Computer Science and Application 计算机科学与应用, 2022, 12(11), 2460-2471
Published Online November 2022 in Hans. http://www.hanspub.org/journal/csa
https://doi.org/10.12677/csa.2022.1211252

Hans汉斯

基于Unity3D的RPG类游戏关键技术实现

刘端烨，陈立哲

北方工业大学，北京

收稿日期：2022年10月11日；录用日期：2022年11月9日；发布日期：2022年11月16日

摘　要

RPG类游戏可以让玩家通过角色扮演身临其境地参与到游戏剧情当中，具有极强的趣味性，是当今游戏市场的主流游戏类型之一，深受广大玩家的喜爱。Unity3D游戏引擎由于其跨平台和易用性，越来越受到游戏开发者们的青睐。本文基于Untiy3D游戏引擎，以实现RPG类游戏的关键功能为目标，将基于有限状态机的角色控制、背包商店系统、剧情加载系统等关键技术做了系统的分析和实现，为游戏开发者提供参考。

关键词

RPG，Unity3D，角色控制，有限状态机，背包系统，剧情加载

The Key Technology Achieve about RPG Games Based on Unity3D

Duanye Liu, Lizhe Chen

North China University of Technology, Beijing

Received: Oct. 11th, 2022; accepted: Nov. 9th, 2022; published: Nov. 16th, 2022

Abstract

RPG games allow players to immerse themselves in the plot of the game through role-playing, which is highly playfulness and is one of the mainstream game types in today's game market, and is loved by the majority of players. Unity3D game engine is increasingly popular among game developers because of its cross-platform nature and easy to use. This article, based on Untiy3D game engine with the goal of implementing key features of RPG games, systematically analyzes and implements key technologies such as character control of finite-state machine, backpack storage system, plot loading system, etc. to provide a reference for game developers.

文章引用: 刘端烨, 陈立哲. 基于Unity3D的RPG类游戏关键技术实现[J]. 计算机科学与应用, 2022, 12(11): 2460-2471.
DOI: 10.12677/csa.2022.1211252

Keywords

RPG, Unity3D, Character Control, Finite-State Machine, Bagpack System, Plot Loading

Copyright © 2022 by author(s) and Hans Publishers Inc.
This work is licensed under the Creative Commons Attribution International License (CC BY 4.0).
http://creativecommons.org/licenses/by/4.0/
Open Access

1. RPG 类游戏的历史及现状

RPG 游戏，全称 Role-Playing Game，意为角色扮演类游戏，指玩家在游戏中扮演某一游戏角色并且在将要展示的世界中游玩探索。

早期的电子 RPG 在上世纪 70 年代诞生，1974 年，《龙与地下城》的出现对 RPG 游戏产生了深远影响。RPG 类游戏总结起来具备三大特性，“故事性”、“成长性”、“交互性”[1]。现如今，流行的 RPG 类游戏有《原神》《塞尔达传说》等，其优势在于玩家通过角色扮演，可以身临其境体验到游戏剧情当中，将游戏中的剧情当作自己身上发生的故事，既满足了玩家对于参与感的追求，同时，游戏的真实性也大大提升。并且 RPG 还能与冒险游戏、模拟游戏、射击游戏等结合，形成新的游戏模式，从而提高游戏的趣味性[2]。这就是 RPG 游戏的优势和引人之处。

从 RPG 类游戏产生至今，已经有无数优秀的作品出现，且仍有不少 RPG 类游戏作品是依靠于游戏引擎中提供的相关接口功能以及先前相关课题提供的思路完成游戏开发[3]。相关课题中也有提到控制角色操作等相关操作，其课题提供的实现方案有一定可行性，但会浪费一些程序运行时间[4]。基于此原因，本课题将给出 RPG 类游戏关键技术优化以及实现功能。

2. Unity3D 的应用现状

Unity3D 是目前主流的游戏开发引擎，并且实现了真正的跨平台，且 U3D 开发的游戏可大可小，从简单小游戏到高成本、高质量、高体量的游戏(即 3A 游戏)都可以胜任。在跨平台方面的优点，使得 Unity 3D 同样也是目前游戏行业里使用最广泛的游戏引擎。不少精良的游戏都是出自 U3D，例如《炉石传说》《神庙逃亡》《奥里与迷失森林》等，其中，米哈游开发的《崩坏学园》《原神》，鹰角网络开发的《明日方舟》，X.D. Network Inc 开发的《ICEY》等都是利用 U3D 开发的 RPG 类型游戏。

