直播字幕实时回调接入说明

当您在管理后台配置后，系统会在产生/汇总到一批字幕数据时，向该地址发起回调请求，将字幕内容推送到您的服务端。 后台配置如图：

callbackUrl 如何填写

- **填写内容**：一个您方可公网访问的完整 URL，例如：`https://api.example.com/polyv/subtitles/callback`
- **请求方式**：必须支持 `POST`
- **数据格式**：必须支持解析 `multipart/form-data`（表单方式，不是 JSON Body）
- **响应要求**：建议在 10 秒内返回；返回任意 **2xx** 即视为成功（响应体内容不做强校验）

回调请求说明

请求方式

`POST`

Content-Type

`multipart/form-data`

超时与重试

- **超时**：10 秒
- **重试**：失败会自动重试，最多 3 次
- **幂等建议**：由于可能重试，请您按 `roomId + sessionId + index`（或其它业务唯一键）做去重/幂等处理

请求参数

参数名	必选	类型	说明
roomId	是	string	频道号（房间号）
sessionId	是	string	本场直播/会话 ID
subtitles	是	string	字幕数组的 JSON 字符串（见下方字段说明）
timestamp	是	number	请求13位时间戳（毫秒，`Date.now()`）
sign	是	string	签名，用于鉴权与防篡改（见下方验签说明）

`subtitles` 字段说明

`subtitles` 是一个 JSON 字符串，解析后为数组，每个元素代表一条字幕片段对象。示例：

```
[
  {
    "text": "红不再是地产与金融的鼓舞，中环的键盘声中，一场重构全球资本格局的科技革命正在发生。",
    "language": "Chinese",
    "index": 0,
    "relativeTime": 0,
    "duration": 7610
  }
]
```

字幕片段字段如下（如存在新增字段，请按“向后兼容”原则忽略即可）：

字段名	必选	类型	说明
text	是	string	字幕文本
language	是	string	语言，当前支持（括号内为中文名称）：`Chinese`（中文）、`English`（英语）、`Tagalog`（他加禄语）、`Thai`（泰语）、`Cantonese`（粤语）、`Korean`（韩语）、`Japanese`（日语）、`Indonesian`（印尼语）、`Vietnamese`（越南语）、`Malay`（马来语）、`Portuguese`（葡萄牙语）、`Turkish`（土耳其语）、`Arabic`（阿拉伯语）、`Spanish`（西班牙语）、`Hindi`（印地语）、`French`（法语）、`German`（德语）、`Uyghur`（维吾尔语）
index	是	number	字幕下标（递增，用于排序/幂等）
relativeTime	是	number	相对时间（毫秒）

字段名	必选	类型	说明
duration	是	number	持续时长（毫秒）

签名（sign）生成与校验

为保证回调请求未被篡改，系统会在请求中携带 `sign`。您方可使用同样算法验签：

- **签名密钥（appSecret）**：`polyvlog`
- **签名算法**：MD5（输出大写 16 进制）
- **参与签名字段**：除 `sign` 本身外的所有请求字段（本回调即 `roomId`、`sessionId`、`subtitles`、`timestamp`）

签名算法步骤

1. 取所有参数（包含 `timestamp`，不包含 `sign`），按 key 的字典序升序排序（等同 JS 的 `Object.keys(data).sort()`）。
2. 按顺序拼接字符串：`key + value`（value 若为对象则 `JSON.stringify`；本回调里 `subtitles` 本身是字符串）。
3. 在拼接串首尾各加一次密钥：`appSecret + 拼接串 + appSecret`
4. 对上一步结果做 MD5，得到十六进制字符串，并转为大写，作为 `sign`。

Node.js 验签示例

```
const crypto = require('crypto');

function sortKeyAndValues(data) {
  return Object.keys(data).sort().reduce((acc, key) => {
    if (key === 'sign') return acc;
    const v = data[key];
    return acc + key + (v && typeof v === 'object' ? JSON.stringify(v) : String(v));
  }, '');
}

function createApiSign(appSecret, data) {
  const md5 = crypto.createHash('md5');
  md5.update(`${appSecret}${sortKeyAndValues(data)}${appSecret}`, 'utf8');
  return md5.digest('hex').toUpperCase();
}

// 验签
function verifySign(body) {
  const { sign } = body;
  const expect = createApiSign('polyvlog', body);
  return String(sign).toUpperCase() === expect;
}
```

时间戳校验建议（可选但推荐）

为防止重放攻击，建议校验 `timestamp` 与当前服务器时间差不超过 30 分钟（按毫秒）。

请求示例（curl）

下面示例展示了系统回调请求的形态（`multipart/form-data` 表单字段）：

```
curl -X POST 'https://api.example.com/polyv/subtitles/callback' \
  -F 'roomId=123456' \
  -F 'sessionId=test_session_id' \
  -F 'subtitles=[{"text":"hello","language":"English","index":0,"relativeTime":0,"duration":1000}]' \
  -F 'timestamp=1700000000000' \
  -F 'sign=YOUR_SIGN'
```

响应要求

- **成功**：请返回 HTTP 2xx（建议 `200`），响应体可为 JSON 或纯文本
- **失败**：返回非 2xx 会被视为失败并触发重试

建议成功响应 JSON 示例：

```
{
  "code": 200,
  "status": "success",
  "message": "ok",
  "data": ""
}
```

常见接入问题

- **无法收到参数/参数为空**：请确认后端是否支持解析 `multipart/form-data`（Express 默认的 `json/urlencoded` 中间件无法解析该格式）。
- **验签不通过**：
  - 确认 `timestamp` 是毫秒；
  - 确认拼接顺序为 key 升序；
  - 确认 MD5 输出需要转大写；
  - 确认参与签名的字段不包含 `sign` 本身，且 `subtitles` 参与签名时按“原始字符串”处理。