分类号：**中图分类号** UDC：D10621-408-(选择)流水号-0

密　级：选择编号：输入学号

成都信息工程大学

学位论文

输入论文题目

|  |  |
| --- | --- |
| 论文作者姓名： | 作者姓名 |
| 申请学位专业： | 计算机科学与技术 |
| 申请学位类别： | 工学学士 |
| 指导教师姓名（职称）： | 导师姓名（选择） |
| 论文提交日期： | 选择日期 |

输入论文题目

**摘　要**

建立数据结构课程管理系统，实现数据结构课程的数字化管理，搭建师生之间的在线交流平台，对于提高学校教学水平，改善教学环境具有重大意义。以Oracle为数据库，采用Struts2+Ibatis架构建立基于B/S模式的数据结构课程管理系统，实现了教师对数据结构这门课程的日常教务管理，提供了师生之间的在线交流平台。系统的主要功能包括：教师可以在线维护课程的教学大纲，发布课程公告，维护个人信息，管理教师课件，在线布置学生作业，记录学生出勤，维护管理试卷库，编辑学生上传的学习资源，发布考试安排信息，录入考试成绩等日常教务管理；学生可以在线查询课程相关信息，下载教师课件，查看作业内容，上传作业，查询考试成绩，共享学习资源，在线交流学习。系统还为师生之间的学习交流提供了一个在线交流平台—BBS，方便了教师的答疑解惑，学生的学习交流，为传统的面授交流提供了一种很好的辅助。

**关键词：**Struts2；Ibatis；数据结构；课程管理；Oracle；BBS

Thesis Title

**Abstract**

Developing a data structure course management system, implementing data structure course on digital management, and setting up an online communication platform between students and teachers is of great significance for improving teaching and improving teaching and learning environment. With the database Oracle, the system is developed under the integration of Struts2 and Ibatis framework and based on B/S mode. It provides data structure of daily educational administration to teachers, and an online communication platform between students and teachers. System's main features include: teachers can maintain the course syllabus , post course announcements online, protect personal information, manage teachers courseware, assign homework online, record students attendance, maintain and manage the paper library, edit resources uploaded by students, publish information of Exam schedule, record students achievement and do other daily educational administration; students can check course information online, download teachers courseware, view homework, upload assignments, check examination results, share learning resources, take exchange of learning online. In order to facilitate teachers’ FAQ, and students’ learning, it also provides an online learning platform-BBS between teachers and students. So it is a supplementary system for the traditional face to face communication.

**Key words:** Struts2; Ibatis; Data Structure; Course Management; Oracle; BBS

**目 录**

论文总页数：插入页数页

[1 引言 1](#_Toc512703524)

[1.1 课题背景 1](#_Toc512703525)

[1.2 国内外研究现状 1](#_Toc512703526)

[1.3 本课题研究的意义 1](#_Toc512703527)

[1.4 本课题的研究方法 1](#_Toc512703528)

[2 需求分析及项目方案 2](#_Toc512703529)

[2.1 需求分析 2](#_Toc512703530)

[2.1.1 功能需求 2](#_Toc512703531)

[2.1.2 系统约束条件 2](#_Toc512703532)

[2.2 项目方案 2](#_Toc512703533)

[2.2.1 项目特色 2](#_Toc512703534)

[2.2.2 XXX技术简介 3](#_Toc512703535)

[2.2.3 XXX技术简介 3](#_Toc512703536)

[2.2.4 开发工具和运行环境 3](#_Toc512703537)

[2.2.5 项目实施计划 3](#_Toc512703538)

[3 项目设计及实现 3](#_Toc512703539)

[3.1 数据库设计 3](#_Toc512703540)

[3.2 功能模块的设计与实现 4](#_Toc512703541)

[3.2.1 系统整体框图 4](#_Toc512703542)

[3.2.2 XXX模块的设计与实现 4](#_Toc512703543)

[3.2.3 XXX模块的设计与实现 4](#_Toc512703544)

[4 特殊问题及解决方案 4](#_Toc512703545)

[4.1 XXX问题 4](#_Toc512703546)

[4.1.1 问题描述 4](#_Toc512703547)

[4.1.2 解决方案 4](#_Toc512703548)

[4.1.3 结果 4](#_Toc512703549)

[4.2 XXX问题 4](#_Toc512703550)

[4.2.1 问题描述 4](#_Toc512703551)

[4.2.2 解决方案 4](#_Toc512703552)

[4.2.3 结果 4](#_Toc512703553)

[5 结果测试及性能分析 4](#_Toc512703554)

[5.1 测试概要 4](#_Toc512703555)

[5.1.1 测试环境 4](#_Toc512703556)

[5.1.2 测试范围 4](#_Toc512703557)

[5.2 性能分析 4](#_Toc512703558)

[结 束 语 4](#_Toc512703559)

[参考文献 6](#_Toc512703560)

[致　　谢 7](#_Toc512703561)

[声　　明 8](#_Toc512703562)

