

串口控制协议

目录

- [1. 指令格式](#)
 - [1.1 串口参数](#)
- [2. JSON 结构](#)
 - [2.1 字段说明](#)
 - [2.2 发送提示](#)
 - [2.3 查询命令 \(AT+QUERY\)](#)
- [3. 支持服务与字段 \(详解\)](#)
 - [3.1 switch \(开关\)](#)
 - [3.2 brightness \(亮度\)](#)
 - [3.3 cct \(色温\)](#)
 - [3.4 lightMode \(场景模式\)](#)
 - [3.5 progressSwitch \(渐变时长\)](#)
 - [3.6 colourMode \(色温模式\)](#)
- [4. 约束](#)
- [5. 错误码](#)
 - [5.1 NotControllable \(112\) 用法建议](#)
- [6. 时序图示 \(Mermaid\)](#)
- [7. 示例](#)

1. 指令格式

- 下发: `AT+CTRL={JSON}\r\n`
- 查询: `AT+QUERY={JSON}\r\n`
- 回执: `OK,<id>\r\n` 或 `ERROR,<code>[,<message>]\r\n`
- 异步结果: `AT+RESP={JSON}\r\n`
- 规则:
 - 单行 JSON (内部换行请写成 `\n`)
 - 行结束统一为 CRLF `\r\n` (Windows 兼容)
 - 接收端应兼容仅 LF `\n`

1.1 串口参数

- 波特率: `9600`
- 数据位: `8`
- 校验位: `None`
- 停止位: `1`

- 流控: `None`
- 编码: `UTF-8` (无 BOM)
- 建议: 行缓冲读取, 按 CRLF `\r\n` 作为帧结束。

2. JSON 结构

- 单服务控制 (必须携带 id) :
 - 下发: `{"id": <u32>, "sid": "<serviceId>", "data": {<fields>}}`
 - 返回: `{"id": <u32>, "sid": "<serviceId>", "data": {<fields>}, "error": <u32>, "message": "<可选>"}` (`error=0` 表示成功)

2.1 字段说明

- `id` (number, u32>0, 必填)
 - 会话标识, 用于将异步结果与请求一一对应、便于定位问题。
 - 由发起方分配与维护, 建议使用一个全局自增计数 (溢出后回绕到 1)。
 - 受控方不做去重与顺序检查, 仅在 RESP 中原样回显。
- `sid` (string, 必填)
 - 服务功能 ID。合法值: `switch`、`brightness`、`cct`、`lightMode`、`progressSwitch`、`colourMode`。
 - 区分大小写; 不在集合内的值视为不支持服务。
- `data` (object, 必填)
 - 具体服务的数据负载。各服务的字段与范围见“支持服务与字段 (详解)”。
 - 缺字段或类型不符为业务错误, RESP `error=105` (`message` 可携带原因)。
- `error` (number, u32, RESP 必带)
 - 业务执行结果码: `0` 成功; 非 `0` 失败。常见值见“错误码”。
- `message` (string, 可选, RESP 可带)
 - 人类可读的错误或提示信息, 用于调试。

2.2 发送提示

- 多次控制: 请按需多次发送 `AT+CTRL=...`, 每次使用不同的 `id`。
- ACK 超时与重发建议:
 - 发送后等待 `OK,<id>`, 超时建议 `200-500 ms`。
 - 未收到 ACK 则按原 `id` 与完全相同的 JSON 重发, 重发 `2-3` 次, 间隔 `200-300 ms`。
 - 注意: 设备不做去重, 重复下发可能重复执行; 建议尽量使用幂等设置 (如重复设置相同亮度)。
- 结果等待与超时:
 - 收到 `OK,<id>` 后等待 `AT+RESP`, 建议超时阈值 `1-2 s` (视业务而定)。
 - 超时未收到可按失败处理并在上层重试 (使用新的 `id`)。
- 换行约定: 所有发送与回执均以 CRLF `\r\n` 结束; 示例为单行展示, 实际发送需包含 CRLF。

2.3 查询命令 (AT+QUERY)

- 作用：查询指定 `sid` 的当前状态（只读，不改变设备状态）。
- 下发：`AT+QUERY={"id": <u32>, "sid": "<serviceId>"}` \r\n
- 回执：`OK,<id>` \r\n
- 异步结果：`AT+RESP={"id": <u32>, "sid": "<serviceId>", "data": {<current-fields>}, "error": <u32>, "message": "<可选>"}` \r\n
- 说明：
 - `data` 与“支持服务与字段（详解）”中各 `sid` 的状态格式一致（等同于 Get 接口的输出）。
 - 若 `sid` 不支持，返回 `ERROR,104,UnsupportedSid`（协议级错误）。