现有的 Unity3D 游戏引擎中内置的人物角色控制系统用法单一，并且利用 C#脚本实现后，需要逐帧进行检测，对于动作较多的角色控制，会使得程序的时间复杂度大大增加。本文中利用有限状态机可以很好地解决此问题，降低时间复杂度，同时，脚本代码量也会减小，程序的可读性和鲁棒性也会大大增加[5]。

背包商店系统在 Unity3D 游戏引擎中没有给出背包商店系统的专门实现方法，本文通过利用 Prefab(预制体)和 ScriptableObject 存储方法来实现背包商店系统之间的联系与交互。

剧情加载和文本读取，在 Unity3D 中大多实现都是基于在 UI 组件的文本框中预先定义好，结合协程进行调用，此类方法不具有灵活性。且 RPG 类游戏以多选择触发为特色，玩家选择不同的对话结果后触发的事件也将不同，在文本框中预先定义好文本剧情是相当繁琐且冗长的实现方式。本文通过制定一个剧情加载文本规范，给出的剧情读取解决方案，并且优化增加程序灵活性的同时保证空间复杂度不被大量文本占用。

文章的组织结构：本文根据 RPG 类游戏制作中应用到的关键技术，按如下结构组织后文：

第 3 章基于状态机的角色状态，第 4 章背包商店系统，第 5 章剧情加载系统。

3. 基于状态机的角色控制

3.1. Unity3D 有限状态机工作原理

有限状态机(英语：Finite-State Machine，缩写：FSM)，一个可以枚举出有限个状态，并且这些状态在特定条件下能够实现来回切换。为了使程序更加具有灵活性，新加入的状态只需要继承基类即可添加实现。

如图 1 状态机类类图所示，FSMState 包含状态列表，每一状态的 ID，同时也包含了当前状态的转换条件，以及调用此状态时状态开始的方法，状态持续过程中的方法以及状态结束时的方法。为状态机在 U3D 更好的实现提供了接口。

为了让 FSMState 状态类与 MonoBehaviour 类之间进行更好的交互，新建一个 FSMSystem 管理状态类来进行对 FSMState 类的管理，并且与 MonoBehaviour 类中的 Start 函数和 Update 函数相结合，从而实现对一个状态机的管理。其类图如图 2 所示。

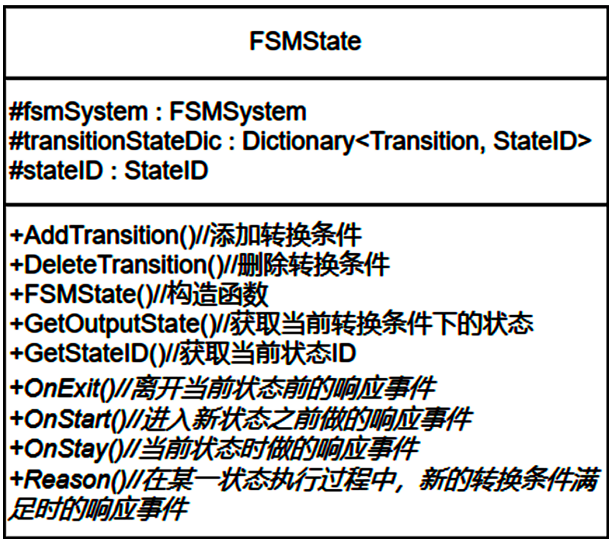

Figure 1. FSMState state class diagram
图 1. FSMState 状态类图

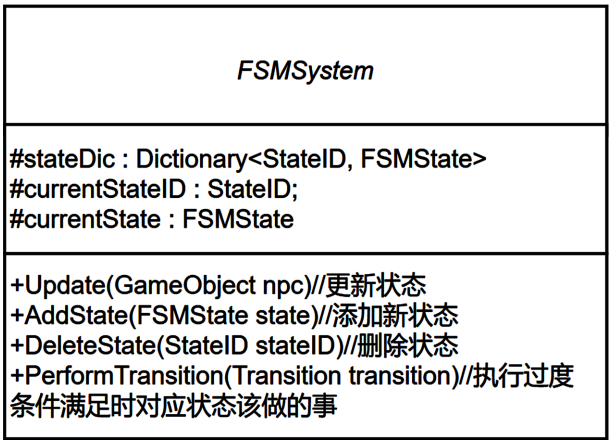

Figure 2. FSMSystem management state class diagram
图 2. FSMSystem 管理状态类图

状态机能够分离逻辑代码，从而提高代码的可维护性。在游戏中控制一个角色的移动、攻击等属性时，将代码放到一个脚本中不可取，需要通过一定的方法将他们划分成多个脚本或多个部分，依次来实现。而状态机就可以很好地将复杂的代码划分成一个个小部分进行实现，提高代码的可维护性和重要性。当人物拥有多个状态时，利用有限状态机更有利于代码的维护和整理，可读性更强，更加有条理性。使用状态机后，角色的动作状态切换时，只需检测触发状态的方法，符合条件时才会进行一次动作的更新。相比之下，单纯利用 C#脚本以及 Unity3D 中提供的函数组件实现各个动作功能切换，将所有状态写在同一个角色运动类中，会导致 unity 每帧进行更新时会遍历大量冗长的代码，使得代码可读性较差并且时间复杂度较高。而使用有限状态机大大降低了程序的时间复杂度，同时，程序也更加简洁，具有鲁棒性。

