这是一个基于你提供的 手绘草图 **(ui design.jpg)**、**Wwise** 文档以及 架构设计指南 整理而成的 **UI** 设计与技术规格说明书。

这份文档专为生成 AI Prompt 或直接指导代码编写而设计，详细描述了 **C# + Avalonia MVVM** 架构下的布局、控件层级、数据绑定逻辑和样式规范。

Wwise SoundCaster 独立客户端 - UI 设计规格说明书

1. 项目概述

- 目标框架: .NET 8 / C#
- **UI** 框架: Avalonia UI (MVVM Pattern)
- 设计目标: 实现一个 Wwise SoundCaster 的独立客户端，用于浏览事件结构、实时调试 RTPC 和 Switch 开关。
- 参考依据: ui design.jpg (布局), 软件架构设计概述.pdf (MVVM/WAAPI 逻辑).

---

2. 全局布局架构 (Global Layout)

根容器: Grid

- 列定义 **(ColumnDefinitions)**: 1*, 4, 3*
  - Col 0: 左侧事件浏览区 (Left Panel - Event Browser) - 占比约 25-30%。
  - Col 1: GridSplitter (允许用户拖拽调整左右比例)。
  - Col 2: 右侧操作详情区 (Right Panel - Operation Detail) - 占比约 70-75%（根据需求“右半区更大”）。
- 行定义 **(RowDefinitions)**: * (充满全屏)。

---

3. 详细区域设计 (Component Detail)

3.1. 左侧：事件结构概览区 (Left Panel)

对应手绘图左侧部分。

- 布局容器: DockPanel 或 Grid (Rows: Auto, *)
- 顶部：搜索栏 **(Search Bar)**
  - 控件: TextBox
  - 属性:

        - Watermark: "搜索事件名称..."
        - Margin: "10"
    - **MVVM** 绑定:
        - Text -> {Binding SearchKeywords, Mode=TwoWay} (在 VM 中实现防抖动过滤)。
- 主体：树形列表 **(Event Tree)**
    - 控件: TreeView
    - 位置: 位于搜索栏下方，填满剩余空间。
    - **MVVM** 绑定:
        - ItemsSource -> {Binding EventTreeNodes}
        - SelectedItem -> {Binding SelectedEventNode, Mode=TwoWay}
    - 数据模板 **(DataTemplate)**:
        - 使用 HierarchicalDataTemplate。
        - 显示内容: StackPanel (水平排列) -> PathIcon (文件夹/事件图标) + TextBlock (名称)。
    - 交互: 点击树节点触发右侧面板的数据更新。

3.2. 右侧：具体操作区 (Right Panel)

对应手绘图右侧部分。此区域仅当 SelectedEventNode 不为空时显示内容。

布局容器: Grid

- 行定义: Auto, Auto, Auto, *, Auto (分别对应：头部、RTPC区、Switch区、空白填充、底部反馈)。

A. 事件概览头 (Header - Event Info) [Row 0]

- 外观: 带边框的卡片式设计。
- 布局: DockPanel 或 Grid (Cols: *, Auto)。
- 左侧: TextBlock
    - 内容: 当前选中的事件名称。
    - 样式: 大号字体 (FontSize=24), 加粗。
    - 绑定: {Binding SelectedEventNode.Name}。
- 右侧: 播放控制组 (StackPanel Orientation="Horizontal")
    - 停止按钮: Button (Icon: Square/Stop). Command -> {Binding StopEventCommand}.
    - 播放按钮: Button (Icon: Play/Triangle). Command -> {Binding PlayEventCommand}.
    - 暂停按钮 **(可选)**: 根据草图有 "||" 图标。

B. RTPC 操作区 (RTPC Control Section) [Row 1]

- 容器: ItemsControl (用于动态生成不定数量的滑块)。
- 绑定: ItemsSource -> {Binding CurrentEventRTPCs}。
- **ItemTemplate (单行 RTPC 模板)**:
    - 布局: Grid (Cols: 100, *, 50) -> 左中右结构。
    - **Col 0 (标签)**: TextBlock

        - Text: {Binding Name}
        - VerticalAlignment: Center
    - **Col 1 (滑块)**: Slider
        - Minimum: {Binding MinValue}
        - Maximum: {Binding MaxValue}
        - Value: {Binding CurrentValue, Mode=TwoWay} (拖动时实时触发 SetRTPC)。
        - TickFrequency: 根据范围自动计算。
    - **Col 2 (数值)**: TextBlock
        - Text: {Binding CurrentValue, StringFormat={}{0:F1}}
        - VerticalAlignment: Center
        - HorizontalAlignment: Right
- 视觉样式: 外层包裹 Border，统一背景色，Margin="0,10,0,10"。