# 引言

## 课题背景

目前，基于WEB的管理信息系统已经进入到各行各业中，而且在企业信息的管理中发挥了重要的作用，如基于WEB的人事管理信息系统、基于WEB的销售管理信息系统、基于WEB的医院管理信息系统、基于WEB的材料管理信息系统以及新兴的数字化图书馆、网上购物系统等等，通过WEB可以实现企业信息的快速收集、发布、存储、处理和交流，以较低的费用使企业的业务得到迅速扩展，并使企业获得了可观的社会和经济效益。因此，基于WEB的管理信息系统将成为现代企事业管理信息系统发展的必然趋势。

随着教育现代化，管理信息化时代的到来，为提高学校教学效率，改善教学环境，减少教师不必要的重复劳动，提高教学资源的利用率，加快信息的记录、查阅、传播速度，方便同学间学习交流，实现课程的数字化管理已成必要。

数据结构课程管理系统的开发使用，可以提高学校教学水平，改善教学环境，促进学生和教师的交流，提高学生学习数据结构这门课程的兴趣。

## 国内外研究现状

## 本课题研究的意义

数据结构课程管理系统是一个由人和计算机等设备组成的能进行日常教务管理的系统。它可以让老师从繁杂的工作中解脱出来，从而节省了很多人力和物力。还可以加速教学信息的传播，促进师生之间的交流，从而提高教学效率，达到教育现代化的要求。

本系统本着提高教学水平，改善教学环境，促进教学管理信息化的思想，认真贯彻落实教育面向现代化的精神，严格按照学校课程管理的功能需求设计开发。通过本系统，教师可以轻松，方便地管理学生和教务；学生不仅可以方便、快捷地获取数据结构课程的教学信息、教学资源，还可以在线向老师请教问题，与同学交流学习。从而实现教育现代化和管理信息化的结合。

## 本课题的研究方法

本系统是一个以Oracle为数据库，在Struts2+Ibatis架构下建立的基于B/S结构的课程管理系统。服务器版本是Tomcat6.0，在MyEclipse环境（MyEclipse6.5M1-BlueEdition）下进行开发，编程语言是Java语言。

# 需求分析及项目方案

## 需求分析

### 功能需求

本系统的用户有学生、教师和系统管理员。通过本系统，教师可以进行数据结构课程的日常教务管理；学生可以获取数据结构课程的相关信息、共享学习资源、在线学习交流；系统管理员可以进行系统管理、维护，确保系统的正常运行。总共分为6个模块：学习资源管理、课程相关、BBS论坛、考勤管理、用户管理、成绩管理。

1) 学习资源管理

在此模块中，教师可以管理课件，作业，试卷和其它学习资源；学生可以浏览查看学习资源，下载和上传学习资料与作业。本模块的用例图如图2‑1所示。



图2‑1 学习资源管理模块的用例图

### 系统约束条件

## 项目方案

### 项目特色

1) 良好的平台移植性：本项目采用Unity3D作为开发引擎，Unity3D的C#基础脚本模块是通过Mono来实现的，Unity通过Mono将C#脚本代码编译成CLI，然后Mono运行时利用JIT或者AOT将CLI编译成目标平台的原生代码实现跨平台。

2) AI系统：本项目中敌人AI行为丰富，有自动巡逻行为、静止观察行为与追击行为等，且可对障碍物、地面边缘等特殊情况进行智能检测和规避。

3) 道具系统：本游戏中提供多种属性、内容各不相同的丰富道具，并通过可拾取系统对其进行交互设计，使得道具在被拾取后可加入技能栏，通过对战斗属性的类型与数值修改，加上各类视觉特效，给予玩家丰富的游戏反馈，提升战斗体验。

4) 仓库系统：对道具系统进行交互支持，由此衍生仓库系统，可支持对拾取道具的插槽显示与管理；

5) 技能系统：结合道具系统与仓库系统，玩家可根据道具处于技能栏位置所对应的指定按键来释放技能，并加入技能冷却等视觉效果。

### 关卡异步加载技术简介

1. 背景：

游戏中经常需要进行场景的加载，而有时候有些场景因为需要加载大量的资源，导致在加载场景时感觉很卡；另一方面，在场景后台进行耗时较长的资源加载的过程中，若始终不给予玩家以必要的视觉反馈，玩家将在这段时间内的等待中感到疑惑和不确定，产生糟糕的游戏体验。这时候需要使用异步场景加载来解决这样的问题。

1. 异步加载场景：

**Func**:

public static AsyncOperation LoadSceneAsync(sceneName, mode)

public static AsyncOperation LoadSceneAsync(sceneIndex, mode)

**Parameters**:

sceneName: Name or path of the Scene to load.

sceneBuildIndex: Index of the Scene in the Build Settings to load.

Mode: If LoadSceneMode.Single then all current Scenes will be unloaded before loading.