3.2. 状态机在 Unity3D 中的实现方式

在 Unity 中实现状态机，需要与 Unity3D 中的动画机相互关联，状态机根据状态与动画机中动画的映射关系实现玩家或 NPC 角色的动画控制。创建一个 AI 类脚本挂载到需要检测动画机状态的物体上，通过 AI 类脚本进行调用状态机管理状态类，从而获取有限状态机中所有的状态，以及相关需要检测的资产信息列表。同时，每种状态都要创建一个脚本继承 FSMState 状态类，如图 3 所示为状态控制流程图，图 4 为 AI 类脚本挂载在 inspector 窗口中公开变量信息。

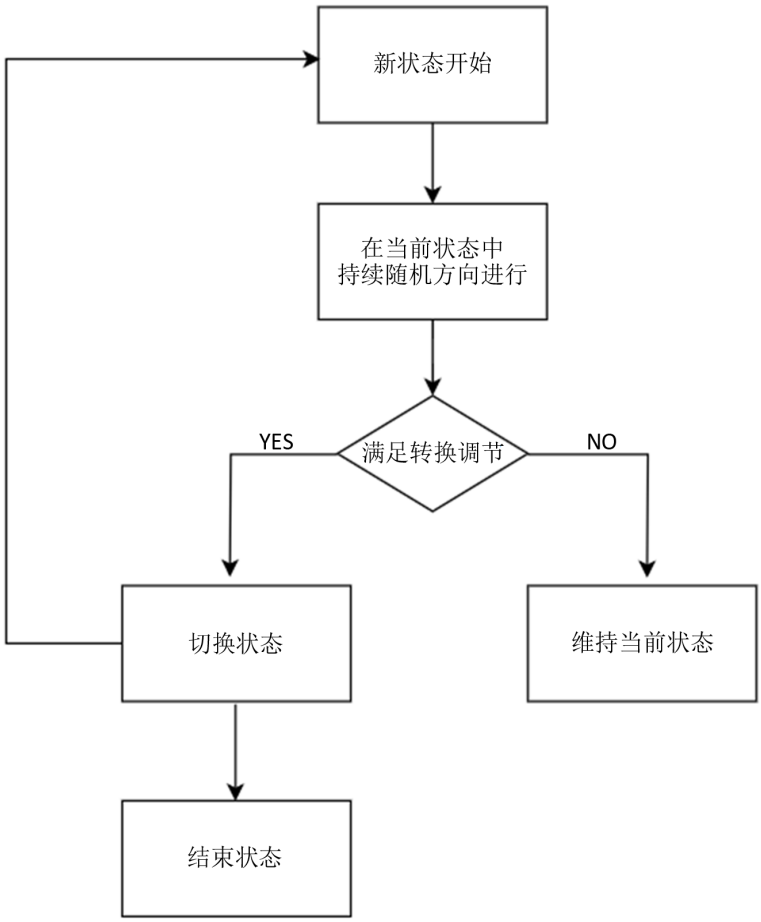

Figure 3. State state control flowchart
图 3. State 状态控制流程图

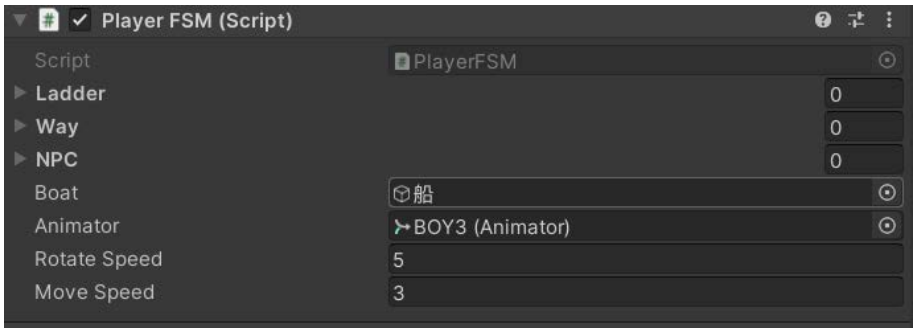

Figure 4. AI script information
图 4. AI 类脚本信息

对于每个 AI 都有不同的状态，如：行走状态、看向 NPC、跑步状态、跟随状态、跳跃状态、游泳状态等等，在转换条件方法中编写状态转换的条件代码，同时将各个状态声明到 FSMState 类脚本中，在 AI 类中实现条件转换。