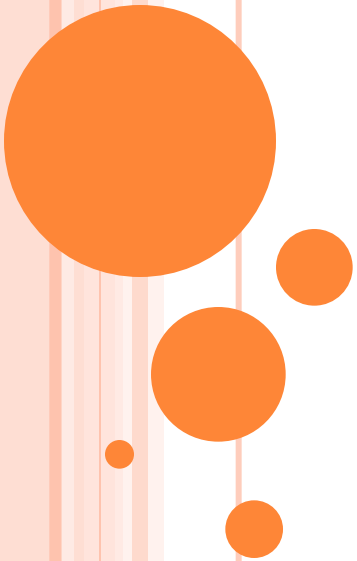
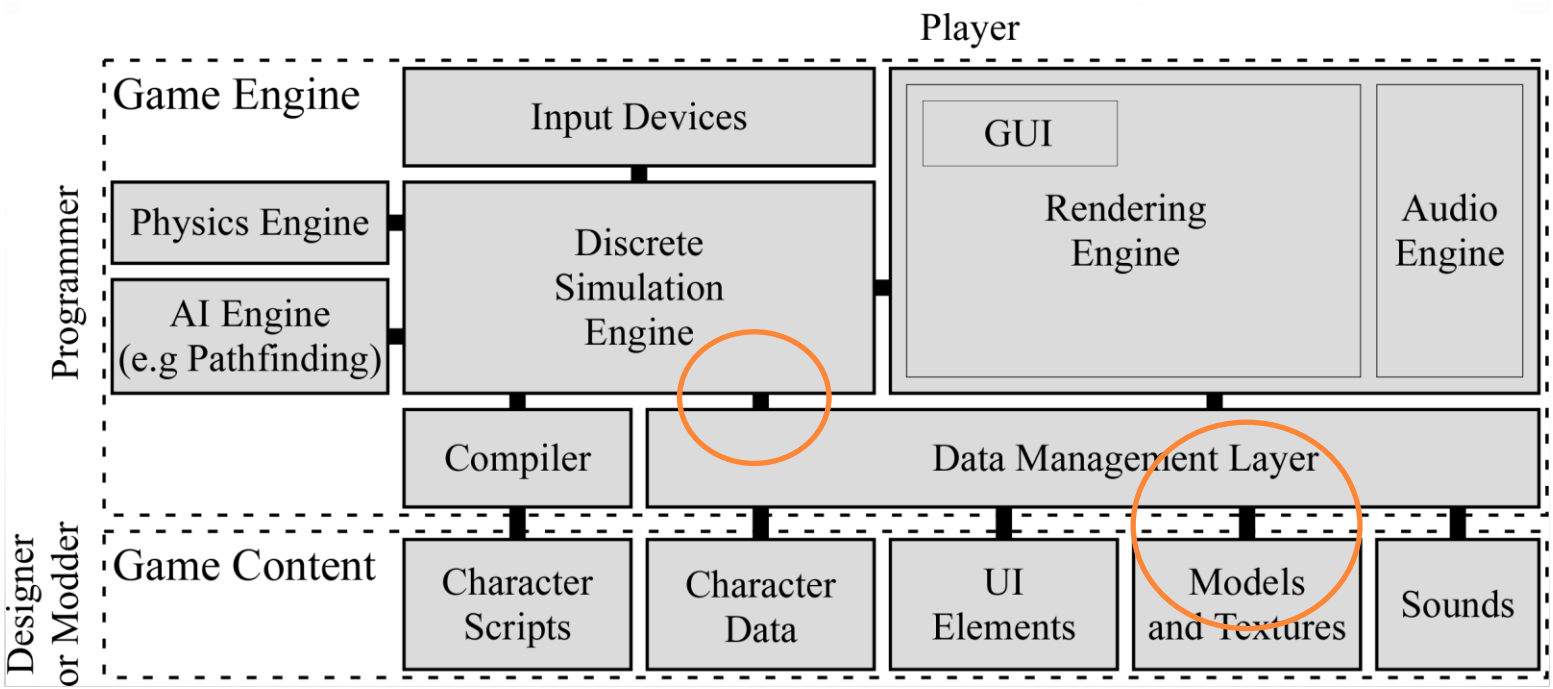