C. Switch 操作区 (Switch Group Section) [Row 2]

- 容器: ItemsControl
- 绑定: ItemsSource -> {Binding CurrentEventSwitches}。
- **ItemsPanel (布局逻辑)**:
    - 要求: 每行展示两个 Switch。
    - 实现: 使用 UniformGrid (Columns="2") 或者 WrapPanel。
- **ItemTemplate (单个 Switch 组模板)**:
    - 布局: StackPanel (Orientation="Vertical") 或 GroupBox。
    - 标签: TextBlock -> {Binding GroupName}。
    - 控件: ComboBox (下拉菜单)。
        - ItemsSource: {Binding AvailableStates} (List<String>)。
        - SelectedItem: {Binding SelectedState, Mode=TwoWay}。
        - HorizontalAlignment: Stretch。
- 间距: Margin="5"，确保两个 Switch 之间有空隙。

D. 底部反馈区 (Feedback Footer) [Row 4]

- 控件: Border (背景色: 深红/深绿/灰色，取决于状态)。
- 内容: TextBlock
- 绑定:
    - Text: {Binding StatusMessage} (例如："Connected to localhost:8080" 或 "Connection Failed").
    - Foreground: 根据状态变色 (使用 Converter)。
- 位置: 锚定在右半区的最底部。

---

4. MVVM 数据结构参考 (For Code Generation)

AI 在生成 ViewModel 代码时应参考以下结构：

C#

```
// 核心视图模型
public class MainViewModel : ViewModelBase
{
    // 左侧树数据
    public ObservableCollection<EventNodeViewModel> EventTreeNodes { get; }

    // 当前选中事件
    public EventNodeViewModel? SelectedEventNode { get; set; } // [Reactive]

    // 右侧面板数据源 (根据 SelectedEventNode 动态加载)
    public string EventName { get; } // [Reactive]
    public ObservableCollection<RtpcViewModel> CurrentEventRTPCs { get; }
    public ObservableCollection<SwitchGroupViewModel> CurrentEventSwitches { get; }

    // 底部状态
    public string StatusMessage { get; } // [Reactive]

    // Commands
    public ICommand PlayEventCommand { get; }
    public ICommand StopEventCommand { get; }
}

// RTPC 子项模型
public class RtpcViewModel : ViewModelBase
{
    public string Name { get; set; }
    public double MinValue { get; set; }
    public double MaxValue { get; set; }
    public double CurrentValue { get; set; } // [Reactive] -> 触发 WAAPI SetRTPCValue
}

// Switch 子项模型
public class SwitchGroupViewModel : ViewModelBase
{
    public string GroupName { get; set; }
    public List<string> AvailableStates { get; set; }
    public string SelectedState { get; set; } // [Reactive] -> 触发 WAAPI SetSwitch
}
```

---

5. 样式建议 (Styling Prompt)

提供给 AI 的样式提示词：

> "Use a modern, dark theme inspired by professional audio software (like Wwise or Digital Audio Workstations). Use deep gray backgrounds (#1E1E1E, #252526), lighter gray borders, and an accent color (e.g., Purple #9C27B0 or Blue #007ACC) for sliders and active selections. Ensure text contrast is high (White/Light Gray). Use standard Avalonia styles or FluentTheme."

---

6. AI 提示词 (Prompt Template)

你可以直接复制下面的 Prompt 发送给 AI 编码助手：

> **Role**: You are an expert C# Avalonia developer proficient in MVVM and WAAPI.
>
> **Task**: Generate the XAML (View) and C# (ViewModel) code for a Wwise SoundCaster client based on the attached design requirements.
>
> **Design Requirements**:
>
> 1. **Layout**: Use a Grid with two columns. Left column (30%) is for the Event Tree; Right column (70%) is for controls.
> 2. **Left Panel**: Contains a TextBox for search (top) and a TreeView (fill) binding to EventTreeNodes.
> 3. **Right Panel**:
>   - **Top**: A header showing EventName and Play/Stop buttons.
>   - **Middle 1**: An ItemsControl for RTPCs. Use a Grid template: Name (Left), Slider (Middle, stretch), Value (Right). Bind Slider Value TwoWay.
>   - **Middle 2**: An ItemsControl for Switches. Use a UniformGrid with Columns="2" as the ItemsPanel. Template contains a Group Name and a ComboBox.
>   - **Bottom**: A Status Bar TextBlock showing connection status.
> 4. **Tech Stack**: Use Avalonia UI, ReactiveUI (or CommunityToolkit.Mvvm).
>
> **Referenced Models**:
>
> Create ViewModels for RtpcViewModel (Name, Min, Max, CurrentValue) and SwitchGroupViewModel (GroupName, AvailableStates, SelectedState).
>
> Please generate the MainWindow.axaml and MainWindowViewModel.cs following

this structure.