在 Unity3D 中创建一个 Animation 动画机，将 AI 动画导入动画机中，同时，在 FSMState 转换条件的方法中设置一个参数，与 U3D 的动画机相关联，设置一个接口，从而实现基于状态机的 AI 动画在特定条件下的相互转换。图 5 为角色玩家的动画机，其中包含行走、跑步、跳跃、游泳、攀爬、站立状态。图 6 为动画机与状态机之间转换条件参数。

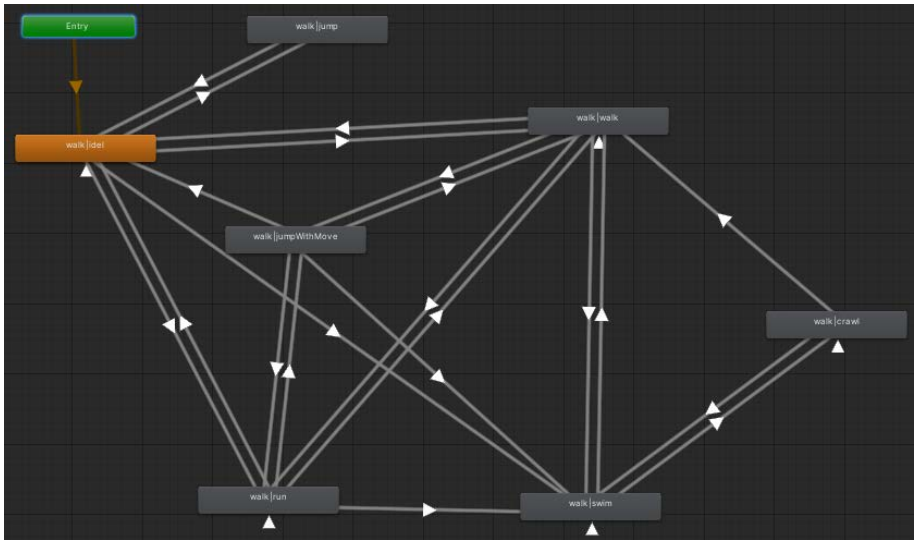

Figure 5. Unity3D animator
图 5. U3D 动画机

Figure 6. State transition condition
图 6. 状态转换条件

4. 背包商店系统

4.1. 背包，商店系统在 RPG 游戏中的应用及设计

在 RPG 类游戏中，背包和商店系统是必不可少的，角色在进行任务剧情交互过程中，大世界自由探索过程中，武器进行装备以及强化，恢复药水等，都要涉及到背包和商店系统。本章通过 C#脚本结合 ScriptableObject 来实现背包与商店信息实时读取。通过创建一个管理类，将 UI 页面的信息显示，数据库信息获取，商店背包中物体选中显示信息、玩家信息、物品信息等相互结合起来。

ScriptableObject 是一个可独立于类实例来保存大量数据的数据容器，其主要用例是通过避免重复值来减少项目的内存使用量。图 7 为商店背包系统数据库 E-R 图。