Introduction to Computer 3D Game Development

Particle System

潘茂林，panml@mail.sysu.edu.cn
中山大学·软件学院

游戏引擎架构


Player
Game Engine
Input Devices
GUI
Rendering Engine
Audio Engine
Physics Engine
Discrete Simulation Engine
AI Engine (e.g Pathfinding)
Programmer
Compiler
Data Management Layer
Designer or Modder
Game Content
Character Scripts
Character Data
UI Elements
Models and Textures
Sounds

目录

- 粒子系统（Particle System）
  - 粒子系统简介
  - 部件与使用
  - 制作粒子系统
  - 粒子系统学习方法
- 课堂编程练习
  - 粒子海洋
- 面向对象的编程思考
  - 高内聚、低耦合
  - 协程技术

粒子系统（Particle System）（1）简介

- 粒子系统
  - 为了节省计算资源，用一些简单的面(粒子)来模拟一些特定的随机性流动的现象，产生 —— 例如火、爆炸、烟、水流、火花、落叶、云、雾、雪、尘、流星尾迹或者象发光轨迹这样的视觉效果。
  - 粒子部件的原理
    - 粒子：一个基本 2D 显示单元
    - 粒子工厂：循环使用的粒子对象数量
    - 运动对象：各种控制粒子变化的对象
    - 粒子渲染器：材料（纹理＋shader+元数据）
    - 约束空间：控制粒子运动的范围
    - 粒子播放管理器：管理播放初始化属性

粒子系统（Particle System）
（2）部件与使用

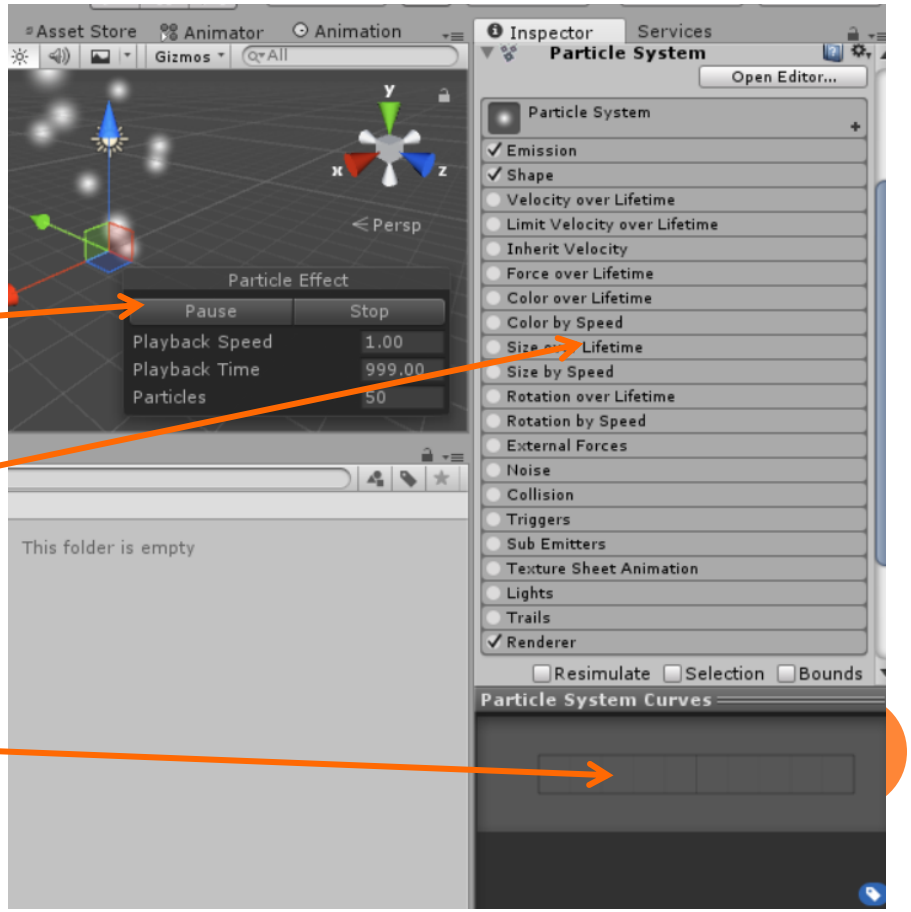
- 创建粒子对象
  - GameObject -> Particle System
  - 从预制中创建