* 1. 描述：

若调用同步加载场景API：LoadScene(int sceneBuildIndex)，则当前场景中的所有游戏逻辑都将停止，以进行目标场景资源的加载过程，但若调用后台异步加载功能，则当前场景的游戏逻辑正常执行，资源加载过程在后台进行异步执行，并可以实时获取加载进度，还可对是否需要在目标场景资源加载完毕后进行自动跳转。

在后台异步加载场景，给定的场景名称既可以是完整的场景路径，也可以是构建设置窗口中显示的路径，也可以是场景名称或场景下标值。 如果只给出场景名称，则会加载匹配列表中的第一个场景。 如果有多个具有相同名称但路径不同的场景，则应在构建设置中使用完整的场景路径。

* 1. 应用：

通常游戏的主场景包含的资源较多，这会导致加载场景的时间较长。为了避免这个问题，可以首先加载Loading场景，然后再通过Loading场景来加载主场景。因为Loading场景包含的资源较少，所以加载速度快。在加载主场景的时候，由于耗时可能较长，一般会在Loading界面中显示一个进度条来告知玩家当前加载的进度，以获得更佳的游戏体验与视觉效果。在Unity中可以通过调用SceneManager.LoadSceneAsync函数来异步加载游戏场景，通过查询AsyncOperation.progress的值来得到场景加载的进度。

float displayProgress, destProgress;

displayProgress = destProgress = 0;

while (operation.progress < 0.9f) {

destProgress = operation.progress \* 100f;

while (displayProgress < destProgress) {

// 对于当前进度数值，每一帧在实际的基础上+1

loadingBar.transform.Find("LoadingProgText").GetComponent<Text>().text = ++displayProgress + "%";

loadingBar.value = displayProgress / 100f; // 设置进度条数值

yield return new WaitForEndOfFrame();

}

}

### 帧动画技术简介

1. 概念：通过在关键帧设置该帧对应的精灵图片，将多帧连接起来，则形成由图片序列组成的2D帧动画；
2. 应用：导入所需动画的精灵图片序列，通过Unity自带的动画编辑器，选取关键帧，并在该帧从导入的动画图片序列中选取并设置欲表现的单张精灵图片，完成对该帧的编辑。对每一帧做相同步骤的操作，再播放此动画剪辑，则形成由精灵图片序列组成的2D帧动画。

### 协同程序技术简介

1. 概念：Coroutine在Unity3D中叫做协程或协同程序，和多线程类似，也就是说开启协同程序就是开启一个线程。但是在任意指定时刻只有一个协程执行，其他协程挂起。
2. Coroutine的相关函数：
   1. StartCoroutine：启动一个协程。
   2. StopCoroutine：终止一个协程。
   3. StopAllCoroutine：终止所有协程。
   4. WaitForSeconds：等待几秒。
   5. WaitForFixedUpdate：等到下一次FixedUpdate调用时执行。

使用MonoBehaviour.StartCoroutine方法即可开启一个协同程序，也就是说该方法必须在 MonoBehaviour 或继承于MonoBehaviour的类中调用。

1. Coroutine in Unity:

在Unity3D中，StartCoroutine(string methodName),

StartCoroutine(IEnumerator routine)都可以开启一个线程。

区别在于使用字符串作为参数可以开启线程并在线程结束前终止线程，相反使用IEnumerator 作为参数只能等待线程的结束而不能随时终止(除非使用StopAllCoroutines()方法)；

另外使用字符串作为参数时，开启线程时最多只能传递 一个参数，并且性能消耗会更大一点，而使用IEnumerator 作为参数则没有这个限制。

在Unity3D中，使用StopCoroutine(string methodName)来终止一个协同程序，使用StopAllCoroutines()来终止所有可以终止的协同程序，但这两个方法都只能终止该 MonoBehaviour中的协同程序。

### 跨场景存储技术简介

游戏中经常存在一些重要的数据和属性，它们需全局存在、贯穿整个游戏，以便随时获取和显示出来，而在Unity中，所有同一场景下的游戏物体，在跳转至新场景后，将全部被自动销毁。为了避免上述重要的数据和属性随场景的跳转而被自动销毁，本项目中采用结合单例（Singleton）和DontDestroyOnLoad的方案，对跨场景的数据存储提出相应的解决方案：

单例（Singleton）是一种基本的设计模式，对单例类的实现能够确保在任何时候该类只有一个对象实例存在。对于在游戏过程中不需要进行多次实例化的类或物体，建议使用单例模式，典型的如一些控制类：GameManager, AudioController等。

public static PlayerController2D GetInstance {

get {

if (\_instance == null) {

\_instance = FindObjectOfType<PlayerController2D>();

}

return \_instance;

}

}

private void Awake() {

if (\_instance == null) {

\_instance = this;

DontDestroyOnLoad(transform.root.gameObject);

} else {

Destroy(gameObject);

