请你花几分钟时间

看一下前面写的内容

会对你写书有帮助

以下快捷键（只有这个文档中可以用），可以打印出来，编写过程中可以随时边看边用，以提升编写效率。

|  |  |
| --- | --- |
| Ctrl+1 一级标题  Ctrl+2 二级标题  Ctrl+3 三级标题  Ctrl+4 四级标题  Ctrl+T 插入图形  Ctrl+5 图的格式  Ctrl+6 表文  Ctrl+7 图和表下的图题、表题  Ctrl+8 项目符号  Ctrl+9 提示格式  Ctrl+0 正文  Ctrl+，带底纹的程序格式  Ctrl+. 不带编号的程序格式  Ctrl+/ 带编号的程序格式 | Ctrl+1 一级标题Ctrl+2 二级标题Ctrl+3 三级标题Ctrl+4 四级标题 Ctrl+T 插入图形  Ctrl+5 图的格式  Ctrl+6 表文  Ctrl+7 图和表下的图题、表题  Ctrl+8 项目符号  Ctrl+9 提示格式  Ctrl+0 正文   1. Ctrl+/ 带编号的程序格式 2. Ctrl+. 不带编号的程序格式 |

程序共有三种：第一种是带底纹的，第二种是带方框的，第三种是方框里还有序号的

如果只是书上讲讲，读者看看就行的，就用第一种，即Ctrl+，

1. 如果还需要给读者提供代码的，就用第二种，即Ctrl+.

