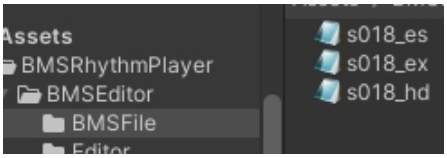
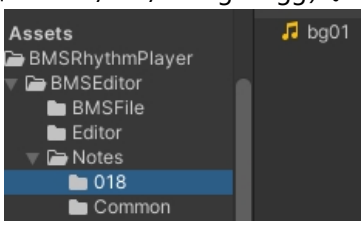
概要：

BMSRhythmPlayer 是一个扩展工具，可以在 Unity 中播放由 iBMSC 导出的 BMS 文件， 并且可以将 BMS 文件信息转换为 Asset 数据进行保存。

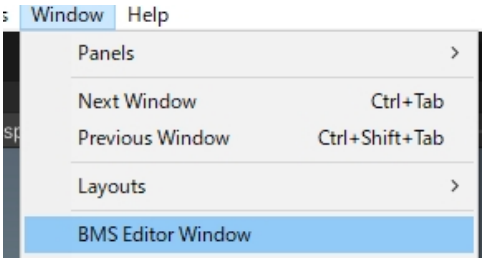
使用 Editor 扩展功能，将 BMS 文件转换为 Asset 数据。 请将 BMS 文件放入 **BMSFile** 文件夹中。



音源文件等请放入 **Notes** 文件夹中（例如：Notes/018/018bgm.ogg）。

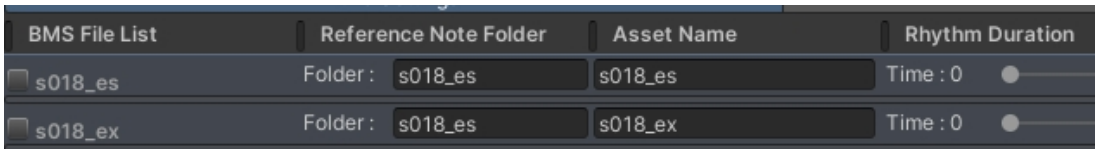


从菜单 **Window → BMS Editor Window** 打开扩展窗口。

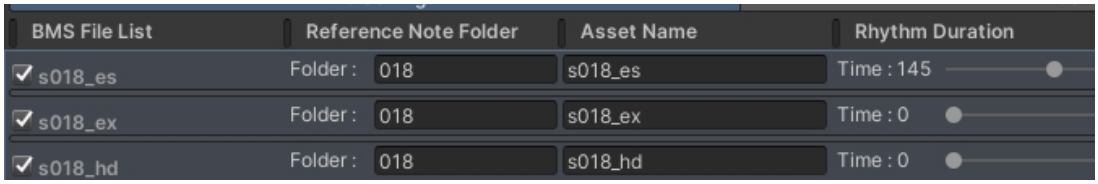


BMS File List：显示 BMS 文件名称。
Reference Note Folder：指定音源文件参考文件夹。
Asset Name：指定保存为 Asset 数据时使用的名称。
Rhythm Duration：指定节奏长度。 如果为
0，则会根据最后一个按键自动计算最合适的时间长度。（如果时间不合适，请手动调整。）