}

}

DontDestroyOnLoad()是Unity自身提供的API，它的主要功能是：使得指定的游戏物体在加载新场景时不被自动销毁。当加载新场景时，旧场景的所有游戏物体都将被销毁，然后再加载新场景的游戏物体。在场景加载过程中，若要保存一个对象，则对其调用DontDestroyOnLoad()，若该对象是组件或游戏对象，则其整个层次结构，包括其所有的子物体及其层次结构，都不会被销毁和改变。 DontDestroyOnLoad()方法常用于保存一个全局的游戏脚本，比如一些第三方插件的Utility类等，使其全程在场景中得以保留，使用起来非常方便。但即便如此，它同时存在一个很重要的隐患：场景来回跳转导致产生多个实例。比如：在场景A的某个脚本的Start方法中调用：DontDestroyOnLoad(ObjectA)，而后场景从A切换至B，ObjectA随之被保留了下来，存在于场景B；但如果再从场景B跳转回场景A，DontDestroyOnLoad()就又被执行了一次，然而之前场景A中本就存在的ObjectA依然存在，这样就在同一场景中产生了多个实例，且只要场景不断来回跳转，这种情况就有可能无限循环下去。

因此，在对DontDestroyOnLoad()的使用上，需要结合单例模式，使场景中在任何时候都存在且只存在一个对象实例，为跨场景保存数据的技术提供可靠、有保障的解决方案。

### 开发工具和运行环境

1. 系统开发工具和开发环境如表2‑1所示。

表2‑1 开发工具表

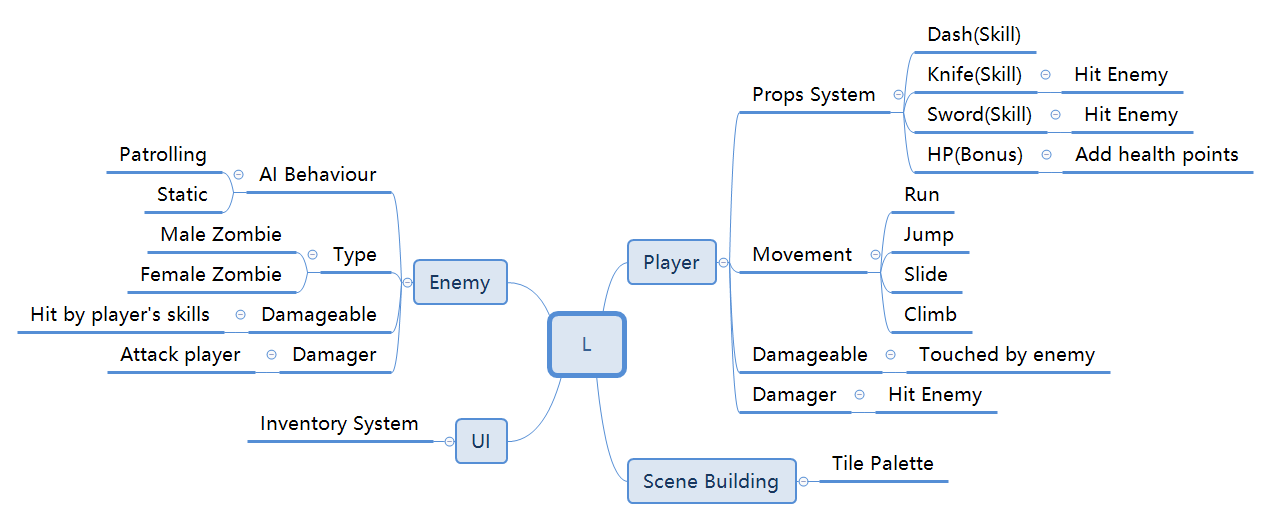
|  |  |
| --- | --- |
| **开发工具名称** | **软件用途** |
| **Unity5.6.0f** | 开发平台 |
| **Windows 10 Pro** | 运行环境 |
| **应用宝** | 移动端版本打包工具 |
| **XMind** | 系统设计，创建流程图等 |
| **Adobe Photoshop CS6** | 美术素材制作与编辑 |
| **ImageMagick** | 美术素材批处理 |

### 项目实施计划

# 项目设计及实现

## 功能模块的设计与实现

### 系统整体框图



### AI模块的设计与实现

1. 设计：

本项目中AI行为模式主要分为以下两种：巡逻式和等待式。巡逻式行为模式，是在玩家远离敌人视线之外时，敌人对周围环境进行自动巡逻，巡逻过程中，将在距离地面边缘指定的距离位置或障碍物面前自动停下，并静止一段时间，模拟观察行为，之后转身，朝反方向进行对另一侧的巡逻；而在玩家进入敌人的可视范围内时，敌人将加速朝玩家进行追击，并在接近玩家时触发僵尸啃食的攻击行为。而等待式行为，与巡逻式的主要差别在于，玩家远离敌人时敌人只停留在原地静止不动，不进行自动巡逻，其余部分与巡逻式大致相同。

