时间系统

重点部分：

1. 时间变化影响光照、音效、雾
2. timeScale
3. 时间变化同步UI

核心脚本：

1. TimeManager

TimeManager Init

1. 设置初始迷雾效果
2. 设置当前时间阶段（早晨、中午、晚上、深夜）的背景音效
3. 触发事件管理器更新UI界面

TimeManager UpdateTime

1. 更新当前时间 `timeData.calcTime -= Time.deltaTime * timeScale;  // 减掉每一帧的时间, 乘上缩放系数便于debug`
2. 检查当前是否进入下一个时间阶段，如果进入下一阶段则更新光照强度、光照角度等数据并触发UI更新当前界面并播放新的音效

时间系统数据结构

```
// Unity光源组件, 目前使用的是方向光
[SerializeField] private Light mainLight;   // 太阳(主要光照)
// 时间流逝比例
[SerializeField,Range(0,30)] public float timeScale = 1;
// 时间数据: 用于数据存档, 恢复存档时
private TimeData timeData;
// 时间配置: 持续时间、光照强度、光照角度、光照颜色、背景音乐
private TimeConfig timeConfig;
// 当前天数
public int currentDayNum { get => timeData.dayNum; }
// 时间配置索引
private int nextIndex;

public class TimeData
{
    public int stateIndex = 0;          // 游戏时间状态配置
    public float calcTime = 0;          // 时间倒计时
    public int dayNum;                  // 天数
}
```