示例：变更前



示例：变更后



选择 Track Settings 面板进行轨道设置。

使用 **[+ Add Track]** 和 **[- Delete Track]** 添加或删除轨道。

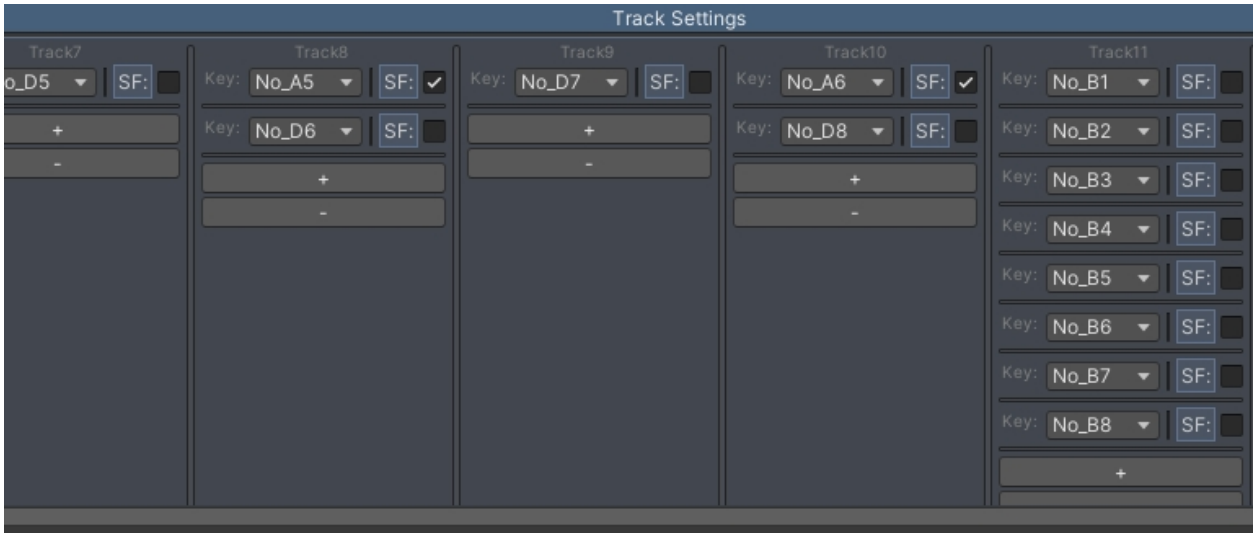
KEY：

选择希望在该轨道上移动的按键编号。

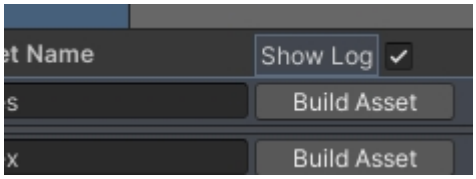
按键编号对应 iBMSC 的A1 ~ B15。(B1 ~ B15为 BGM 轨道)

