1. 编写字符串类，实现字符串相关接口，方便对象到字符串的转换、方便字符串间操作；（框架搭建完成）
2. 编写数学库；（框架搭建完成）
3. [编写文件输入输出层](#编写文件输入输出层)，即 定义配置文件内部编写格式、建立配置文件读写接口，配置文件建议：如xml或ini；（框架搭建完成）
4. [将Direct窗体层使用配置文件输入层进行对接](#配置文件输入层)，使玩家可以自行配置视频输入输出参数（如：分辨率、抗锯齿）；（未开始）
5. 编写人体工程学输入设备输入层，提供相应接口，方便获取键盘鼠标手柄的输入，同时可以自动切换当前输入模式（即：当前的输入设备是 键鼠 还是 手柄）；（未开始）
6. 将输入层使用配置文件输入层进行对接，使得玩家可以持久化储存自己的按键配置；（未开始）
7. 编写纹理类，使得可以从文件获取纹理，并可以将纹理数据提交给GPU（包含法线贴图、高度贴图等）；（未开始）
8. [可选]编写Effect类；（未开始）
9. [可选]编写HLSL Shader文件；（未开始）
10. 编写部分算法（如：空间分割算法）；（未开始）
11. 编写Animation类，完成从Animation文件获取动画并输出的功能；（未开始）
12. 待定

3. 编写文件输入输出层

文件IO系统

Summary

决定使用ini与JSON格式ASCII文件作为配置文件，使用二进制文件保存游戏存档。

编写虚基类 IFileIOBase ，代表文件输入输出基类，

编写虚类 IAsciiFileIOLayer, 代表ASCII文件输入输出层，

编写虚类 IBinaryFileIOLayer，代表二进制文件输入输出层，

编写类JsonCommunicator继承自IAsciiFileIOLayer，包装实现对于Json格式文件的读写，

编写类IniCommunicator继承自IAsciiFileIOLayer，包装实现对于Ini格式文件的读写，

编写类BinaryCommunicator继承自IBinaryFileIOLayer，包装实现对于二进制文件的读写，

其中，二进制文件的读写格式包装为类（或struct）BinaryFormat进行传参。

IFileIOLayer

作为纯虚基类。

编写纯虚函数virtual bool SetFileDirectory(const String& directory) = 0; 传入文件地址，输出该文件是否存在（若不存在则可以认为是第一次启动游戏，则可以例如读取玩家电脑配置进行默认设置，但是无论返回是true还是false，都必须保证此文件此时已经创建）

IAsciiFileIOLayer

编写纯虚函数virtual const String GetAttributeValue(const String& attribute) = 0; 传入属性（属性字符串的格式有待决定）输出属性值字符串（由此得知应当完善String类的转换至int或float的函数）

编写纯虚函数virtual bool SetAttributeValue(const String& attribute, const String& value) = 0; 传入目标属性，设置目标属性的值

BinaryFileIOLayer

编写纯虚函数virtual const BinaryBuffer ReadBinary() = 0;

编写纯虚函数virtual bool WriteBinary(const BinaryBuffer& buffer) = 0;

BinaryBuffer

成员函数待定

BinaryFormat

待定

4. 将Direct窗体层使用配置文件输入输出层进行对接

利用IniCommunicator进行对文件的设置及读取

class Config

public :

Config(const String& fileName);

virtual void SetConfig() = 0;

virtual void GetConfig() = 0;

视频设置分别暂定含有：

分辨率（Resolution）Size2D

刷新率 Int

是否全屏（FullScreen）enum

是否启用抗锯齿 enum

抗锯齿类型（AntiAlias）enum

MSAA

FXAA

抗锯齿等级（AntiAlias Level）int

2x 4x 8x

class VideoSettingConfig : public Config

public:

VideoSettingConfig(const String& fileName);

virtual void SetConfig() override;

virtual void GetConfig() override;

//以分辨率举例

//获取

Size2D GetResolution() const;

//设置

void SetResolution() const;

private:

Size2D mResolution;

Ini文件内：

[ResolutionLevels]

Level1Width=800

Level1Height=600

…

…

Level9Width=1920

Level9Height=1080

[VideoSettings]

Resolution=9 (即代表1920x1080)