "2025-01-08T12:35:30.456789" // 时间戳（手机端可忽略）

}

手机小程序进入**识别完成提示界面**。

2. 失败

失败返回消息

失败标志: type="error" + action="error" + error\_message包含"上传成功确认处理失败"

{

"type": "error", // 消息类型

"action": "error", // 动作类型，表示错误

"data": {

"status": "error", // 状态类型，表示错误

"error\_message": "上传成功确认处理失败" // 错误消息

},

"timestamp": "2025-01-08T12:35:30.456789" // 时间戳（手机端可忽略）

}

手机小程序进入**识别失败提示界面**。

（2）若数据和图片中任何一步返回失败消息

失败返回消息

失败标志: type="error" + action="error" + error\_message包含"信息处理失败"

{

"type": "error", // 消息类型

"action": "error", // 动作类型，表示错误

"data": {

"status": "error", // 状态类型，表示错误

"error\_message": "信息处理失败" // 错误消息

},

"timestamp": "2025-01-08T12:35:30.456789" // 时间戳（手机端可忽略）

}

手机小程序进入**识别失败提示界面**。

【识别完成提示界面】

界面文字提示「数据保存成功，3秒后将自动跳转返回首页」

【识别失败提示界面】

界面文字提示「数据保存失败，请重新操作，3秒后将自动跳转返回首页」

手机端发送入库结束请求

\*入库结束

请求

{

"type": "inbound\_end" // 消息类型，表示入库结束请求

}

1. 返回成功消息

成功返回消息

成功标志: type="inbound\_end" + action="success" + status="end\_success"

{

"type": "inbound\_end", // 消息类型，表示入库结束

"action": "success", // 动作类型，表示成功

"data": {

"status": "end\_success" // 状态类型，表示入库结束成功

},

"timestamp": "2025-01-08T12:35:30.456789" // 时间戳（手机端可忽略）

}

3秒后跳转回小程序首页

2. 返回失败消息——程序报错，中止程序

失败返回消息

失败标志: type="error" + action="error" + error\_message包含"入库结束处理失败"

{

"type": "error", // 消息类型

"action": "error", // 动作类型，表示错误

"data": {

"status": "error", // 状态类型，表示错误

"error\_message": "入库结束处理失败" // 错误消息

},

"timestamp": "2025-01-08T12:35:30.456789" // 时间戳（手机端可忽略）

}

**出库流程**

树莓派出库一个食品后会立即对所有连接客户端发出出库广播

\*出库广播：树莓派 → 小程序

广播消息

{

"type": "outbound\_notification",

"action": "success",

"data": {

"status": "delete\_food\_item",

"food\_id": 123,

"timestamp": "2024-01-01T12:00:00",

"message": "检测到食品出库，请删除ID为123的食品条目"

},

"timestamp": "2024-01-01T12:00:00.123456"

}

小程序收到后用data中的数据做处理：

根据 action: "success" 判断操作成功

根据 data.status: "delete\_food\_item" 判断需要删除食品条目

根据 food\_id: 123 执行删除操作：如果剩余食品数量>1就减1，如果剩余食品数量=1就直接在显示界面删除这一条

在废纸篓界面生成一条信息，显示出库食品名称、出库时间