

理解文明6音视频绑定逻辑（中）

好的，下面是对 **文明6（Civ6）** 中 **SoundBank** 绑定 **.wem** 音频文件的逻辑总结。

1. 基本概念

在《文明6》游戏中，音频文件使用 **Wwise** 引擎管理和播放。**.wem**（Wwise Encoded Media）是 **Wwise** 使用的音频文件格式。游戏中的音效和背景音乐通过 **SoundBank** 进行管理和加载。

- **SoundBank**: 是一个集合，包含了音效事件、音频文件（如 **.wem** 文件）以及其他音频资源信息。每个 **SoundBank** 会包含多个音效文件。
- **.wem**: 是 **Wwise** 引擎使用的压缩音频格式，用于存储游戏中的音效或音乐。**.wem** 文件包含音频的编码数据。
- **Event**: 指示游戏中某个特定时刻触发的音效，比如播放一段背景音乐、人物对话等。

2. SoundBank 的结构和内容

一个 **SoundBank** 文件通常包含以下几个部分：

- **SoundBank** 标签：每个 **SoundBank** 对应一个 XML 配置。
 - **Id**: SoundBank 的唯一标识符。
 - **ShortName**: SoundBank 的简短名称，用于引用和加载。
 - **IncludedEvents**: 包含的事件，指明该 SoundBank 关联的音效事件。
 - **ReferencedStreamedFiles**: 指向 **.wem** 文件的引用路径。音效会存储为 **.wem** 格式，且通过这些引用路径进行关联。
 - **IncludedMemoryFiles**: 一些内存中的音效文件（用于较高性能需求的音效）。

3. XML 配置文件与音效绑定

在游戏的 XML 配置文件中，**SoundBank** 会绑定 **.wem** 文件。每个 XML 文件通常包含多个 **SoundBank**，其中绑定了不同的音效或音乐资源。

示例结构：

```

<SoundBanks>
    <SoundBank Id="4209095829" GUID="{B63CD4A0-1A5C-476A-B6FD-1
795E2B4AFA9}" Language="Chinese(Taiwan)">
        <ShortName>XP2_Cinematic_Intro_Speech</ShortName>
        <Path>Chinese(Taiwan)\XP2_Cinematic_Intro_Speech.bnk</Path>
        <IncludedEvents>
            <Event Id="173393568" Name="Play_Cinematic_Intro_EXP2" GUID
            ="{697FDF6B-3FC8-4A35-8AB0-1FCD1B5100F4}" />
            <Event Id="1016118146" Name="Stop_Cinematic_Intro_EXP2" GUID
            ="{903AC2F4-C39B-4E7F-ADE4-7E4E27C51586}" />
        </IncludedEvents>
        <ReferencedStreamedFiles>
            <File Id="1005480662" Language="Chinese(Taiwan)">
                <ShortName>CIV6_VESUVIUS_SURR_DX.wav</ShortName>
                <Path>Voices\Chinese(Taiwan)\CIV6_VESUVIUS_SURR_DX_E1E65
BF4.wem</Path>
            </File>
        </ReferencedStreamedFiles>
        <IncludedMemoryFiles>
            <File Id="1005480662" Language="Chinese(Taiwan)">
                <ShortName>CIV6_VESUVIUS_SURR_DX.wav</ShortName>
                <Path>Voices\Chinese(Taiwan)\CIV6_VESUVIUS_SURR_DX_E1E65
BF4.wem</Path>
                <PrefetchSize>324064</PrefetchSize>
            </File>
        </IncludedMemoryFiles>
    </SoundBank>
</SoundBanks>

```

解释：

- **IncludedEvents**: 列出了与 **SoundBank** 关联的音效事件。
- **ReferencedStreamedFiles**: 列出了该 **SoundBank** 中包含的所有 **.wem** 文件，每个 **.wem** 文件通过 **File Id** 和 **Path** 引用。文件路径（如 **CIV6_VESUVIUS_SURR_DX_E1E65BF4.wem**）指向实际的 **.wem** 音频文件。
- **IncludedMemoryFiles**: 包含一些内存中的音效文件，用于特定的优化或性能需求。

4. 音频文件的关联

每个 **.wem** 文件通过其 **路径**（相对路径）和 **File Id** 绑定在 **SoundBank** 中。**SoundBank** 文件定义了该音效的播放事件，触发这些事件时，游戏会加载相关的 **.wem** 文件进行播放。

5. 事件与 SoundBank 的关系

- **Events** 是游戏中触发的音效，通常是通过游戏内的脚本触发，例如：当用户进入某个场景时，触发一个音效。
- **SoundBank** 中的 **IncludedEvents** 节点列出了每个事件和相关的音效文件。一个事件通常会触发多个 **.wem** 音频文件的播放。

6. 如何加载和播放音效

在游戏引擎中，加载和播放音效的过程如下：

1. **加载 SoundBank**：首先加载对应的 **SoundBank**。
2. **事件触发**：根据游戏进程触发 **Event**，例如开始某个场景时，触发播放背景音乐的事件。
3. **播放 .wem 文件**：根据 **Event** 中绑定的 **.wem** 文件路径，游戏引擎加载并播放音效。