- 基本粒子对象

编辑播放器

粒子系统部件：
子部件

曲线编辑器



粒子系统（Particle System）
（2）部件与使用

- 导入标准资源
  - Assets → Import → Particle systems

- 粒子基本使用
  - Renderer
    - 修改材料，观看效果
  - Shape
    - 修改 shape 和 emit from 观看效果
  - Emission
    - 修改 rate 观看效果
  - Particle System
    - 修改 Max particle 数量，例如 10，观看效果
    - 修改初始速度

粒子系统（Particle System）
（3）制作粒子系统

- 拖入粒子资源
  - 下载 Fx Explosion Pack 解压，将目录拖入项目资源
  - 从预制中创建游戏对象 Explosen1
  - 运行！ 出错
  - Assets → run API udates…
  - OK!

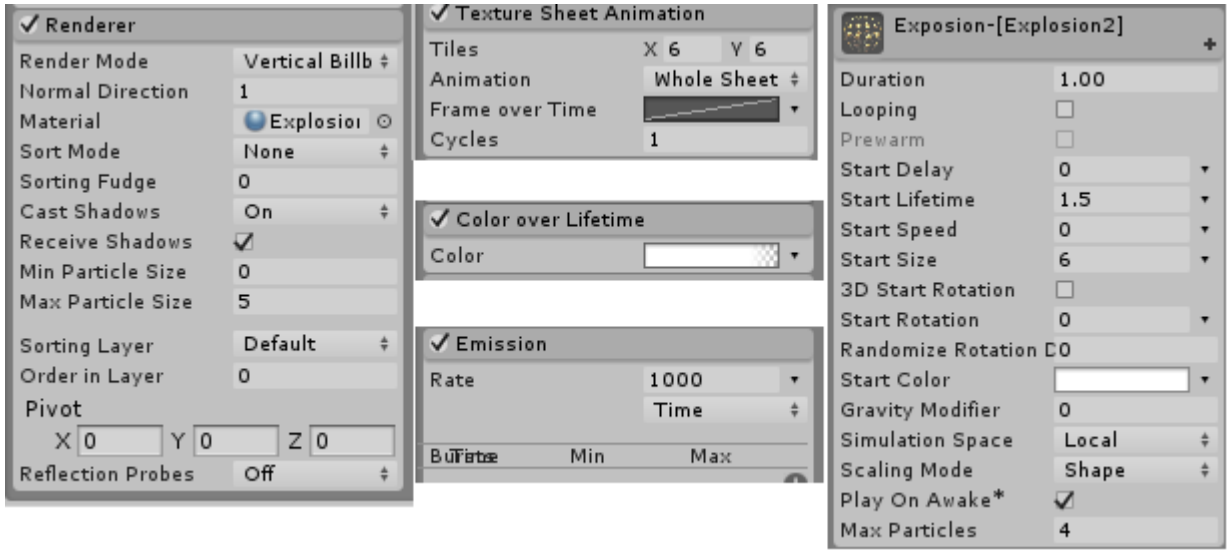
- 研究预制结构
  - 预制包含几个粒子系统子对象？
  - 各个子对象的效果（多次点解编辑器中 simulate）？
  - 将每个预制拖入对象层次树，观看每一种爆炸的效果

粒子系统（Particle System）
实验一，仿制 Explosion3 预制

- 准备
  - 将 Explosion3 预制放置到拖入对象层次树，根节点有一段代码，卸载它（这代码简单，1.5 秒后销毁自己）。
  - 展开 Explosion3，它由 Exposion-[Explosion2]、Ring、Decal 三个粒子系统与一个 Point light 构成。
  - 将 Ring、Decal、Point light 设置为不活动的。将 Exposion 的 loop 属性设为 True，以便于观察效果。
  - 选择 Exposion-[Explosion2]，你就看到 #Scene 面板右下有一个 Particle Effect， 按 stop -> simulate 播放它的效果！

