

# djiVue

---

This template should help get you started developing with Vue 3 in Vite.

## Recommended IDE Setup

[VSCode](#) + [Volar](#) (and disable Vetur) + [TypeScript Vue Plugin \(Volar\)](#).

## Customize configuration

See [Vite Configuration Reference](#).

## Project Setup

```
npm install
```

## Compile and Hot-Reload for Development

```
npm run dev
```

## Compile and Minify for Production

```
npm run build
```

## LiveStream Setup

```
docker run -p 8000:8000 -it mpromonet/webrtc-streamer -n rtsp_stream -u  
rtsp://aaa:aaa@{rc_ip}:8554/streaming/live/1
```

this rtsp ip should be written in /src/views/LiveStream.vue replaced by ?

```
const RTSP_URL = '? '
```

## WEBRTC\_STREAMER\_IP

http://localhost:8000 which was supplied by docker container.

## TELEMETRY

it is supported by python script, so we get it from <http://localhost:8765>

# What you SHOULD DO

---

## 1. beautify all the .vue files in `/src/views` without change the main logic communicating with the server.

就是说，这个目录下的 `vue` 文件都可以被修改，但是你主要修改的部分应当在布局也就是 `template` 和 `css` 样式部分，而不是其中的关于与后端交互的逻辑，

为了便于后期修改。首先介绍第一个界面，**控制参数**。这个界面的主要功能是调试无人机控制相关的各项参数，通过按钮点击事件发送信号。

第二个界面是**直播界面**，通过 `webrtc-streamer` 将无人机控制器端的 `rtsp` 直播流转化为 `mp4` 流，以便网页播放。

第三个界面是**信息界面**，通过与无人机遥控端构建 `websocket` 实现实时的信息接收。

第四个界面则是引申出来的**控制模式**，即用键盘来控制无人机移动，`wsad` 用来控制移动，上下左右分别用来控制上升下降和转动。

其中还有 `utils` 中的 `request.js` 提供的方法可以利用，写出新的 `views` 功能，

## 页面一：控制参数界面（Control Parameters View）

### 页面功能说明

该页面用于 **调试无人机控制相关参数**，  
通过点击不同的控制按钮，向无人机控制端发送对应的控制信号。

---

### 页面改造 TODOList

#### 布局与结构

- ☐ 页面划分为清晰的控制区域（如参数分组）
- ☐ 不同类型的控制参数分区展示
- ☐ 页面逻辑一目了然，避免按钮堆叠

#### 交互与按钮设计

- ☐ 所有按钮具备明确语义（如：启动 / 停止 / 重置）
- ☐ 危险或关键操作按钮样式明显
- ☐ 点击反馈清晰（颜色 / 状态变化）

#### 逻辑约束

- ☐ 按钮点击事件逻辑保持不变
  - ☐ 发送给后端的控制信号完全不修改
  - ☐ 仅优化视觉表现与布局结构
-

## 页面二：直播界面（Live Stream View）

### 页面功能说明

- 使用 `webrtc-streamer`
- 将无人机控制器端的 **RTSP 视频流**
- 转换为浏览器可播放的 **MP4 / FLV / WebRTC 流**
- 实现网页端实时视频预览

☒ 页面改造 TodoList

### 视频布局

- ☐ 视频区域作为页面视觉核心
- ☐ 保证视频显示比例正确（防拉伸）
- ☐ 页面其余元素不干扰视频观看

### UI 美化

- ☐ 视频容器边框弱化、工业风处理
- ☐ 无信号 / 加载中 状态有占位提示
- ☐ 页面整体暗色，突出视频内容

### 逻辑约束

- ☐ 视频流地址与播放逻辑保持不变
- ☐ 不修改 `webrtc-streamer` 的调用方式
- ☐ 仅调整 `template` 与 `style`

## 页面三：信息界面（Telemetry / Info View）

### 页面功能说明

- 通过 **WebSocket**
- 与无人机遥控端建立实时通信
- 接收并展示无人机状态信息

☒ 页面改造 TodoList

### 数据展示设计

- ☐ 信息以卡片 / 仪表盘形式展示
- ☐ 状态信息分类清晰（位置 / 姿态 / 电量等）
- ☐ 数据刷新时视觉稳定，不跳动

### 交互与可读性

- ☐ 字体大小适合长时间监控
- ☐ 单位清晰（m / ° / % 等）
- ☐ 异常状态（如断连）明显提示

逻辑约束

- ☐ **WebSocket 连接逻辑保持不变**
- ☐ **数据解析与字段名称不修改**
- ☐ 只优化 UI 展示方式

---

## 页面四：控制模式界面（Keyboard Control View）

页面功能说明

- 使用键盘控制无人机移动
- 控制规则如下：
  - **W / A / S / D**：前 / 左 / 后 / 右 移动
  - **↑ / ↓**：上升 / 下降
  - **← / →**：旋转方向

