

KikoPlay Web接口参考

2022.10 By Kikyou, 本文档适用于KikoPlay 0.9.1及以上版本

KikoPlay局域网服务功能提供了一些Web接口供其他客户端（包括网页端，Android端等）使用。

[Get] /media/{mediaID}

mediaID: 媒体文件ID, 可从/api/playlist返回结果中获取

访问媒体文件。

[Get] /sub/{subFormat}/{mediaID}

subFormat: 字幕文件格式, ass,ssa,srt

mediaID: 媒体文件ID

获取媒体文件的字幕, 字幕文件需要和媒体文件名字相同且位于同一目录, 可先通过 /api/subtitle 检查是否存在字幕文件。

[Get] /api/playlist

获取播放列表。

返回类型: JSON, Array[Node]

Node:

```
{
  "animeName"(string): 动画名称,
  "color"(string): 文字颜色, 16进制字符串,
  "danmuPool"(string): 弹幕池ID,
  "mediaId"(string): 媒体文件ID,
  "playTime"(number): 播放时长(s),
  "playTimeState"(number): 播放状态(0=未播放, 1=未看完, 2=已看完),
  "text"(string): 标题,
  "marker"(number): 标记(0=红色, 1=蓝色, 2=绿色, 3=橙色, 4=粉色, 5=黄色)。如果没有标记则不存在该项,
  "nodes"(Array[Node]): 子节点列表, 如果包含子节点, 则不会包含除了text和marker以外的项目
}
```

[Post] /api/updateTime

更新播放时间, 只有用户在局域网服务中开启“同步播放时间”才有效

JSON数据格式:

```
{
  "mediaId"(string): 媒体文件ID,
```

```
"playTime"(number): 播放时长(s),  
"playTimeState"(number): 播放状态(0=未播放, 1=未看完, 2=已看完),  
}
```

[Get] /api/danmu/v3/

获取弹幕。只包含基本信息。在返回弹幕前，KikoPlay会使用本地的屏蔽规则进行过滤，同时应用延迟和时间轴修改

参数：

id: 弹幕池ID
update: boolean, true/false, 是否更新弹幕池，如果为true则只会返回新添加的弹幕

返回JSON格式：

```
{  
  "code"(number): 始终为0,  
  "data"(Array[CommentL]): 弹幕列表,  
  "update"(boolean): 和参数中的update一致  
}
```

CommentL:

```
[  
  time(number, 弹幕时间(s)),  
  type(number, 0: 滚动, 1: 顶部, 2: 底部),  
  color(number, 弹幕颜色),  
  sender(string, 用户),  
  text(string, 弹幕文本)  
]
```

[Get] /api/danmu/local/

获取和媒体文件同名的本地xml弹幕文件中的弹幕。在返回弹幕前，KikoPlay会使用本地的屏蔽规则进行过滤。当媒体文件未关联到弹幕池时，可通过这个api尝试获取本地弹幕。 参数：

mediald: 媒体文件ID

返回JSON格式：

```
{  
  "comment"(Array[CommentL]): 弹幕列表  
  "local"(string): 本地xml弹幕文件  
}
```

[Get] /api/subtitle

查询字幕文件。返回的type为空表示不存在字幕，可以使用/sub/{subFormat}/{mediaID}获取具体的文件。
参数：

id: 媒体文件ID

返回JSON格式：

```
{
  "type"(string): 字幕文件格式, "", "ass", "ssa", "srt"
}
```

[Get] /api/danmu/full/

获取完整的弹幕池。KikoPlay会使用屏蔽规则过滤，但不会应用延迟和时间轴修改，这些信息包含在source中，首先对弹幕原始时间应用时间轴偏移(timeline)，之后应用整体延迟(delay)，才能得到最终弹幕的时间。如果update=true，则不会返回source和launchScripts。

参数：

id: 弹幕池ID

update: boolean, true/false, 是否更新弹幕池，如果为true则只会返回新添加的弹幕

返回JSON格式：

```
{
  "source"(Array[Source]): 弹幕源信息,
  "comment"(Array[CommentF]): 弹幕列表,
  "launchScripts"(Array[string]): 支持发送弹幕的脚本ID列表,
  "update"(boolean): 和参数中的update一致
}
```

CommentF:

```
[
  time(number, 弹幕原始时间(s)),
  type(number, 0: 滚动, 1: 顶部, 2: 底部),
  color(number, 弹幕颜色),
  src(number, 弹幕源ID),
  text(string, 弹幕文本),
  sender(string, 用户)
]
```

Source:

```
{
  "delay"(number): 整体延迟(ms),
  "duration"(number)": 弹幕源视频时长(s),
  "id"(number): 弹幕源ID,
  "name"(string): 标题,
  "scriptData"(string): 脚本数据,
  "scriptId"(string): 脚本ID,
  "scriptName"(string): 脚本名称,
  "timeline"(string): 时间轴信息, 格式为"时间点1(ms) 偏移1(ms);时间点2 偏移2(ms);...", 时间点对应的偏移会影响时间点之后的全部弹幕
}
```

[Post] /api/updateDelay

更新弹幕池中某个弹幕源的延迟信息。

JSON数据格式:

```
{
  "danmuPool"(string): 弹幕池ID,
  "delay"(number): 新的延迟(ms),
  "source"(number): 弹幕源ID
}
```

[Post] /api/updateTimeline

更新弹幕池中某个弹幕源的时间轴信息。

JSON数据格式:

```
{
  "danmuPool"(string): 弹幕池ID,
  "timeline"(string):时间轴信息, 格式为"时间点1(ms) 偏移1(ms);时间点2 偏移2(ms);...",
  "source"(number): 弹幕源ID
}
```

[Post] /api/screenshot

远程截图/片段。KikoPlay会在后台启动ffmpeg完成截取并加入媒体库中。JSON数据格式:

```
{
  "animeName"(string): 动画名称,
  "mediaId"(string): 媒体文件ID,
  "pos"(number): 截取时间(s),
  "duration"(number): 截取时长(1~15s), 如果包含这一项就是截取片段, 否则为截图,
```

```
"info"(string): 显示在媒体库中的信息
}
```

[Post] /api/danmu/launch

发送弹幕。JSON数据格式:

```
{
  "danmuPool"(string): 弹幕池ID,
  "text"(string): 弹幕文本,
  "time"(number): 弹幕时间(ms), 如果小于0且弹幕池ID和KikoPlay当前播放文件的弹幕池ID
  相同, 直接将弹幕显示到当前位置(如果弹幕池ID相同, 且时间和KikoPlay当前播放时间差距小于
  3s, 也会显示弹幕),
  "color"(number): 弹幕颜色,
  "fontsize"(number): 文字大小, 0: 正常, 1: 大, 2: 小,
  "date"(string): unix时间戳(s),
  "type"(number): 0: 滚动, 1: 顶部, 2: 底部,
  "launchScripts"(Array[string]): 发送脚本ID列表
}
```