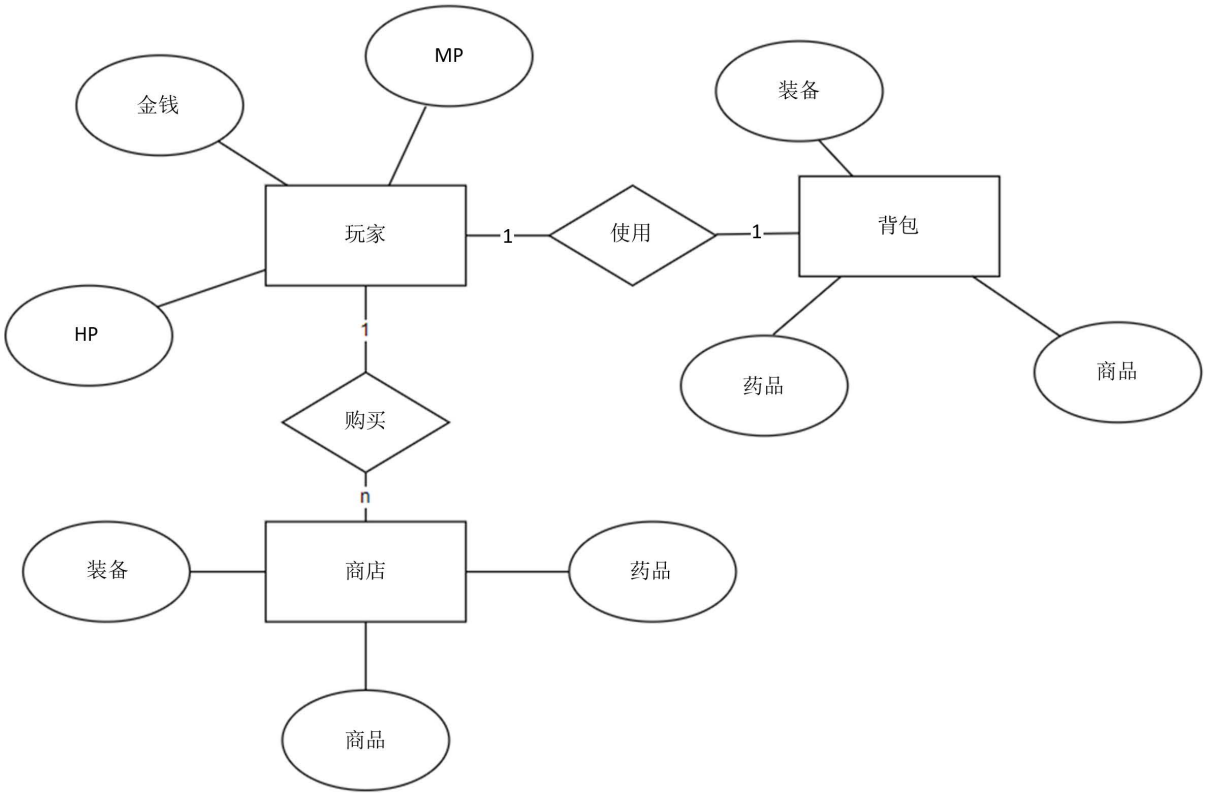

Figure 7. E-R diagram of backpack store system
图 7. 背包商店系统 E-R 图

4.2. 背包，商店系统的框架实现

在 Unity3D 中创建脚本 StoreInventory 和 BagInventory 管理类，进行管理背包和商店的相关信息，与设计好的 UI 页面布局相关联(如图 8、图 9 所示)。为了更好地展现出效果，将类中变量设为公有。在程序中建议设为私有以保护程序信息。

在 Inventory 管理类脚本中，利用 GetComponent<>方法获取 UI 的修改方式，进行信息实时更新。图 10 为 StoreInventory 和 BagInventory 类的相关类图。

UI 的预制体也创建赋予一个命名为 PrefebSlot 的脚本用来实现玩家与数据库信息读取以及交互功能，如：点击后显示描述文字，显示持有数量，模型展示等。如图 11 所示为脚本变量信息，在此脚本中添加 Button 按钮响应事件控制其他组件可视性，从而达到点击后显示对应商品以及背包信息的功能。

Figure 8. UI design page and GameObject list of Backpack and store
图 8. 背包和商店 UI 设计页面以及 GameObject 列表

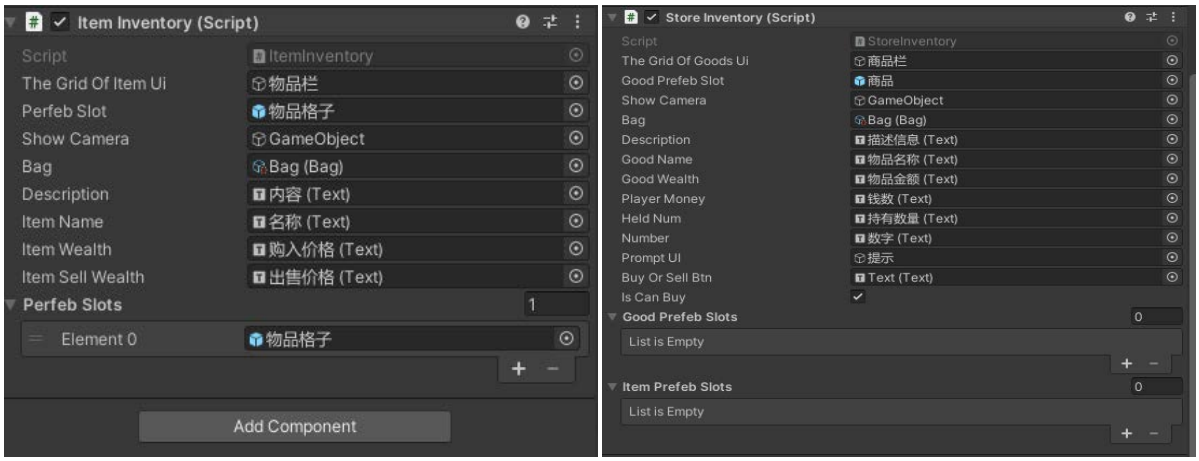

Figure 9. The backpack and store management system scripts expose variables and corresponding location of the UI
图 9. 背包和商店管理系统脚本公开变量以及 UI 对应位置