7. 音频文件的转换

如果你需要将 **.wem** 文件转换成 **.wav** 文件用于进一步处理，通常使用 **vgmstream-cli** 工具来执行这种转换。你只需提供 **.wem** 文件的路径和输出路径，工具会将 **.wem** 转换成 **.wav** 格式。

```
vgmstream-cli.exe input.wem output.wav
```

7.核心思路

现在是时候进入下一步了。我今天已经想好怎么来导出并管理这些音频文件了。

第一、大的方面，脚本必须保持传入参数的原始目录结构，以我传入的

"F:\Civ6_Assets\Audio\Expansion2\test_from"地址为例，其目录结构是

test_from

|- Chinese(PRC)

|- Chinese(Taiwan)

|- English(US)

|- ...

| - 一些其他的本地化文件夹
| - 本地的一些*.wem文件及.bnk、.txt、.xml
那么输出目录请先保持一样的结构。

第二、在上面的目录结构下，开始解析.xml，然后开始解析根目录的.xml，然后按照xml名字创建文件夹，比如在test_from根目录下有一个XP2_UI_Bank，那么在输出目录下需要创建一个XP2_UI_Bank文件夹。再比如在test_from\Chinese(Taiwan)目录下存在XP_Common_Speech.xml，则在test_from\Chinese(Taiwan)目录下创建一个XP_Common_Speech文件夹。然后将该xml内ReferencedStreamedFiles节点下包含所有的<File>节点进行转化，根据File Id，把对应的<File Id>.wem文件转成wav文件，wav文件名就用<File>节点下<ShortName>值，wav文件保存在对应的.xml命名的文件夹内，如果xml内不存在<ReferencedStreamedFiles>，则忽略，保持文件夹为空文件夹，不用对<IncludedMemoryFiles>里面的进行转换。

第三、请你帮我在<SoundBankName>文件夹内创建一个文件，名字叫included_memory_files（最好是excel），用来记录<IncludedMemoryFiles>节点信息。具体来说文件有2列，第1列填该<IncludedMemoryFiles>内包含的每个File的<File Id>，第2列填<File ShortName>，其中<IncludedMemoryFiles>包含多少个<File>根节点，该文件就有多少行。

第四、请把上面第二步，即在解析SoundBank.xml文件时，把<File Id>.wem转成wav时遇到的<File Id>.wem文件找不到的情况记录下来，最好也是记录到excel，名字叫missing_wem(最好是excel)。

该文件包含两列，第1列是丢失的wem文件Id，第2列该文件<ShortName>的值。

第五、我需要你帮我记录Event和SoundBank的对应关系。具体需求如下，我需要你在输出文件夹的根目录下，帮我生成一个events的映射关系的文件（最好是excel），其中第1列为主键，值为Event Id，第2列

值为Event Name，第3列值为SoundBank ShortName，一般都是.xml文件名，该列值运行同时存在多个值，如果记录文件是excel(则把该值写到下一行，具体举例，A1填Event Id，B1填EventName，从C1开始

填写SoundBank Name，C1有值了就写C2，依次类推)。如果用面向对象的编程模型来实现的话，需要定义一个对象，我们假设就叫EventModel，该对象包含三个属性，第1个属性是字符串类型的Event Id，

第2个属性是字符串属性的EventName，第3个属性是一个Set类型的SoundBanks。

然后再用一个Map(或是字典，反正是Key-Value类容器)来存储这些EventModel，这个Map的Key是EventModel的Event Id，值是对应的EventModel对象。然后在解析具体的SoundBank.xml时，如果文件里面有<IncludedEvents>节点。则依次解析里面的<Event>节点，在解析时，先把EventId拿出来，去Map容器查一下对应的EventModel是否已经存在。如果存在，则拿出该EventModel对象，往该对象的SoundBanks属性里append一个SoundBank；如果Map容器里不存在该对象，则初始化一个EventMode对象，把Event Id，Event Name赋值给该对象，同时往SoundBanks属性里写入一个SoundBank值，并把该EventMode对象添加到Map容器里。当程序处理完所有.xml文件后，遍历该Map容器，再把EventMode对象的值写入到Excel就行了。

总结

1. **SoundBank** 是《文明6》音频管理的核心，用于将音效和事件进行组织和管理。
2. 每个 **SoundBank** 包含多个 **Event** 和多个 **.wem** 文件，定义了音效的加载和触发方式。
3. 游戏会通过事件加载相关的 **.wem** 文件并进行播放。
4. 通过解析 XML 配置文件，能确定每个 **SoundBank** 和音效文件之间的关系，并且可以通过工具（如 **vgmstream-cli**）将 **.wem** 转换为其他格式的音频文件。

如果你有任何其他问题或需要进一步细化的内容，欢迎随时提问！