一般是这样的

关于XXX见代码5-1。

代码5-1 XXXXXX

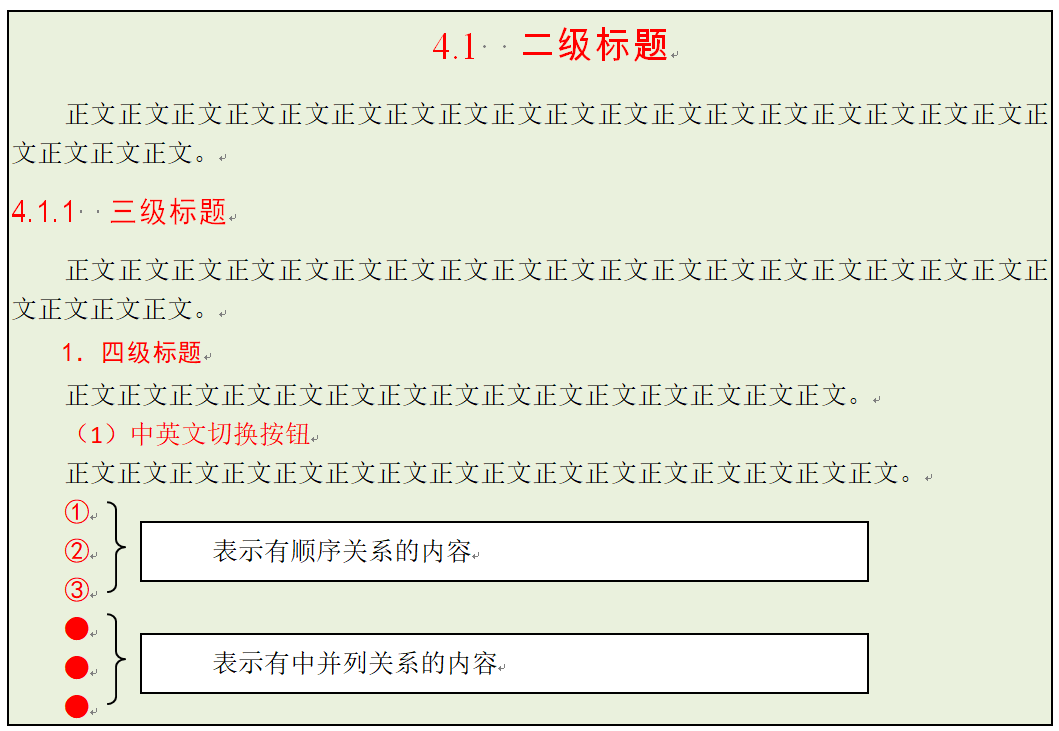
Xxxxx

Xxxxxx

xxx

如果代码，还要讲解，为了后文更方便说是哪一行代码，可以用第三种。

每章的基本层次关系如下：



# <一级标题 Ctrl+1 >

XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX

## 1.1 <二级标题 Ctrl+2>

XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX X<正文 Ctrl+0>

### 1.1.1 <三级标题 Ctrl+3 >

XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX

#### 1. <四级标题 Ctrl+4>

XXXXXXXXXXXX



图1-1 图题Ctrl+7

XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX

表1-1 表题 Ctrl+7

|  |  |
| --- | --- |
| XXXXXXX< Ctrl+6 表文> | XXXXXXX |
| XXXXXXX | XXXXXXX |
| XXXXXXX | XXXXXXX |

XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX

XXXXXXXXXXXXXXXXXXXXXXXXXXX <项目符号 Ctrl+8 >

XXXXXXXXXXXXXXXXXXXXXXXXXX

提示：XXXXXX <提示格式 Ctrl+9 >

Xxxxxxx <程序格式 Ctrl+ , >

**●●学会利用大纲视图**

建议，

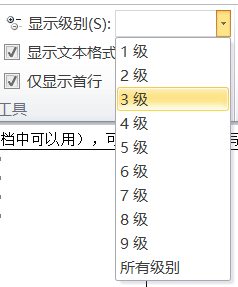
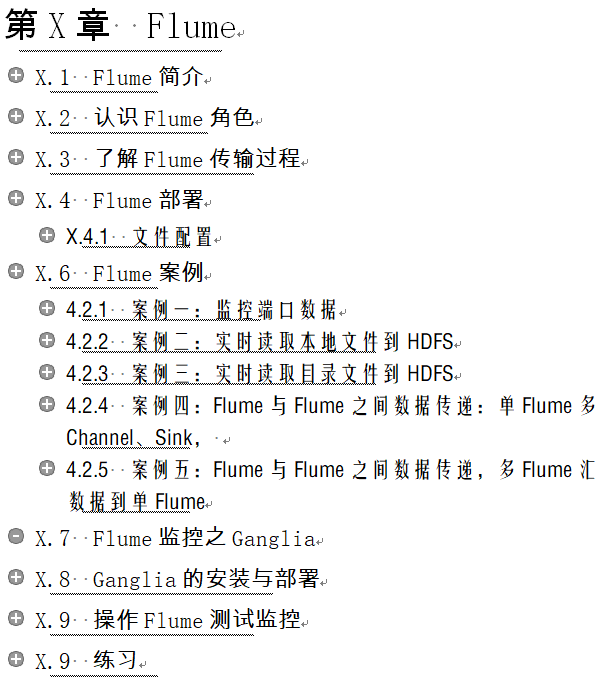
（1）你先把 一级标题 二级标题 三级标题 四级标题的快捷键弄熟。