3. 支持服务与字段（详解）

3.1 switch（开关）

- 功能：控制灯具开/关。
- 下发：`{"id":<u32>,"sid":"switch","data":{"on":0|1}}`
- 返回：`{"id":<u32>,"sid":"switch","data":{"on":0|1},"error":<u32>,"message":"<可选>"}`
- 参数：`on` 为 int，取值 0 关、1 开。
- 行为：当处于“离家模式”（mode=7）时，收到 `on=1` 会先退出离家模式再开灯。
- 示例：
 - 开灯
 - 【发起方发送】`AT+CTRL={"id":101,"sid":"switch","data":{"on":1}}`
 - 【受控方返回】`OK,101` 与 `AT+RESP={"id":101,"sid":"switch","data":{"on":1},"error":0}`
 - 关灯
 - 【发起方发送】`AT+CTRL={"id":102,"sid":"switch","data":{"on":0}}`
 - 【受控方返回】`OK,102` 与 `AT+RESP={"id":102,"sid":"switch","data":{"on":0},"error":0}`

3.2 brightness（亮度）

- 功能：设置亮度百分比。
- 下发：`{"id":<u32>,"sid":"brightness","data":{"brightness":0..100}}`
- 返回：`{"id":<u32>,"sid":"brightness","data":{"brightness":<0..100>},"error":<u32>,"message":"<可选>"}`
- 参数：`brightness` 为 int，范围 0..100；超出范围按边界钳制。
- 示例：
 - 设为 60%：`AT+CTRL={"id":201,"sid":"brightness","data":{"brightness":60}}`
 - 最小值：`AT+CTRL={"id":202,"sid":"brightness","data":{"brightness":0}}`
 - 最大值：`AT+CTRL={"id":203,"sid":"brightness","data":{"brightness":100}}`

3.3 cct (色温)

- 功能: 设置色温 (单位 K) 。
- 下发: `{"id":<u32>,"sid":"cct","data":{"colorTemperature":2700..6000}}`
- 返回: `{"id":<u32>,"sid":"cct","data":{"colorTemperature":<2700..6000>},"error":<u32>,"message":"<可选>"}`
- 参数: `colorTemperature` 为 int, 范围 2700..6000 ; 超出范围按边界钳制。
- 示例:
 - 暖色 3000K: `AT+CTRL={"id":301,"sid":"cct","data":{"colorTemperature":3000}}`
 - 中性 4000K: `AT+CTRL={"id":302,"sid":"cct","data":{"colorTemperature":4000}}`
 - 冷色 6000K: `AT+CTRL={"id":303,"sid":"cct","data":{"colorTemperature":6000}}`

3.4 lightMode (场景模式)

- 功能: 按预设场景设置亮度与色温, 或进入离家模式。
- 下发: `{"id":<u32>,"sid":"lightMode","data":{"mode":0..7}}`
- 返回: `{"id":<u32>,"sid":"lightMode","data":{"mode":<0..7>},"error":<u32>,"message":"<可选>"}`
- 参数: `mode` 为 int:
 - 0 customer (自定义模式)
 - 1 relax (休闲模式) (约 50%, 4000K)
 - 2 movie (观影模式) (约 10%, 3000K)
 - 3 dining (用餐模式) (约 100%, 4000K)
 - 4 home (回家模式) (约 80%, 3500K)
 - 5 winter (冬天模式) (约 100%, 2700K)
 - 6 summer (夏天模式) (约 100%, 6000K)
 - 7 leave (离家模式) (直接关灯)
- 示例:
 - 观影模式: `AT+CTRL={"id":401,"sid":"lightMode","data":{"mode":2}}`
 - 离家模式: `AT+CTRL={"id":402,"sid":"lightMode","data":{"mode":7}}`
 - 回家模式: `AT+CTRL={"id":403,"sid":"lightMode","data":{"mode":4}}`

3.5 progressSwitch (渐变时长)

- 功能: 设置亮度/色温变化的平滑过渡时间 (秒) 。
- 下发: `{"id":<u32>,"sid":"progressSwitch","data":{"fadeTime":0..30}}`
- 返回: `{"id":<u32>,"sid":"progressSwitch","data":{"fadeTime":<0..30>},"error":<u32>,"message":"<可选>"}`
- 参数: `fadeTime` 为 int, 范围 0..30 秒; 0 表示尽快变化。
- 示例:
 - 设为 5 秒: `AT+CTRL={"id":501,"sid":"progressSwitch","data":{"fadeTime":5}}`

- 立即变化: `AT+CTRL={"id":502,"sid":"progressSwitch","data":{"fadeTime":0}}`

3.6 colourMode (色温模式)