1. 实现：

巡逻式的AI行为模式其中有一点至关重要：自动判断是否行至障碍物面前或地面边缘处，若是则应自动停下。两者均用到对指定层级的物体的碰撞检测。第一，判断是否行至障碍物面前，使用Physics2D.CircleCast()方法，该方法在敌人位置投射一个指定半径的圆圈，返回第一个与之接触的、并属于指定layerMask，即障碍物所属层级的碰撞体，若返回值不为空，则说明敌人已行至障碍物前，不能再继续前进；同理，判断是否行至地面边缘，只需将圆圈的投射位置相对于敌人运动方向平移若干个单位，并将投射方向改为垂直向下，则可提前判断该圆圈是否与地面相接触，若是，则尚未行至边缘，可继续前进，否则行至边缘，应停止前进。综上所述，若敌人已行至障碍物前，即判定圈在水平方向上与任意一个属于障碍物层级的碰撞体相接触，或行至地面边缘，即判定圈在垂直方向上不与任何一个属于地面层的碰撞体相接触，则应停止运动，进行静止观察行为；否则继续朝当前方向运动。

静止的观察行为，其逻辑大致描述如下：将速度设置为0，使其停止，然后使用协程等待一段时间，期间保持静止状态，而后改变其Sprite组件的flipX属性，将其置负，使其进行水平翻转，模拟转身；再将运动速度设置为与静止前反向等大的值，使其朝反方向运动。

public bool CheckForObstacle(float forwardDistance) {

if (Physics2D.CircleCast(m\_Collider.bounds.center, m\_Collider.bounds.extents.y - 0.2f, m\_SpriteForward, forwardDistance, m\_Filter.layerMask.value)) {

return true;

}

// origin point of ground edge detecting

Vector3 castingPosition = (Vector2)(transform.position) + m\_SpriteForward \* m\_LocalBounds.extents.x \* 0.5f;

// if about to hit nothing of Ground layer, which means at the edge of ground

if (!Physics2D.CircleCast(castingPosition, 0.1f, Vector2.down, m\_LocalBounds.extents.y + 0.2f, m\_CharacterController2D.groundedLayerMask.value)) {

return true;

}

return false;

}

### 关卡加载模块的设计与实现

1. 设计
   1. 异步加载：不影响当前游戏场景的前提下加载新场景。通常异步加载的方式分为两种：第一种是异步加载新游戏场景，当新场景加载完成后进入新场景并且销毁之前的场景。第二种，同样异步加载新场景，新场景加载完毕后，保留旧场景的游戏对象并且进入新场景。本项目使用第一种方式进行场景的异步加载。
   2. 加载界面：通常游戏的主场景包含的资源较多，这会导致加载场景的时间较长。为了避免这个问题，可以首先加载Loading场景，然后再通过Loading场景来加载主场景。因为Loading场景包含的资源较少，所以加载速度快。在加载主场景的时候一般会在Loading界面中显示一个进度条来告知玩家当前加载的进度。
2. 实现：
   1. 异步加载：

在Unity中可以通过调用SceneManager.LoadSceneAsync函数来异步加载游戏场景；

* 1. 进度条：通过查询AsyncOperation.progress的值来得到场景加载的进度。通过Slider组件实现加载界面。

float displayProgress, destProgress;

displayProgress = destProgress = 0;

while (operation.progress < 0.9f) {

destProgress = operation.progress \* 100f;

while (displayProgress < destProgress) {

// 对于当前进度数值，每一帧在实际的基础上+1

loadingBar.transform.Find("LoadingProgText").GetComponent<Text>().text = ++displayProgress + "%";

loadingBar.value = displayProgress / 100f; // 设置进度条数值

yield return new WaitForEndOfFrame();

}

}

destProgress = 100f;

while (displayProgress < destProgress) {

loadingBar.transform.Find("LoadingProgText").GetComponent<Text>().text = ++displayProgress + "%";

loadingBar.value = displayProgress / 100f;

yield return new WaitForEndOfFrame();

}

operation.allowSceneActivation = true;

### 道具系统模块的设计与实现

1. 设计：

根据道具属性，分为两类：Skill和Bonus，Skill是技能类道具，Bonus是奖励类道具，技能类道具只能主动使用，若不使用则不作用；奖励类道具只作被动释放，得到后即作用。

根据道具内容，共细分为四类：Sword, Dagger, Dash\_Shoe, Health\_Point，分别为：短剑、飞镖、滑板鞋、血瓶。短剑、飞镖与滑板鞋均为技能类道具，需要主要释放，且具有冷却时间；血瓶为奖励类道具，被动释放，无冷却时间。

其中短剑与飞镖作为攻击型技能，释放后对敌人具有伤害效果，滑板鞋为运动类技能，释放后可迅速向前滑行一段距离；血瓶为奖励类道具，拾取后被动释放，直接为玩家增加一点生命值。

上述所有道具均为可拾取类物体，且仅与Player发生可交互触发。

对所有道具设计一道具类TackleInfo.cs，存储道具对象必需的信息。道具类包含的字段：tackleName(道具名), tackleProperty(道具类型), tackleContent(道具内容), isPicked(是否被拾取), tackleSprite(道具图片), tackeIcon(道具图标)；并包含一个事件：onPickUp, 即该道具被玩家拾取后的回调。