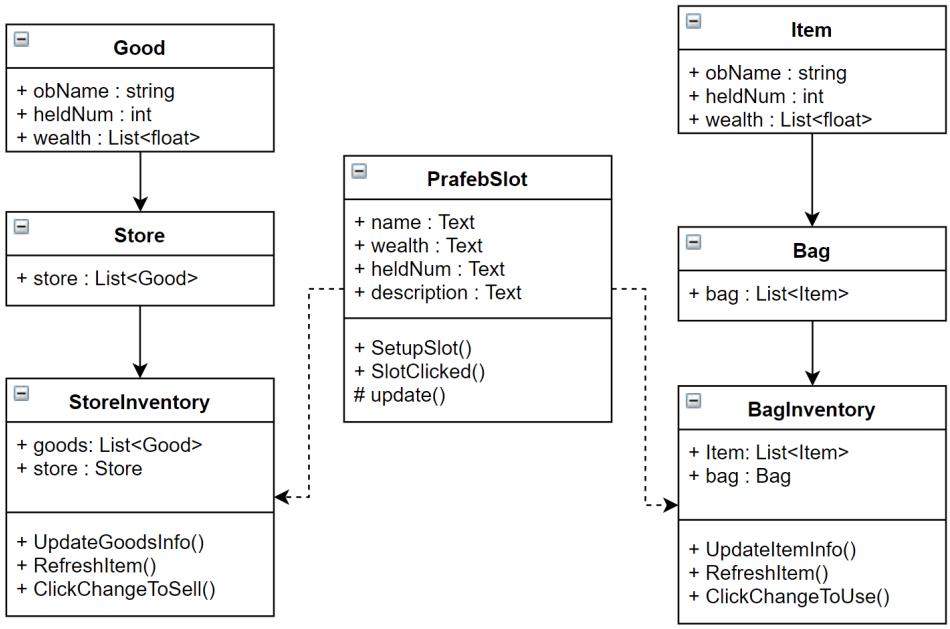

Figure 10. Class diagram of backpack store management system
图 10. 背包商店管理系统类图

Figure 11. UI prefab information of backpack store
图 11. 背包商店 UI 预制体信息

在 StoreInventory 与 BagInventory 脚本公开变量中可以看到 Bag 数据信息，其中存放的是每个商品(Item 类)的信息，通过在 Inventory 类脚本中调用 Bag 属性(数据库信息)和 PrefabSlots(UI 预制体信息类)列表，实现将数据库与游戏 UI 页面之间信息传递。图 12 为 Bag 类数据信息展示，图 13 为背包商店功能成功实现样例。