时间系统配置

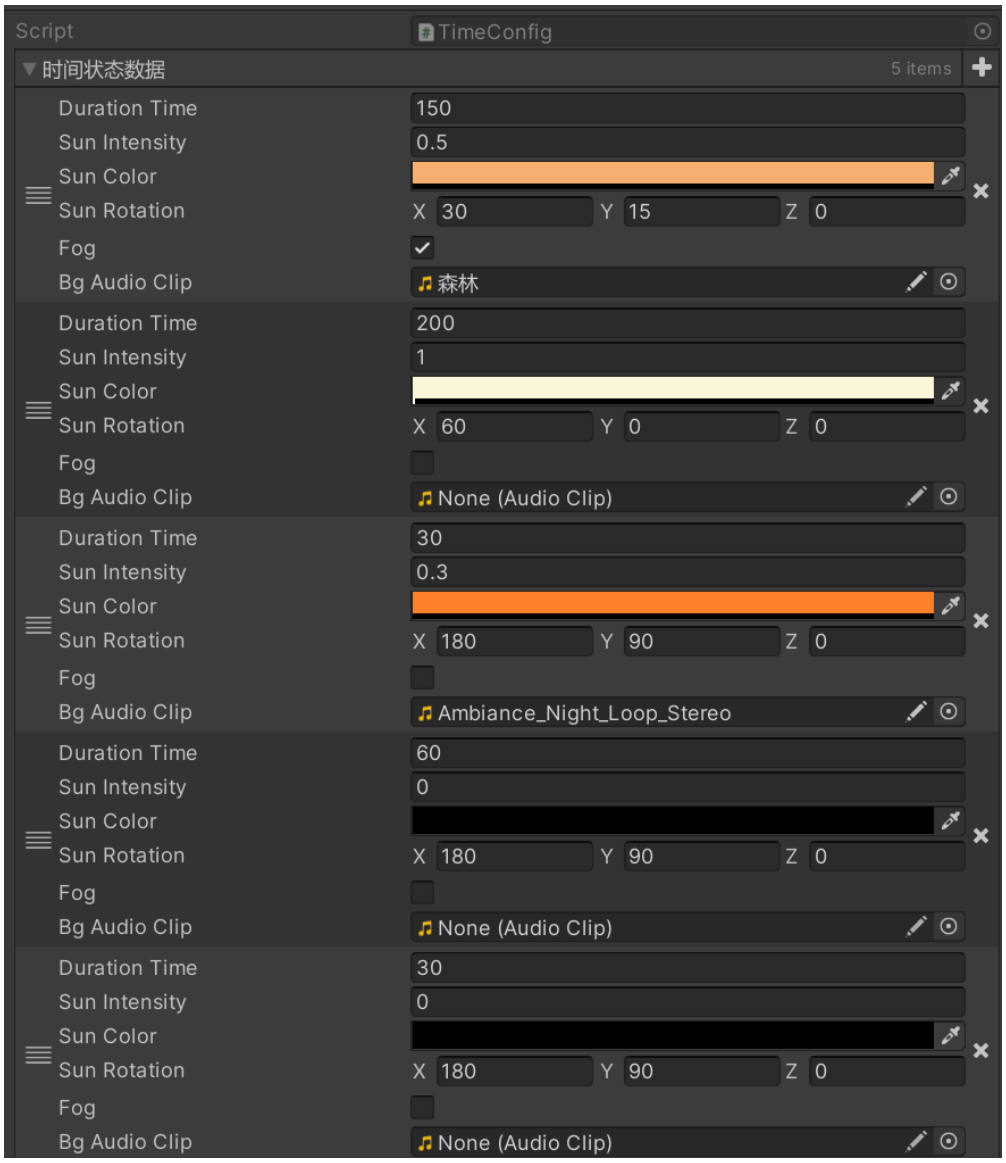
```
public class TimeConfig : ConfigBase
{
    [LabelText("时间状态数据")]
    // 默认index = 0时为白天
    public TimeStateConfig[] timeStateConfig;     // 时间配置
}
```

```
// 配置时间: 亮度/时间/颜色
[Serializable]
public class TimeStateConfig
{
    public float durationTime;          // 持续时间
    public float sunIntensity;          // 阳光强度
    public Color sunColor;              // 阳光颜色
    [OnValueChanged(nameof(SetRotation))]
    public Vector3 sunRotation;         // 阳光角度
    [HideInInspector]
    public Quaternion sunQuaternion;    // 阳光角度-四元数

    public bool fog;                    // 迷雾
    public AudioClip bgAudioClip;       // 背景音乐

    // 当阳光角度发生变化, 需要计算出四元数
    private void SetRotation() {
        sunQuaternion = Quaternion.Euler(sunRotation);
    }

    // 检测并计算下一个时间配置
    public bool CheckAndCalTime(float currTime, TimeStateConfig nextState, out Quaternion rotation, out Color color, out float sunIntensity
        float ratio = 1.0f - (currTime / durationTime);       // 计算当前时间比例
        rotation = Quaternion.Lerp(this.sunQuaternion, nextState.sunQuaternion, ratio);
        color = Color.Lerp(this.sunColor, nextState.sunColor, ratio);
        sunIntensity = UnityEngine.Mathf.Lerp(this.sunIntensity, nextState.sunIntensity, ratio);
        if (fog == true) {
            // 迷雾强度随时间递减, 没有考虑多个时间点都有雾的情况(无过渡状态)
            RenderSettings.fogDensity = 0.1f * (1 - ratio);
        }
        return currTime > 0;
    }
}
```


Script
TimeConfig
时间状态数据
5 items
Duration Time
Sun Intensity
Sun Color
Sun Rotation
Fog
Bg Audio Clip
150
0.5
X 30 Y 15 Z 0
森林
Duration Time
Sun Intensity
Sun Color
Sun Rotation
Fog
Bg Audio Clip
200
1
X 60 Y 0 Z 0
None (Audio Clip)
Duration Time
Sun Intensity
Sun Color
Sun Rotation
Fog
Bg Audio Clip
30
0.3
X 180 Y 90 Z 0
Ambiance_Night_Loop_Stereo
Duration Time
Sun Intensity
Sun Color
Sun Rotation
Fog
Bg Audio Clip
60
0
X 180 Y 90 Z 0
None (Audio Clip)
Duration Time
Sun Intensity
Sun Color
Sun Rotation
Fog
Bg Audio Clip
30
0
X 180 Y 90 Z 0
None (Audio Clip)