ChatGPT

Wwise SoundCaster 独立客户端实现指南

技术选型： 推荐使用 C# 语言开发，借助 Avalonia 或 WPF 框架构建 GUI。Avalonia 是开源跨平台的 .NET UI 工具包，XAML 语法可复用 WPF 知识，支持 Windows/macOS/Linux 等多平台 [1]。若仅限 Windows，可选 WinForms/WPF，开发速度快但移植受限。采用 C# 可利用 Audiokinetic 官方提供的 WAAPI C# 客户端示例库（Wwise SDK 2019.1+） [2] 或第三方 WaapiCS（MIT 许可）封装，简化 WAAPI 调用。总体架构建议分层设计：将 WAAPI 通信逻辑封装在后端服务类（如 `WwiseService`），前端 UI（MVVM 模式）通过绑定和事件与后端解耦交互，便于扩展和维护。

WAAPI 准备与安全： 在 Wwise 编辑器中必须先启用 WAAPI：打开 **Project > User Preferences**，勾选 “Enable Wwise Authoring API” 并重启 Wwise [3]。默认情况下，WAAPI 提供 WAMP（端口8080）和 HTTP（端口8090）两种接口，并**仅允许本机(127.0.0.1)** 连接 [4]。为了安全，建议在防火墙中阻止这两个端口的外部访问；若需远程访问，可在设置中将特定 IP 加入白名单 [4]。WAAPI 还内置跨站脚本安全机制，默认只接受来自本地应用或本地文件的连接 [5]。总之，仅在可信网络环境使用，并限制连接范围。

```
// Create the WAMP connection
connection = new autobahn.Connection({
    url: 'ws://localhost:8080/waapi',
    realm: 'realm1',
    protocols: ['wamp.2.json']
});
```

Wwise 内置 WAAPI 服务器，只需用 WAMP 协议连接其监听端口即可 [6]。例如，通过 URL `ws://localhost:8080/waapi` 建立 WebSocket 连接（需确保 Wwise 正在运行，且已启用 WAAPI [3] [6]）。连接成功后，客户端可通过 RPC 调用或订阅 WAAPI 主题来远程查询和控制 Wwise 项目。下图示例展示了用 AutobahnJS 建立到本机 8080 端口的 WAAPI 连接（C# 客户端也类似） [6]。

事件查询与依赖提取

实现 SoundCaster 功能，需先根据事件名定位 Wwise 事件对象并找出其关联的 RTPC 和 Switch。可按以下步骤操作：

- **查询事件对象：** 使用 WAAPI 方法 `ak.wwise.core.object.get` 按名称或路径查找事件。例如，可构造查询参数：

```
{ "from": { "path": ["\\Events\\Default Work Unit\\MyEventName"] } }
```

这样可获取名为 `MyEventName` 的事件对象（路径需包含项目工作单元），并返回其 ID 等信息 [7]。注意 WAAPI 路径字符串需以双反斜杠开头。

- **列出事件动作：** 获取事件对象后，可使用 **transform** 选项"select children"列出此事件下所有动作（Action） [7]。例如：

```
{
  "from": {"path": ["\\Events\\Default Work Unit\\MyEventName"]},
  "transform": [{"select": ["children"]}]
}
```

返回的结果中包含每个子动作的 `id`, `name` 以及 `@ActionType`、`@Target` 等字段 [7]。我们可根据 `@ActionType` 判断动作类型：若其值为 `GameParameter`（或类似"SetGameParameter"）即表示此动作设定了一个 RTPC；若为 `SetSwitch` 或 `SetState` 则对应 Switch Group/State 动作。通过分析这些动作，可以识别事件所依赖的 RTPC（GameParameter）和 Switch Group。

- **获取参数及状态列表：** 对于识别出的 GameParameter（RTPC）或 SwitchGroup，对应地从项目中查询其属性和选项。例如，可再次调用 `ak.wwise.core.object.get` 针对指定 GameParameter 查询其 `@Min`, `@Max` 等属性，以确定滑块范围；下图即演示如何一次性查询所有 GameParameter（返回 `name,id,@Min,@Max,@SimulationValue`） [8] [9]（示例给出了查询所有 GameParameter 的方式）。对于 Switch Group，可查询该 SwitchGroup 或 SwitchContainer 对象的子项，以获取所有状态名称，填充到下拉列表。通过这些查询，我们便可动态生成控制 RTPC 值和切换状态所需的 UI 控件。

```
var getParamArgs = {
    from: {ofType: ["GameParameter"]}
};
var options = { return: ["name", "id", "@Min", "@Max", "@SimulationValue"]};

wampsession.call(ak.wwise.core.object.get, [], getParamArgs, options).then(
    function (objectspayload) {
        ///...
```