SF（SimultaneouslyFalling）：

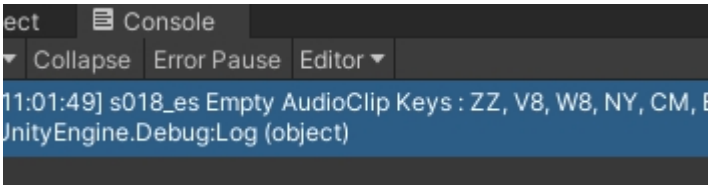
勾选后，会在所选按键之间进行“同时下落”处理。



Show Log：构建 Asset 完成后，会输出找不到音源文件的按键。 如果这些按键不使用音源文件，可以忽略。



Log表示参考図：



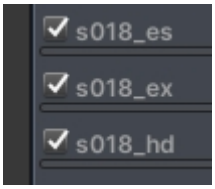
Save Settings：完成构建设置和轨道设置后，如果希望保存当前配置，

点击 **[Save Settings]** 按钮。

将会生成：BMSEditorPlayer/Resources/BMSEditorReference.json

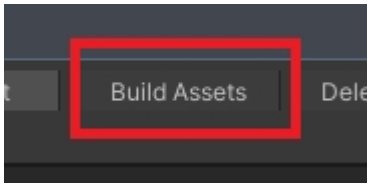
构建单个 Asset：点击 **[Build Asset]**。

构建多个 Asset：勾选想要构建的 BMS 文件



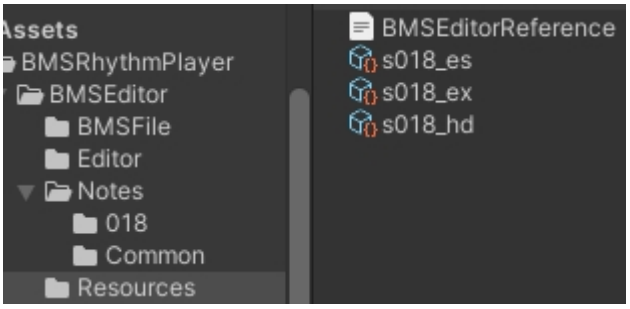
↓

点击 **[Build Assets]**。



↓

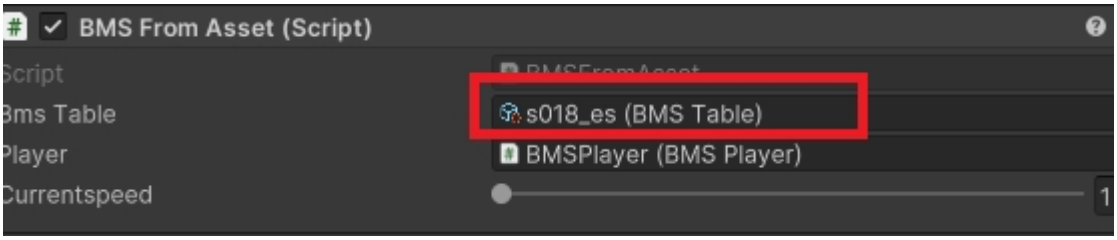
生成的 Asset 数据会保存到：



使用构建好的 Asset开始播放

使用 Demo：**BMSFromAsset**

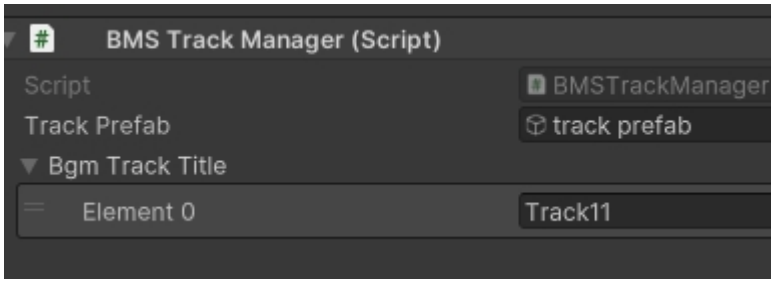
Bms Table：指定要使用的 Asset。



↓

Bgm Track Title：指定 BGM 轨道标题。

完成后即可播放确认。



BMSTableを 类的引用方法

获取按键总数（Long Note 计两次）

public int GetNoteCount

获取基础速度

public float GetBasicNoteSpeed

获取节奏总时长

public float GetRhythmDuration

获取 BPM 轨道信息（用于变速）

public List GetBpmTrackInfo()

获取所有轨道信息（包含 BGM，不含 BPM 轨道）

public List GetTrackPackage()

获取变速信息（用于每帧计算）

index：从 0 开始的 BPM 轨道列表索引用 index，并返回下一个 index。

time：每帧经过的时间

totalTime：从开始到现在的累计时间

basicVariableSpeed：当前的基础速度

public BPMVariableInfo GetBPMVariableSpeedInfo(int index, float time, float totalTime, float basicVariableSpeed)

生成轨道：请参考 Demo 中的 BMSTrackManager 类。

由于 **BGM 轨道没有按键的生成、显示、位置设置**，因此需要单独处理。

生成按键：请参考 Demo 中的 BMSTrack 类。

当 simultaneouslyFall 为 true 时，会生成“同时下落”的按键。

```
if(_track.noteInfo.keyType == KEY_TYPE.Short_Note)
{
    if (_track.noteInfo.simultaneouslyFall)
        noteOb = GameObject.Instantiate(shortRedPrefab);
```

ShortNote（短按键）

生成位置：

```
var notePoint = (info.noteInfo.point[0] - totalDistance) * gearSpeed;

transform.localPosition = new Vector3(transform.localPosition.x, notePoint, 0f);
```

LongNote（长按键）

生成位置同上

长度：

```
var width = info.noteInfo.noteLong * gearSpeed;

widthRect.sizeDelta = new Vector2(width, widthRect.rect.height);
```

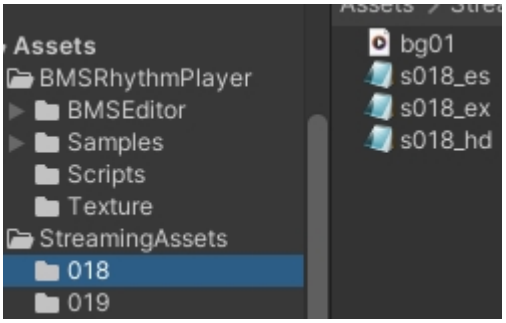
移动距离（参考 BMSPlayer）

```
var f_time = Time.deltaTime;
totaltime += f_time;

var variableInfo = table.GetBPMVariableSpeedInfo(variableIndex, f_time, totaltime, basicVariableSpeed);
variableIndex = variableInfo.index;
basicVariableSpeed = variableInfo.basicVariableSpeed;
totalBasicDistance += variableInfo.distance;
var moveDistance = variableInfo.distance * gearSpeed;

trackmanager.Run(generateTimer, f_time, totaltime, gearSpeed, moveDistance, totalBasicDistance);
```