---

☒ 页面改造 TodoList

视觉提示

- ☐ 页面清晰展示键位说明
- ☐ 当前按下的按键有实时高亮反馈
- ☐ 控制状态清楚，防止误操作

页面布局

- ☐ 控制说明区域与状态显示分离
- ☐ 布局类似“控制台 / 操作面板”

逻辑约束

- ☐ **键盘监听逻辑保持不变**
- ☐ **发送给后端的控制指令不修改**
- ☐ 只增强可视化与交互体验

关于无人机通信的各接口调用如下（仅作参考使用，我已实现，你可以在这个基础上优化一下代码风格）。

**Telemetry Fields**（我已经将其**转化**为**WEBSOCKET**，具体的**转换**在src下的WebSocketServer.py文件中写完了。如果需要利用**连续信息流**，**请查看info.vue**）：

Field	Description
speed	Aircraft velocity (x, y, z)

Field	Description
heading	Compass heading in degrees
attitude	Pitch, roll, yaw values
location	GPS coordinates and altitude
gimbalAttitude	Gimbal orientation
batteryLevel	Battery percentage
satelliteCount	GPS satellite count
homeLocation	Home point coordinates
distanceToHome	Distance to home in meters
waypointReached	Waypoint status flag
isRecording	Camera recording status
flightMode	Current flight mode (GPS, MANUAL, GO_HOME, etc.)
remainingFlightTime	Estimated flight time remaining
batteryNeededToGoHome	Battery % needed for RTH
batteryNeededToLand	Battery % needed to land
timeNeededToGoHome	Time to return home (seconds)
maxRadiusCanFlyAndGoHome	Max flyable radius (meters)

### Control Endpoints (HTTP POST - Port 8080)

Endpoint	Description	Parameters
/send/takeoff	Initiate takeoff	None
/send/land	Initiate landing	None
/send/RTH	Return to home	None
/send/gotoWP	Navigate to waypoint	lat,lon,alt
/send/gotoWPwithPID	Navigate with PID control	lat,lon,alt,yaw
/send/gotoYaw	Rotate to heading	yaw_angle
/send/gotoAltitude	Change altitude	altitude
/send/navigateTrajectory	Follow trajectory (Virtual Stick)	lat,lon,alt;...;lat,lon,alt,yaw

Endpoint	Description	Parameters
<code>/send/navigateTrajectoryDJIINative</code>	DJI native waypoint mission	<code>lat,lon,alt;lat,lon,alt;...</code>
<code>/send/abort/DJIMission</code>	Stop DJI native mission	None
<code>/send/abortMission</code>	Stop and disable Virtual Stick	None
<code>/send/enableVirtualStick</code>	Enable Virtual Stick mode	None
<code>/send/stick</code>	Virtual stick input	<code>leftX,leftY,rightX,rightY</code>
<code>/send/camera/zoom</code>	Camera zoom control	<code>zoom_ratio</code>
<code>/send/camera/startRecording</code>	Start video recording	None
<code>/send/camera/stopRecording</code>	Stop video recording	None
<code>/send/gimbal/pitch</code>	Gimbal pitch control	<code>roll,pitch,yaw</code>
<code>/send/gimbal/yaw</code>	Gimbal yaw control	<code>roll,pitch,yaw</code>

### Status Endpoints (HTTP GET - Port 8080)

Endpoint	Description
<code>/status/waypointReached</code>	Check if waypoint reached
<code>/status/intermediaryWaypointReached</code>	Check intermediary waypoint
<code>/status/yawReached</code>	Check if target yaw reached
<code>/status/altitudeReached</code>	Check if target altitude reached
<code>/status/camera/isRecording</code>	Check recording status

### Legacy Telemetry Endpoints (HTTP GET - Port 8080)

These endpoints are available for backward compatibility. For continuous telemetry, use the TCP socket on port 8081.

Endpoint	Description
<code>/</code>	Connection test
<code>/aircraft/allStates</code>	Complete telemetry package (JSON)

Endpoint	Description
/aircraft/speed	Aircraft velocity
/aircraft/heading	Compass heading
/aircraft/attitude	Pitch, roll, yaw
/aircraft/location	GPS coordinates and altitude
/aircraft/gimbalAttitude	Gimbal orientation
/home/location	Home point coordinates

2. 中英文过渡切换按钮实现。要实现可以将页面全部切换成英文和中文。