示例：使用 `ak.wwise.core.object.get` 查询所有类型为"GameParameter"的对象 [9]。查询参数中 `from: { ofType:["GameParameter"] }` 会返回项目中所有 GameParameter 及其属性（如下图所示返回了每个参数的 name、id 及数值范围）。类似方法可用于获取特定对象的属性列表。

控制界面与交互

- **RTPC 控件：** 为每个查询到的 RTPC (GameParameter) 在界面上创建滑块或数值输入框。滑块范围可设定为参数的最小值到最大值，中间值映射到参数范围。当用户拖动滑块时，程序调用 WAAPI 接口 `ak.soundengine.setRTPCValue` 将数值实时发送给 Wwise 声音引擎 [10]。例如：

```
await waapiClient.Call("ak.soundengine.setRTPCValue", new {
    rtpc = parameterId, value = newValue, gameObject = myGameObjectId
});
```

如 [54] 所示，`setRTPCValue` 可设置指定 Game Object 上的游戏参数值 [10]。注意此处需要先注册一个游戏对象（见下文）。

- **Switch 控件：** 对每个相关的 Switch Group 在 UI 上使用下拉菜单或按钮组。下拉列表项为该 SwitchGroup 下所有可用的 State 名称。用户选择状态后，调用 `ak.soundengine.setSwitch` 将对应 Group 和 State 发送给引擎 [11]：

```
await waapiClient.Call("ak.soundengine.setSwitch", new {
    inGameObject = myGameObjectId,
    inSwitchGroup = switchGroupId,
    inSwitchState = selectedStateId
});
```

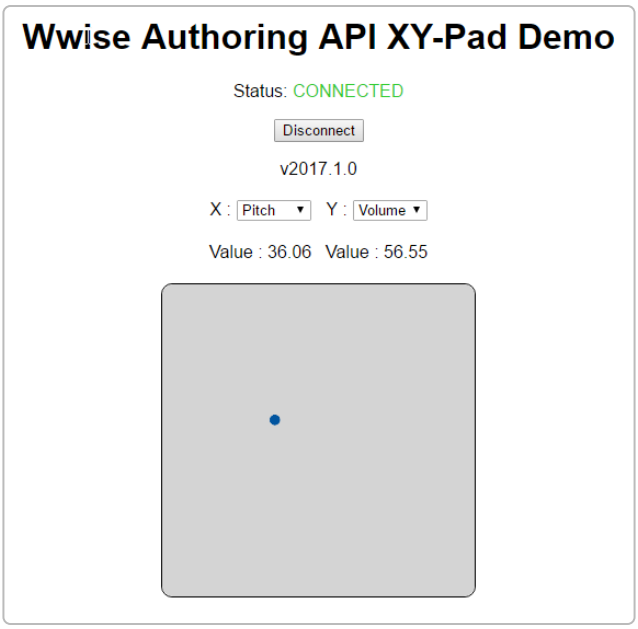
[54] 中 `setSwitch` 的说明指出可设置一个 Switch Group 的当前状态 [11]。同样要提供游戏对象 ID。

- **播放/停止控制：** UI 上提供事件播放和停止按钮。点击播放时，使用 `ak.soundengine.postEvent` 将事件异步发送给声音引擎 [12]；停止时可以调用 `ak.soundengine.stopAll`（或针对特定对象停止）[13]。例如：

```
await waapiClient.Call("ak.soundengine.postEvent", new {
    inEvent = eventId, inGameObject = myGameObjectId
});
```

`postEvent` 会触发引擎播放该事件 [12]。同样，需要指定 gameObject。

- **游戏对象注册：** 因为上述 RTPC、Switch、播放等需要一个游戏对象上下文，应用启动后需先调用 `ak.soundengine.registerGameObj` 注册一个 Game Object，并保存返回的 `gameObjectId` [14]。所有后续的声音引擎操作都以此 ID 为目标即可 [14]。