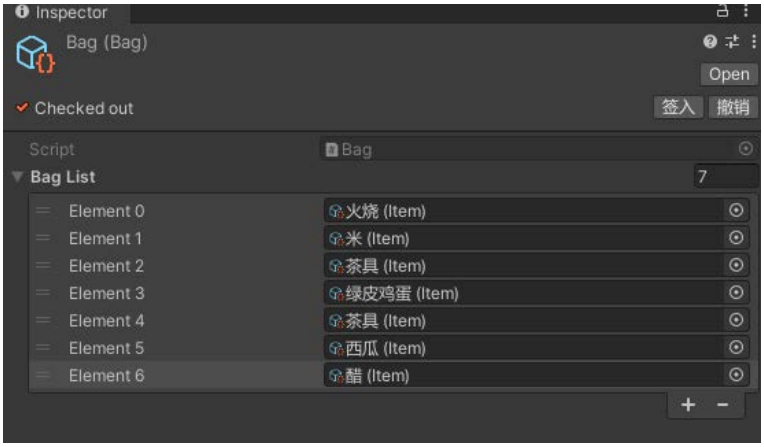

Figure 12. List of Bag data information
图 12. Bag 数据信息列表

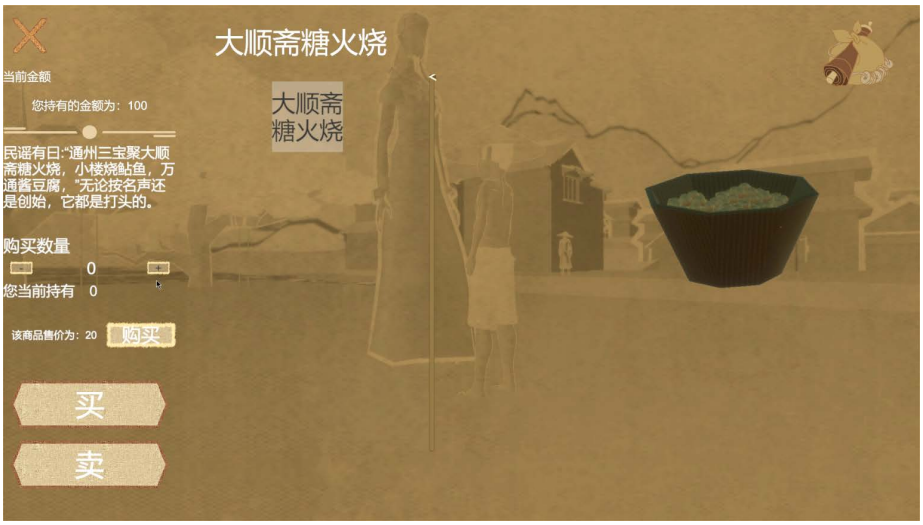

Figure 13. Implementation effect of backpack store
图 13. 背包商店实现效果

5. 剧情加载系统

5.1. 剧情加载系统实现设计

RGP 类型游戏的主要玩法需要依托于大量剧情文字，且根据玩家选择，会触发不同事件以及触发 NPC 不同的对话文字。比较常见的 Unity3D 文本读取功能是在 Text 组件中提前设置好文字信息，利用协程 IEnumerator 从组件本地进行读取文字。这样的方法费时费力，同时也会占用大量的空间，并且需要考虑 RPG 类游戏多结局多方向的文本读取方式，实现起来相当复杂，会浪费大量时间和空间复杂度，使得代码可读性极差。

本章提供了规范文本读取格式，将文字统一封装到 TXT 文件中进行读取。只在触发到对应对话 NPC 文字时进行有目的性的读取文字信息，从而节省大量的空间和时间复杂度。图 14 为常见的文字信息读取方式实现，需要手动敲入文字，并且指定对话 NPC 信息和对话框。图 15 为本章提供的剧情文字信息读取方式实现以及文本信息格式。

Figure 14. Common text reading methods expose variables
图 14. 常见文本读取方式公开变量

Figure 15. This chapter provides text reading modes script exposed variables and text formats
图 15. 本章提供文本读取方式脚本公开变量以及文本格式

除与 NPC 的对话文本文字外，剧情结束后对应会出现任务提示，因此，任务提示版也应与触发的任务事件相匹配。相关任务信息采用 unity 内置的 ScriptableObject 来存放数据信息，结合文本读取对应字段后触发的事件编号来控制任务版的出现与消失。

5.2. 剧情加载文本读取实现方式

文本读取规则如表 1 所示，制定文本读取的相关内容规范，接下来进行代码实现以及对话的 UI 设计即可。

Table 1. Text reading specification
表 1. 文本读取规范

文本读取规则	
占位	功能
1	NPC 剧情起始序号
2	NPC 对话结束标识
3	NPC 下段剧情起始序号

根据文本读取规范，在 TextUIController 类脚本中利用 C#中 String.Split()方法进行分割实现读取相关数据，公开变量 TextUIDA 为指定的文本读取起始序号，TextUIDB 为指定的文本结束标识。过程实现流程图如图 16 所示。