粒子系统（Particle System）
实验一，仿制 Exploson3 预制

- 仿制过程
  - 从底向上的顺序配置有助于了解效果，如图顺序：



- 配置 Exploson 的效果
  - 新建 Empty 对象，并建立一个新 particle system 的子对象（ex copy），配置如图！

粒子系统（Particle System）
实验一，仿制 Explosion3 预制

- 查手册，了解各个属性的含义（课后）
- 如法构建其他子对象，最后挂上代码资源，创建自己与预制。（效果一样吗？）

- 关于 shader
  - 粒子系统编程的难点就是 shader 了。 高大上点就是 CG，粗俗点就是把图片贴到物体上的程序，也称 UV 贴图

粒子系统（Particle System）
（3）粒子系统学习方法

- 粒子系统配置
  - 粒子是配置复杂的系统，没有简单学习方法
  - 注意收集网上粒子效果资源
  - 仿制有利于加深对基本配置的理解

- 粒子预制的组合
  - 观察系统提供的粒子预制的效果
  - 按需制作一些效果（仅需将你喜欢的粒子效果 copy 然后 paste 到你的对象下，作为子对象）。
  - 例如:
  - 礼花效果： Explosion - Shower; Flare - smoke,sparks
  - 英雄放大招时 Explosion - Shockwave， Fx Explosion -ring

粒子系统进阶
实验二：神奇的粒子海洋

- Unity制作神奇的粒子海洋
  - 教程地址 http://www.manew.com/thread-47123-1-1.html

- 源代码地址：
  https://github.com/RafalWilinski/Particle-Sea

面向对象设计思考
（1）高内聚、低耦合

- 为什么？
  - 高内聚低耦合，是软件工程中的概念，是判断设计好坏的标准。
- 高内聚
  - 内聚就是一个模块内各个元素彼此结合的紧密程度。
  - 高内聚模块是内部元素彼此联系紧密，代码相关性强，且对外呈现负责单一任务，即单一责任原则。
- 低耦合
  - 耦合指对象或模块之间相互依赖。
  - 低耦合指对象或模块之间不存在循环依赖，即每个模块可以独立使用，独立打包，独立修改。

面向对象设计思考
（1）高内聚、低耦合

- 游戏设计要求
  - 如何判定一个游戏的设计质量
  - 一个游戏如何能按内部职能划分成若干部分（模块）
    - 每个模块对外呈现单一接口或类
    - 模块之间成层次或树状依赖关系
    - 模块内部修改，影响范围小，可控
- 好处
  - 分解软件系统，降低软件系统开发的复杂性
  - 易于理解、易于维护、易于扩展

面向对象设计思考
（1）包装游戏对象

- 问题：
  - 在项目中添加两个第三方角色控制器（Ethan），你会发现它们不能被独立控制
- 分析
  - 第三方角色控制器有两个代码：
  - ThirdPersonUserControl.cs（人机交互）
  - ThirdPersonCharacter.cs（角色配置与动作管理）
- 设计要求
  - 创建接口 IPersonUserControl.cs
  - 建立新代码PersonUserControl.cs，使得SceneController 拥有每个角色的控制权，最终用户可以通过 UserGUI 按自己的逻辑控制 Ethan
  - 创建独立预制和相关资源，使该对象为独立的模块。即拖入项目即用，不需要依赖其他资源。

面向对象设计思考
（2）初识协程技术

- 见以往课件

课程小结

- 粒子系统（Particle System）
  - 粒子系统常用部件
  - 粒子系统的设置
  - 粒子系统的组合
- 面向对象的编程思考
  - 高内聚、低耦合
  - 协程技术

作业（Lab 9）

- 作业
  - 参考 http://i-remember.fr/en ， 使用粒子制作类似效果
  - 作业参考：
    - http://blog.csdn.net/simba_scorpio/article/details/51251126
    - http://blog.csdn.net/gunnerczh/article/details/51291348
    - https://16sixteen.github.io/unity3d/unity3d_particle_ring