从 StreamingAssets加载BMS文件播放

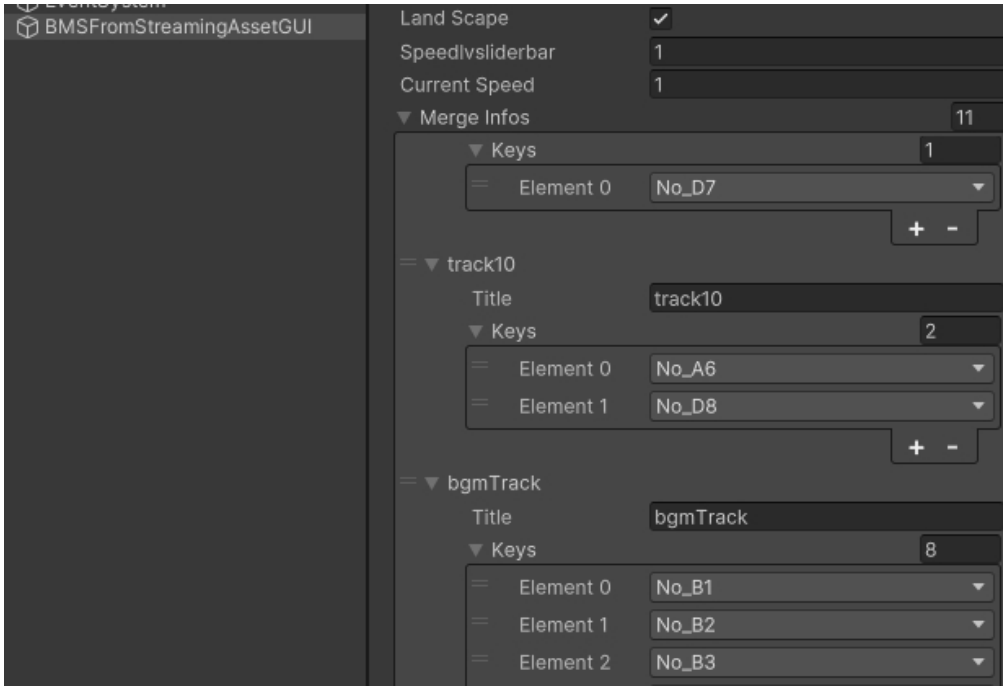
使用 Demo：**BMSFromStreamingAssets**

请将 BMS 文件放入：

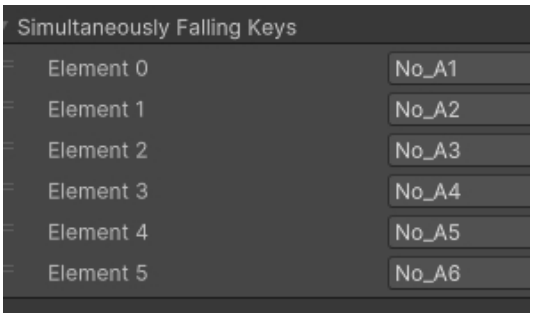


MergeInfos

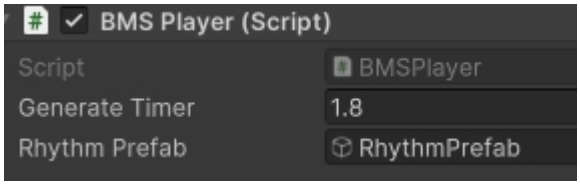
编辑轨道标题与按键编号（A1~B15）。



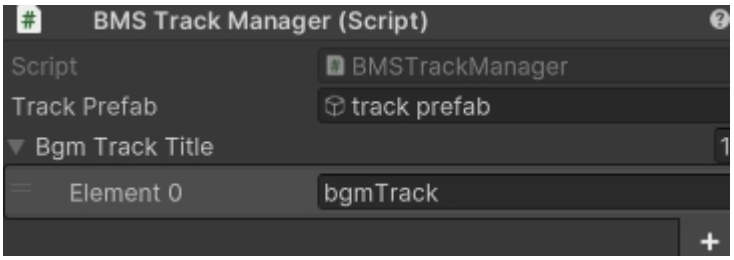
Simultaneously Falling Keys：处理同时下落的按键（例如：同色显示）。



GenerateTimer：指定按键生成并下落的时间点。



Bgm Track Title：指定 BGM 轨道标题（需与 MergeInfos 中一致）。



播放 StreamingAssets 中的 BMS

选择 BMS 文件 → 点击 Play。

Error：当音源文件未指定或找不到时显示。

如果该按键不使用音源文件，可忽略。

Empty AudioClip Keys：ZZ, UP...