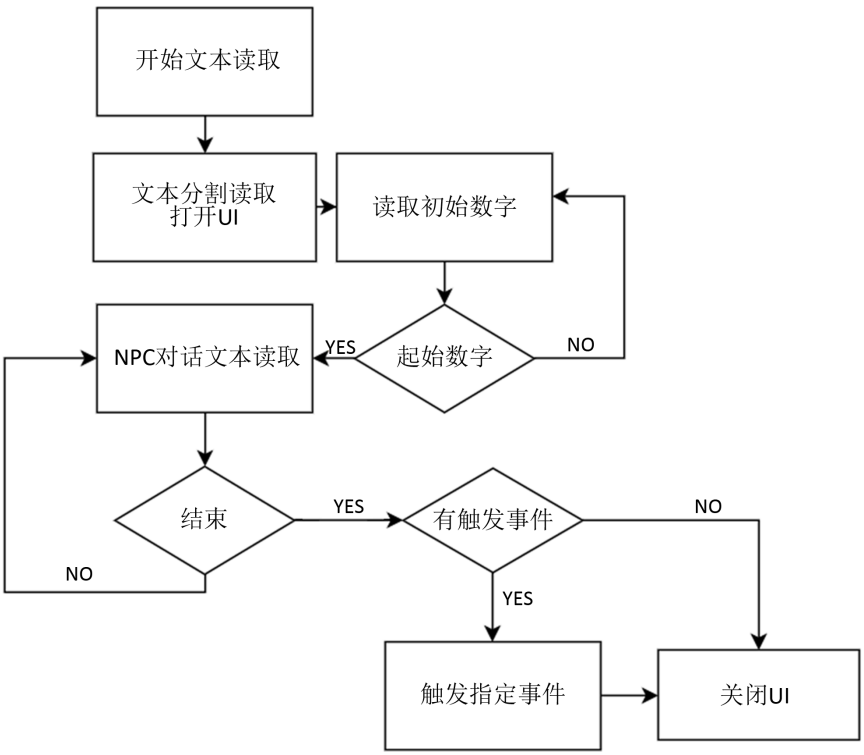

Figure 16. Flowchart of Story loading implementation
图 16. 剧情加载实现流程图

对文字框 UI 挂载命名为 TextPlay 脚本，在脚本内通过 GetComponent 获取 TextUIController 类中数

据，并将其对话信息显示在 UI 中，EvenPool 类脚本为对话结束后触发的相关事件集，内部封装了所有可能触发的事件。图 17 为脚本公开信息，图 18 为文本对话 UI 设计信息。

Figure 17. TextPlay scripts expose variable information
图 17. TextPlay 脚本公开变量信息

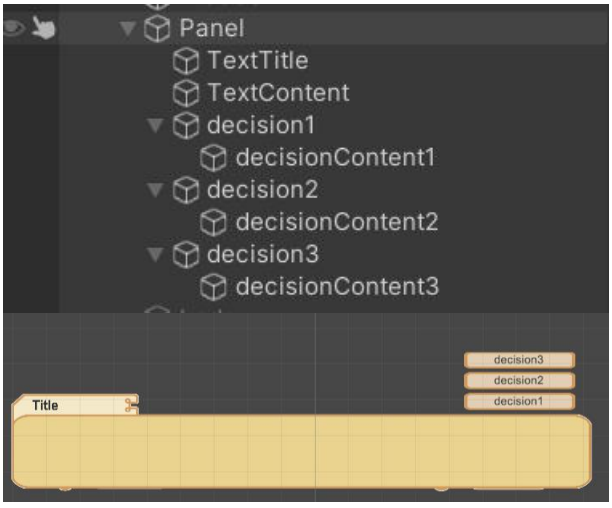

Figure 18. Hierarchy list information
图 18. Hierarchy 列表信息

结合本章上述所述流程，图 19 为最终实现效果。

Figure 19. The final effect
图 19. 最终实现效果

6. 结语

本文针对 RPG 类游戏中关键技术，包括：状态机的角色控制、背包商店系统、剧情加载系统等关键技术给出了实现方案，并且在游戏中进行了验证，验证了技术方案的鲁棒性和兼容性，通过运用在 RPG 类游戏开发中，证明了技术方案的可行性。课题中提供的将有限状态机的思维方式与 Unity3D 角色控制以及动画机相结合，具有一定创新性，使得程序更加简洁，有条理性，同时也节省大量时间复杂度。在背包商店系统中，巧妙地将 ScriptableObject 与 C#脚本之间相互结合，用相对简单的方法实现了背包商店系统之间的交互功能。剧情加载系统上，跳脱出了 Unity 自带的相关文字读取功能，本课题提供了一种更加方便明了的方法，进行本地 TXT 文件读取，实现直接读取对应 NPC 的剧情加载，节省时间复杂度。同时，可以在此课题基础上，将文本信息放到服务器中，从而实现数据库连接读取。课题没有对大量用户同时在线的大型 RPG 类游戏进行方案的验证，在下一步的工作中，将对该项内容进行探索，优化技术方案。

参考文献

[1] 张震. 中国网络游戏发展大事记[J]. 财会月刊(C 财富), 2001(19): 30-31. https://doi.org/10.19641/j.cnki.42-1290/f.2001.19.026

[2] 章国雁. 基于 Unity3D 的多人在线游戏案例设计与实现[J]. 安徽水利水电职业技术学院学报, 2021, 21(3): 46-49.

[3] 张华振. 基于 Unity3D 技术的塔防游戏虚拟人物动作控制方法[J]. 智能计算机与应用, 2022, 12(8): 187-189+195.

[4] 焦灵. Unity3D 引擎 RPG 动作游戏设计[J]. 电脑编程技巧与维护, 2022(3): 141-143. https://doi.org/10.16184/j.cnki.comprg.2022.03.033

[5] 赵若津. 基于 Unity 的游戏功能模块设计与实现[D]: [硕士学位论文]. 山东大学, 2020. https://doi.org/10.27272/d.cnki.gshdu.2020.001620