（2）学会使用大纲视图来调整层次结构

具体方法是单击WORD右下角的大纲视图按钮，如下左图所示。

然后在上方选择3级视图，如下中图所示。

然后，你可以看到下右图所示。

**●●注意事项**

（1）定义的快捷键，只能在这个文档中使用。所以，正式写的时侯，把这个文档复制多份，每一份只留一章，在其中写。

（2）不要所有的文稿都在一个文档里。

**●●样章的目的**

请用下面这个文档，开始写作。

我建议，你把 节写一部分

写书，不是一个一个字，力求完美这么写。可以分为几道流程：

第一遍，快速地把一章的内容铺出来，不要纠结于一段文字，一个图，一个细节，该跳的就先跳过了，不要影响了流畅的结构性思考。

第二遍，才去想一句该怎么完美，去扣细节。

第三遍…………

通过这部分写作

一，熟练使用快捷键

二，了解图书的层次结构

三，我们交互几次，您可以知道图书的一些基本要求。

**下一页开始正式写了！**

第 1 篇 准备

# 第1章 了解Web 3D和搭建开发环境

## 1.1 了解3D编程

XXXXXX

#### 1.一级小标题

#### 2.二级小标题

## 1.2 了解Web 3D的历史

## 1.3 了解WebGL的基础知识

### 1.3.1 渲染管线

### 1.3.2 坐标系变换

## 1.4 准备预备知识

## 1.5 搭建开发环境

### 1.5.1 了解Reason版本和开发环境

### 1.5.2 下载及安装VSCode

### 1.5.3 在VSCode中搭建Reason的开发环境

## 1.6 编写第一个Reason程序

# 第2章 搭建测试环境

## 2.1 了解自动化测试

### 2.1.1 单元测试

### 2.1.2 集成测试

### 2.1.3 端对端测试

## 2.2 了解运行测试

### 2.2.1 通过log测试

### 2.2.2 通过WebGL-Inspector测试WebGL

### 2.2.3 通过Spector测试WebGL

## 2.3 了解性能测试

### 2.3.1 使用Profile测试

### 2.3.2 使用Timeline测试

### 2.3.3 使用Console.profile和Performance测试

### 2.3.4 使用Memory测试

第 2 篇 基础

本篇主要介绍了基础知识；

# 第3章 函数式编程与函数反应式编程

## 3.1 函数式编程

### 3.1.1 函数式编程思想

### 3.1.2 了解函数式编程的知识点

### 3.1.3 函数式编程示例

## 3.2 函数反应式编程

### 3.2.1 函数反应式编程思想

### 3.2.2 了解Most库

### 3.2.3 函数反应式编程示例

# 第4章 Reason基础

(介绍Reason特色，以及为什么要用Reason)