- 功能: 设置色温工作模式。
- 下发: `{"id":<u32>,"sid":"colourMode","data":{"mode":0|1}}`
- 返回: `{"id":<u32>,"sid":"colourMode","data":{"mode":0|1},"error":<u32>,"message":"<可选>"}`
- 参数: `mode` 为 int: 0 单色温、1 双色温。
- 行为: 单色温 (0) 下, 设备固定到预设色温 (6000K); 双色温 (1) 下, 使用/保持当前色温。
- 示例:
 - 单色温: `AT+CTRL={"id":601,"sid":"colourMode","data":{"mode":0}}`
 - 双色温: `AT+CTRL={"id":602,"sid":"colourMode","data":{"mode":1}}`

4. 约束

- `{JSON}` 最大长度: ≤ 1024 字节 (不含前缀与行尾)。
- 单行承载, 不允许真实换行; 内部换行使用 `\n` 字符串。
- 行结束采用 CRLF `\r\n`; 接收端应兼容仅 LF `\n`。
- `id` 必须存在且为 `u32>0`; 缺失/类型不符即返回 `ERROR,105,TypeError`。

5. 错误码

- `100` BadPrefix (未以 `AT+CTRL=` 开头)
- `101` MalformedJSON (JSON 语法错误)
- `102` EmptyJSON (空 JSON)
- `103` PayloadTooLong (负载超长)
- `104` UnsupportedSid (不支持的 `sid`)
- `105` TypeError (字段缺失或类型不符)
- `106` Busy (设备忙)
- `107` ApplyTimeout (内部执行超时)
- `112` NotControllable (非受控状态: 设备处于不可被控制的模式, 例如升级/维护/工厂测试/安全锁定等)
- `111` RateLimited (限频)

5.1 NotControllable (112) 用法建议

- 触发场景 (示例):
 - 升级进行中 (OTA/独立升级/固件校验)
 - 维护/自检/烧录/工厂测试模式
 - 安全锁定/童锁/设备被上级网关临时锁定
 - 热保护/过温降额/低电量保护等安全策略生效
 - 正在配网/关键迁移流程, 不允许外部控制

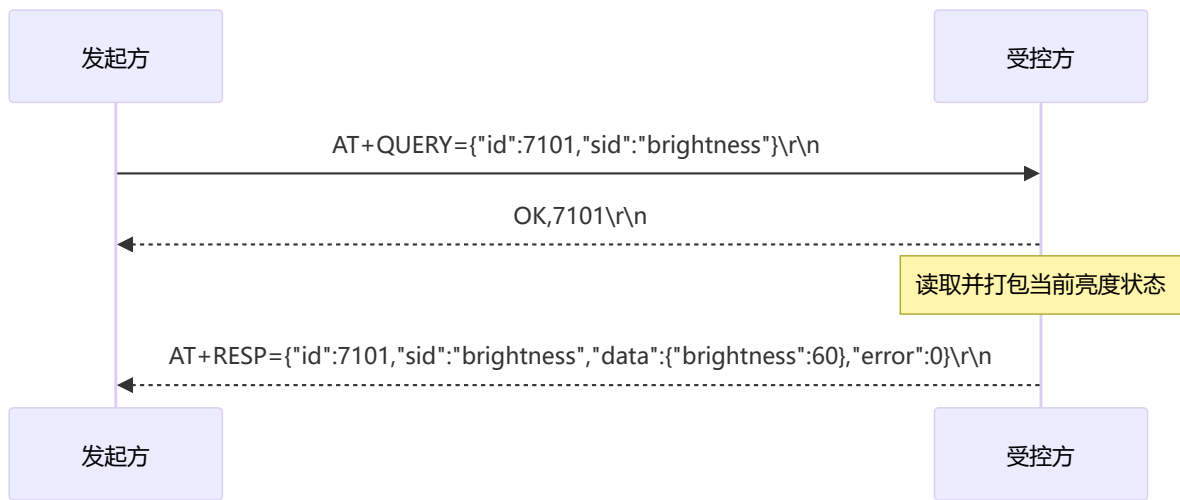
- 返回规范：
 - 收到控制后先回 ACK: `OK,<id>`
 - 业务结束后返回: `AT+RESP={"id":<id>,"sid":"<sid>","data":{...},"error":112,"message":"<原因>"}`
- `message` 建议取值（便于前端统一展示与日志聚合）：
 - `upgrading`（升级中）
 - `maintenance`（维护/自检）
 - `factory_test`（工厂测试）
 - `locked`（安全/家长锁定）
 - `thermal_protect`（热保护）
 - `low_power`（低电量保护）
 - `net_config`（正在配网）
- 查询建议：
 - `AT+QUERY` 通常仍应返回当前状态并 `error=0`；若状态不可读，再返回 `error=112` 与合适的 `message`。