对玩家添加一道具拾取类PlayerTackleControl.cs，管理其对道具的拾取功能及对各类道具拾取后的不同表现。

1. 实现：

在PlayerTackleControl.cs中，实现OnTriggerEnter2D()，仅对道具类标签物体进行响应，一旦成功拾取道具物体，则触发该道具在被拾取后的回调方法。同时，对道具种类进行判断，若是技能类道具，则开启玩家的相应技能的释放权限，如短剑、飞镖与滑行技能；若是奖励类道具，则玩家的相应属性直接得到被动增益，如拾取到血瓶则直接增加玩家生命值。

private void OnTriggerEnter2D(Collider2D collision) {

// not interact with tackle layer

if (!collision.gameObject.layer.Equals(tackleLayerNum)) {

return;

}

// no tackle settings attached

if (collision.gameObject.GetComponent<TackleInfo>() == null) {

return;

}

// the tackle has been picked up already

if (collision.gameObject.GetComponent<TackleInfo>().isPicked) {

return;

}

TackleInfo tackle\_Triggered = collision.gameObject.GetComponent<TackleInfo>();

tackle\_Triggered = collision.gameObject.GetComponent<TackleInfo>();

tackle\_Triggered.GetPickedUp();

}

### 仓库、技能模块的实现

1. 设计：

仓库模块系根据道具模块所衍生出的、进一步增加游戏视觉体验而设计而出的模块，本项目中的仓库模块主要负责将拾取的技能可视化，放置于相应的技能栏插槽中，并在玩家释放技能时，展现其冷却CD效果。

首先需设计一个插槽类InventorySlot.cs，作为技能类插槽与奖励类插槽的基类，本项目目前并未用到奖励类插槽，不过从基类开始写起，便于扩展与复用。插槽类包含的主要字段：slotIndex(插槽下标/id), tackleType(插槽内道具的类型), tackleContent(插槽内道具的内容), isFilled(该插槽是否被填充)；方法：OnSlotContentUsed()，当该插槽的技能被使用时被调用；OnFilledFuncImplement()，当该插槽被填充后被调用。

创建技能插槽类SkillSlot.cs，并继承于InventorySlot，重写OnFilledFunc()与OnSlotContentUsed()方法。

创建插槽管理类InventoryManager.cs，主要用于管理所有可用插槽，包括可用插槽的创建、动态增减和获取，对于所有插槽的管理可用队列结构来进行控制。

1. 实现：

对于技能插槽SkillSlot中的重写方法：

首先，对OnFilledFuncImplement()进行重写：当技能被填充至插槽时，即技能已被拾取且添加至技能栏中后，根据该技能在技能栏中的位置，设置该技能的释放按键，例如，捡到滑板鞋道具并放置于技能栏中第1个技能插槽中，则释放滑板鞋技能的按键为数字键1，以此类推；对于OnSlotContentUsed()进行重写：在释放了该插槽所对应的技能后，应在该方法内执行对技能的冷却操作，即通过协程对技能释放的操作权限进行时间间隔限制，并在UI层面基于对图片的环形填充，对技能图片加上冷却视觉效果。

// mask's cool down effect

IEnumerator Cooldown() {

float elapsedTime = 0f;

this.img\_Mask.gameObject.SetActive(true);

this.img\_Mask.fillAmount = 1f;

while(elapsedTime < coolDownDuration) {

elapsedTime += Time.deltaTime;

this.img\_Mask.fillAmount = 1 - elapsedTime / coolDownDuration;

yield return null;

}

this.img\_Mask.gameObject.SetActive(false);

}

对于InventoryManager的实现：首先设置所有可用插槽的列表，组成插槽队列，每次拾取到需要加入至技能栏的道具时，对插槽队列出队，并获取该出队元素，作为本次使用的插槽，即把刚拾取到的道具放入该插槽中。随着对新道具的拾取，可用插槽队列的数量也随之减少，直至所有可用插槽均被用完，即插槽队列中所有元素均已出队。

private void Start() {

// init the slots queue

for (int i = 0; i < slots.Count; i++) {

availableSlots.Enqueue(slots[i]);

}

}

// Get the first available slot and dequeue it(cause it will be filled once got returned)

public InventorySlot GetAvailableSlot() {

if (availableSlots.Count <= 0) {

Debug.Log("No vailable slot now!!");

return null;

}

return availableSlots.Dequeue();

}

### 可（被）攻击模块的设计与实现

1. 设计：

无论对于玩家还是敌人而言，均可对对方造成伤害，也都可被对方所伤害，故此从中抽象出两个模块：可攻击模块Damager，和可被攻击模块Damageable。

对Damager类，应包含的主要字段有：hittableLayers，表示可与之进行交互的，即可被该类实例对象所伤害的物体的层级；其次，包含以下两个事件：OnDamageableHit, OnNonDamageableHit，分别表示该Damager实例碰撞到可被攻击的物体与不可被攻击的物体时触发的两类事件；另外，还包括以下两个方法：EnableDamage()和DisableDamage()，分别表示开启与关闭攻击能力。