## 4.1 Reason语法

## 4.2 Reason示例

# 第5章 Reason-React和Redux基础

## 5.1 Reason-React基础知识

## 5.2 Redux基础知识

## 5.3 Reason-React 和 Redux综合应用示例

第 3 篇 搭建雏形

这一篇主要搭建引擎和编辑器的雏形。

# 第6章 编写一个最小的3D程序“绘制三角形”

## 6.1 需求分析

## 6.2 总体设计

## 6.3 具体实现

### 6.3.1 给出模块代码

### 6.3.2 给出每个模块的具体代码，具体分析

# 第7章 从程序中提炼引擎和编辑器

## 7.1 引擎需求分析

## 7.2 编辑器需求分析

## 7.3 提炼引擎

### 7.3.1 设计边界

### 7.3.2 提炼数据

### 7.3.3 分析用户逻辑

### 7.3.4 识别领域模型

### 7.3.5 使用引擎

## 7.4 提炼编辑器

### 7.4.1 设计边界

### 7.4.2 提炼数据

### 7.4.3 引入引擎

### 7.4.4 使用编辑器

## 7.5 优化引擎性能

### 7.5.1 测试性能

### 7.5.2 优化“初始化”逻辑

### 7.5.3 优化“主循环”逻辑

### 7.5.4 分析最终优化结果

# 第8章 改进引擎

## 8.1 提炼“GameObject+组件”架构

### 8.1.1 分析现有架构

### 8.1.2 设计

### 8.1.3 实现组件

### 8.1.4 修改GameObject

### 8.1.5 修改Render

### 8.1.6 实例：扩展组件

### 8.1.7 实现新需求

## 8.2 提炼Data Oriented架构

### 8.2.1 分析现有架构

### 8.2.2 设计

### 8.2.3 实现

### 8.2.4 实例：扩展组件

### 8.2.5 实现新需求

## 8.3 改进Shader

### 8.3.1 分析现有架构

### 8.3.2 设计

### 8.3.3 实现

### 8.3.4 实例：扩展Shader

### 8.3.5 实现新需求

## 8.4 提炼渲染队列

### 8.4.1 分析现有架构

### 8.4.2 设计

### 8.4.3 实现

### 8.4.4 实现新需求

## 8.5 支持各种应用场景

### 8.5.1 分析现有架构

### 8.5.2 设计

### 8.5.3 实现

### 8.5.4 实例：支持移动端

### 8.5.5 实现新需求

## 8.6 支持多线程

### 8.6.1 分析现有架构

### 8.6.2 设计

### 8.6.3 实现

### 8.6.4 实例：增加线程

## 8.7 提炼子State

## 8.8 使用函数式反应式编程实现主循环

## 8.9 契约检查

## 8.10 错误处理

# 第9章 改进编辑器

## 9.1 更新引擎

## 9.2 分析现有架构

## 9.3 重构文件夹

## 9.4 契约检查

## 9.5 错误处理

第 4 篇 增加核心功能

# 第10章 增加Redo/Undo功能

# 第11章 增加“场景管理”功能

## 11.1 需求分析

## 11.2 查看场景中的所有GameObject

## 11.3 操作GameObject

# 第12章 增加Inspector功能

## 12.1 需求分析

## 12.2 显示选中的GameObject的所有组件

## 12.3 操作组件

# 第13章 增加相机功能

## 13.1 需求分析

## 13.2 给引擎增加相机组件

## 13.3 更新编辑器

# 第14章 给编辑器增加“运行”/“停止”功能

## 14.1 需求分析

## 14.2 初步设计

## 14.3 重构：提炼新设计

## 14.4 具体实现

# 第15章 给编辑器增加“资产管理”功能

## 15.1 需求分析

## 15.2 增加Material资产

## 15.3 重构：使用Recursive Type

## 15.4 具体实现

### 15.4.1 实现Asset Tree

### 15.4.2 实现Asset Children

### 15.4.3 实现Asset Inspector

# 第16章 支持光照

## 16.1 需求分析

## 16.2 更新引擎

### 16.2.1 增加DirectionLight组件

### 16.2.2 增加LightMaterial组件

## 16.3 更新编辑器

### 16.3.1 更新Inspector

### 16.3.2 更新Asset

### 16.3.3 更新Icon Gizmo

# 第17章 支持纹理

## 17.1 需求分析

## 17.2 更新引擎

### 17.2.1 增加BasicSourceTexture组件

### 17.2.2 给LightMaterial组件增加Diffuse Map

## 17.3 更新编辑器

### 17.3.1 更新Inspector

### 17.3.2 更新Asset

# 第18章 增加“导入”／“导出”功能

## 18.1 需求分析

## 18.2 更新引擎

### 18.2.1 增加Scene Graph文件（.wdb）

（加更多标题，不能只有一个标题）

## 18.3 给编辑器增加导出功能

### 18.3.1 增加“导出场景”功能

### 18.3.1 增加“导出包”功能

## 18.4 给编辑器增加导入功能

### 18.4.1 增加“导入模型资产”功能

### 18.4.2 增加“导入包”功能

# 第19章 给编辑器增加“发布本地包”功能

第 5 篇 增加更多功能

# 第20章 给编辑器增加“控制台”功能

# 第21章 增强编辑器的Scene View的渲染

## 21.1 总体需求分析

## 21.2 给编辑器增加“网格平面”功能

## 21.3 给编辑器增加“Icon Gizmo”功能

### 21.3.1 需求分析

### 21.3.2 实现IMGUI

### 21.3.3 实现相机Gizmo

# 第22章 增强相机功能

## 22.1 总体需求分析

## 22.2 支持事件

### 22.2.1 需求分析

### 22.2.2 让引擎支持事件

### 22.2.3 让编辑器支持事件

## 22.3 增加FlyCameraController组件

### 22.3.1 需求分析

### 22.3.2 给引擎增加FlyCameraController组件

### 22.3.3 让编辑器支持FlyCameraController组件

## 22.4 让编辑器的Scene View的相机使用FlyCameraController组件

# 第23章 让编辑器支持PWA

第 6 篇 应用

这一篇主要讲，引擎和编辑器的应用实例。

# 第24章 使用编辑器制作3D静态场景

## 24.1 需求分析

## 24.2 分析如何实现

## 24.3 准备美术素材

## 24.4 具体实现

## 24.5 总结

# 第25章 使用引擎制作3D动态场景

## 25.1 需求分析

## 25.2 分析如何实现

## 25.3 准备美术素材

## 25.4 具体实现

## 25.5 总结