下图示例展示了一种交互界面：一个二维 XY 平面允许用户同时控制两个参数（Pitch 和 Volume） [15]。移动蓝色小点时即通过 WAAPI 调整关联的 GameParameter 值 [15]。在我们的工具中，可采用类似滑块或其他控件来控制多个 RTPC，并使用下拉菜单等控件设置 Switch 状态。上述示例说明：首先通过 WAAPI 获取参数列表（此例获得了 Pitch 和 Volume 参数），然后在界面上实时更新它们 [15]。

其他要点

- **网络协议和实现：** WAAPI 支持 WAMP 和 HTTP 两种协议 [16]。C# 客户端一般使用 WAMP (WebSocket)，也可用 HTTP POST。Audiokinetic 官方推荐使用其 C# WAAPI 客户端示例（2019.1 以后版本提供），它简化了 JSON 调用过程 [2] [17]。开发时可引用 Audiokinetic 提供的 `Ak.Wwise.Authoring.WaapiClient` 类，或使用第三方库如 WaapiCS。

- **解耦设计：** 应将 WAAPI 逻辑与 UI 严格分离。前端通过事件或数据绑定调用后端服务（如 `WwiseService.PostEvent(...)`），后端完成 WAAPI 调用并返回结果。所有 WAAPI 调用应为异步操作，避免阻塞 UI 线程。

- **3D 声音扩展（可选）：** 若需要 3D 交互，可以可视化声源并更新其空间位置。例如，可在 UI 中拖拽声源坐标，然后调用 `ak.soundengine.setPosition` 设置游戏对象的三维坐标 [18]。这可实现 3D 环境下的声音渲染，但实现复杂，需要额外引擎或库支持，非必要功能。

- **7 天开发计划示例：** 日程可分为：Day1-2：搭建 C# + Avalonia 项目，测试 WAAPI 连接；Day3：实现事件名称查询及依赖提取功能；Day4-5：构建 UI 控件（RTPC 滑块、Switch 下拉）并接入 WAAPI 调用；Day6：完善播放/停止功能和错误处理；Day7：调试、文档编写和扩展测试。前端界面可利用 AI 工具快速生成原型，但需保证与后端 WAAPI 调用正确绑定。

参考资料： Wwise 官方文档说明 WAAPI 可控制 GameParameter 和 Switch 等 [19]；示例博客展示如何连接 WAAPI 并获取项目对象 [20] [9]；WAAPI 函数参考表明可使用

`ak.soundengine.postEvent/setRTPCValue/setSwitch` 等完成控制 [12] [10] [11] [18]；Audiokinetic 社区建议使用新 C# WAAPI 客户端 [2]。以上内容为实现指南所依据的关键点。

---

[1] Avalonia UI – Open-Source .NET XAML Framework | WPF & MAUI Alternative
https://avaloniaui.net/

[2] [17] how to use ak.wwise.core.object.get - Community Q&A
https://www.audiokinetic.com/qa/6367/how-to-use-ak-wwise-core-object-get

[3] [4] [5] 准备使用 Wwise Authoring API
https://www.audiokinetic.com/zh/library/edge/?source=SDK&id=waapi_prepare.html

[6] [9] [15] [16] [20] A Step-by-Step WAAPI Example | Audiokinetic Blog
https://www.audiokinetic.com/en/blog/stepbystep-waapi-example/

[7] Best way to find Event references for Audio objects via WAAPI - Community Q&A
https://www.audiokinetic.com/qa/5842/best-way-to-find-event-references-for-audio-objects-via-waapi

[8] info.audiokinetic.com
http://info.audiokinetic.com/hubfs/Blog_Images/WAAPI%202/image3.png

[10] [11] [12] [13] [14] [18] Wwise Authoring API Reference - Functions
https://www.audiokinetic.com/en/public-library/2025.1.4_9062/?source=SDK&id=waapi_functions_index.html

[19] Using the Wwise Authoring API (WAAPI)
https://www.audiokinetic.com/en/library/edge/?source=SDK&id=waapi.html