对Damageable类，包含的主要字段有：startingHealth(初始生命值)，invulnerableAfterDamage(被伤害后是否无敌)，invulnerabilityDuration(被伤害后无敌状态持续的时间)，disableOnDeath(死亡后隐藏该物体)；四个事件：OnHealthSet，对应生命值的更新；OnTakeDamage，对应受到Damager的伤害；OnDie，死亡；OnGainHealth，增加生命值；几种方法：SetInitialHealth()，设置初始生命值；DisableInvulnerability()，关闭无敌能力；EnableInvulnerability()，开启无敌能力；TakeDamage()，受到伤害；GainHealth()，获得生命值；SetHealth()，设置生命值。

1. 实现：

通过调用Physics2D.OverlapArea()，判断碰撞体是否与指定的矩形区域相交，以此判断该Damager实例是否与指定层级的物体发生了触发式碰撞。若是，则对该与之交互的物体进行检测：若该物体为Damageable类对象，表示其为本Damager实例的攻击对象，则一来调用自身对Damageable物体的攻击回调：OnDamageableHit，二来调用该物体的Damageable.TakeDamage()方法，进行伤害判定；否则为非Damageable类对象，则只调用自身对NonDamageable物体的攻击回调。

int hitCount = Physics2D.OverlapArea

(pointA, pointB, m\_AttackContactFilter, m\_AttackOverlapResults);

for (int i = 0; i < hitCount; i++) {

m\_LastHit = m\_AttackOverlapResults[i];

Damageable damageable = m\_LastHit.GetComponent<Damageable>();

if (damageable) {

OnDamageableHit.Invoke(this, damageable);

damageable.TakeDamage(this, ignoreInvincibility);

if (disableDamageAfterHit)

DisableDamage();

} else {

OnNonDamageableHit.Invoke(this);

}

}

# 特殊问题及解决方案

至少列出2个问题。

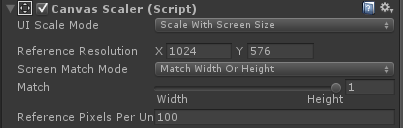
## UI布局问题

### 问题描述

若将游戏分辨率切换至另一不同于初始设置的分辨率，则UI布局杂乱或不可见。

### 解决方案

1. 自适应：游戏中的分辨率自适应主要包括两部分：一是在不同尺寸的屏幕下，整体缩放比例的计算方式；二是在不同比例（宽高比）的屏幕下，UI控件所处的位置关系，也称为布局。
2. UGUI缩放机制：
   * 1. Constant Pixel Size：固定像素尺寸，即按素材的“固定像素”渲染；
        1. Scale Factor：缩放比例，在素材原尺寸上的缩放比例，默认值是1；
        2. Reference Pixels Per Unit：每个unity单位对应的像素数；
     2. Scale With Screen Size：根据屏幕尺寸缩放；
        1. Reference Resolution：标准分辨率，这是我们提供给美术做图的标准分辨率，所有的UI素材都应该按这个分辨率去做；
        2. Screen Match Mode：Match Width Or Height 以宽高权重匹配；
        3. Match：宽高所占权重，默认值是0，相当于以“标准分辨率的宽”和“实际屏幕的宽”的比例作为缩放比例。同理，如果值是1，相当于以“标准分辨率的高”和“实际屏幕的高”的比例作为缩放比例。如果值是0.5，则相当于宽和高的比例权重相等，最终的缩放比=宽缩放比\*宽权重+高缩放比\*高权重；
        4. Reference Pixels Per Unit：每个unity单位对应的像素数；
     3. Constant Physical Size：固定物理尺寸；
        1. Physical Unit：物理单位，包括点，英寸，厘米，毫米等；
        2. Fallback Screen DPI：对应物理单位的像素密度；
        3. Default Sprite DPI：默认精灵的像素密度；
        4. Reference Pixels Per Unit：每个unity单位对应的像素数
3. 使用方法：
   1. Canvas Scaler 选择 Scale With Screen Size；
   2. Screen Match Mode 选择 Match Width Or Height，比例设为1，即只和高度进行适配；



### 结果

在不同分辨率下，UI元素可根据实际情况进行自适应：

1. 保持相对位置不变；
2. 保持宽高比例不变。

## 脚本引用丢失问题

### 问题描述

对脚本资源进行修改后，部分游戏物体上的某些组件提示“Nothing Selected”，脚本引用发生丢失。导致子弹爆炸效果未能正常销毁。

### 解决方案

将版本回退至对脚本资源进行修改前的正常版本，Unity将自动导入先前丢失的资源与引用。

### 结果

引用被重新定位，组件恢复正常，爆炸效果贴图被正常销毁。

## 进度条过渡生硬问题

### 问题描述

通过LoadSceneAysnc进行异步加载时，进度条的数值更新不连续的，效果生硬不自然。

### 解决方案

每一次更新进度条的时候插入过渡数值。当获得AsyncOperation.progress的值后，不立即更新进度条的数值，而是每一帧在原有的数值上加1，这样就会产生数字不停滚动的动画效果。

while (operation.progress < 0.9f) {

destProgress = operation.progress \* 100f;

while (displayProgress < destProgress) {

// 对于当前进度数值，每一帧在实际的基础上+1

loadingBar.transform.Find("LoadingProgText").GetComponent<Text>().text = ++displayProgress + "%";

loadingBar.value = displayProgress / 100f; // 设置进度条数值

yield return new WaitForEndOfFrame();