6. 时序图示（Mermaid）

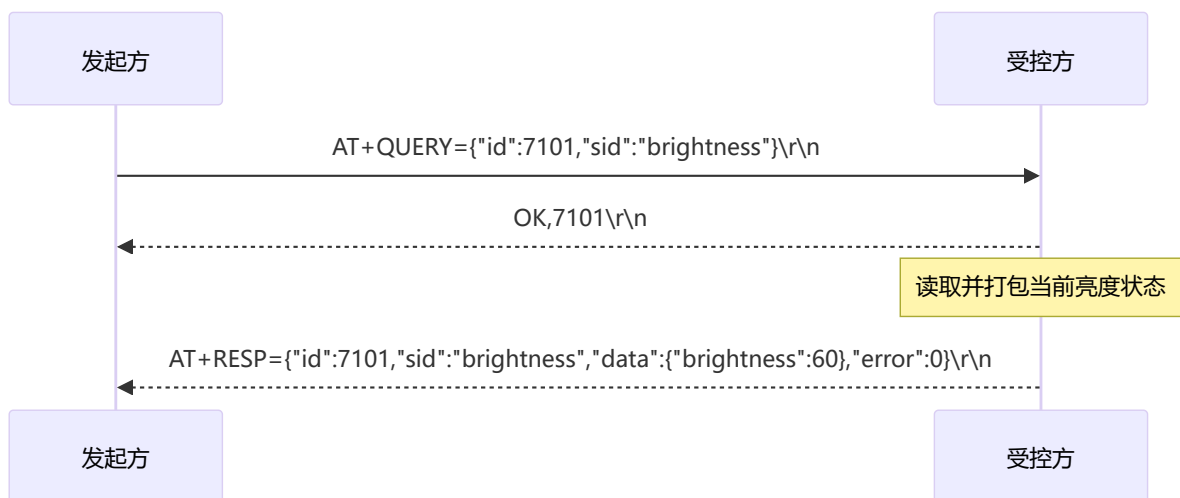
1. 多次控制（均成功，异步返回 `error=0`）



2. 协议级错误（JSON 语法错误，直接 `ERROR`，不产生 `RESP`）



3. 业务错误（收到并受理，但参数类型错误）



4. 查询状态（OK + RESP）



7. 示例

1. 开关：打开 【发起方发送】

```
AT+CTRL={\"id\":7001,\"sid\":\"switch\",\"data\":{\"on\":1}}
```

【受控方返回】

```
OK,7001
```

【受控方返回】

```
AT+RESP={"id":7001,"sid":"switch","data":{"on":1},"error":0}
```

2. 查询当前亮度

【发起方发送】

```
AT+QUERY={"id":7101,"sid":"brightness"}
```

【受控方返回】

```
OK,7101
```

【受控方返回】

```
AT+RESP={"id":7101,"sid":"brightness","data":{"brightness":60},"error":0}
```

3. 场景模式：movie

【发起方发送】

```
AT+CTRL={"id":7201,"sid":"lightMode","data":{"mode":2}}
```

【受控方返回】

```
OK,7201
```

【受控方返回】

```
AT+RESP={"id":7201,"sid":"lightMode","data":{"mode":2},"error":0}
```

4. 渐变时长：5 秒

【发起方发送】

```
AT+CTRL={"id":7301,"sid":"progressSwitch","data":{"fadeTime":5}}
```

【受控方返回】

```
OK,7301
```

【受控方返回】

```
AT+RESP={"id":7301,"sid":"progressSwitch","data":{"fadeTime":5},"error":0}
```

5. 色温模式：单色温

【发起方发送】

```
AT+CTRL={"id":7401,"sid":"colourMode","data":{"mode":0}}
```

【受控方返回】

```
OK,7401
```

【受控方返回】

```
AT+RESP={"id":7401,"sid":"colourMode","data":{"mode":0},"error":0}
```

6. 业务失败（类型错误）

【发起方发送】

```
AT+CTRL={"id":7501,"sid":"brightness","data":{"brightness":"sixty"}}
```

【受控方返回】

```
OK,7501
```

【受控方返回】

```
AT+RESP={"id":7501,"sid":"brightness","data":  
{"brightness":50},"error":105,"message":"TypeError"}
```

7. 非法JSON（协议级错误）

【发起方发送】

```
AT+CTRL={"sid":"switch","data":{"on":1}}
```

【受控方返回】

```
ERROR,101,MalformedJSON
```

8. 不支持的服务（协议级错误）

【发起方发送】

```
AT+CTRL={"id":7601,"sid":"foo","data":{"bar":1}}
```

【受控方返回】

```
ERROR,104,UnsupportedSid
```