}

}

### 结果

经过插值操作之后，进度条数值每一帧都在实际加载进度的范围内进行更新，过渡自然，实现了柔和的动画效果。

# 结果测试及性能分析

具体内容可从系统测试报告中节选

## 测试概要

### 测试环境

### 测试范围

## 性能分析

结 束 语

本系统根据学校课程管理的实际需求，在Struts2+Ibatis架构下设计开发而成。系统以美观友好的人机交互界面和较为完善的功能体系，为教师的数据结构课程日常教务管理和学生的数据结构课程学习提供了一个方便、快捷的辅助平台。运行安装本系统后，教师可以从繁杂的传统管理模式走向高效的数字化管理模式。通过本系统教师可以科学，系统地管理日常教学教务。系统的主要功能包括：教师可以在线维护课程教学大纲，发布课程公告，维护个人信息，管理教师课件，在线布置学生作业，记录学生出勤，维护管理试卷库，编辑学生上传的学习资源，发布考试安排信息，录入考试成绩等日常教务管理；学生可以在线查询课程相关信息，下载教师课件，查看作业内容，上传作业，查询考试成绩，共享学习资源，在线交流学习。系统还为师生之间的学习交流提供了一个在线交流平台—BBS，方便了教师的答疑解惑，学生的学习交流。实现了教育现代化和管理信息化的有机结合，充分提高了教学资源的利用率，为教学水平的提高和教学环境的改善提供了捷径。

参考文献

1. 舒红平, 周定文, 何嘉, 邹书蓉. Web数据库编程-Java[M]. 西安: 西安电子科技大学出版社, 2005.
2. 王慧芳, 毕建权. 软件工程[M]. 浙江: 浙江大学出版社, 2005.
3. 吉根林, 崔海源. Web程序设计(第二版)[M]. 北京: 电子工业出版社, 2006.
4. 王珊, 萨师煊. 数据库系统概论[M]. 浙江: 浙江大学出版社, 2005.
5. 尚学堂-王勇-精通struts1.x. http://www.itcast.net/.
6. 传智播客ibatis视频教程. http://www.itcast.net/.

致　　谢

在论文完成过程中，本人还得到了 老师和 同学的热心帮助，……！

最后向在百忙之中评审本文的各位专家、老师表示衷心的感谢！

**作者简介：**

|  |  |  |  |
| --- | --- | --- | --- |
| 姓 名： | 姓名 | 性别： | **选择** |
| 出生年月： | 选择日期 | 民族： | **民族** |
| E-mail： | 电子邮件 |  |  |

声　　明

本论文的工作是**选择**年10月至**选择**年**选择**月在成都信息工程大学计算机学院完成的。文中除了特别加以标注地方外，不包含他人已经发表或撰写过的研究成果，也不包含为获得成都信息工程大学或其他教学机构的学位或证书而使用过的材料。

关于学位论文使用权和研究成果知识产权的说明：

本人完全了解成都信息工程大学有关保管使用学位论文的规定，其中包括：

（1）学校有权保管并向有关部门递交学位论文的原件与复印件。

（2）学校可以采用影印、缩印或其他复制方式保存学位论文。

（3）学校可以学术交流为目的复制、赠送和交换学位论文。

（4）学校可允许学位论文被查阅或借阅。

（5）学校可以公布学位论文的全部或部分内容（保密学位论文在解密后遵守此规定）。

除非另有科研合同和其他法律文书的制约，本论文的科研成果属于成都信息工程大学。

特此声明！

作者签名：

年 月 日

**参考文献格式说明**

不同类型的参考文献，请使用如下格式进行列出。所有标点符号使用英文半角，如“, . :”，英文标点后通过空格与后面的文字进行分开，不能用中文标点。

**期刊类：**

[序号] 作者. 篇名[J]. 刊名, 出版年份, 卷号(期号): 起止页码

**专著类：**

[序号] 作者. 书名[M]. 出版地: 出版社, 出版年份. 起止页码

**报纸类：**

[序号] 作者. 篇名[N]. 报纸名, 年 月 日

**专利：**

[序号] 专利所有者. 题名[P]. 国别: 专利号, 年 月 日

**标准：**

[序号] 标准编号, 标准名称[S]

**电子文献：**

[序号] 主要责任者. 电子文献题名[EB/OL]. 电子文献出处或可获得地址, 发表日期/更新日期/引用日期

**1 专著、书籍、译著**

【格式】序号 责任者[外国作者请注明国籍]. 书名（版本，第1版不写）[文献类型M]. 其他责任者（如译者）.出版社所在地: 出版社名称（出版者）, 出版年.月：起止页码.

1. 徐德蜀, 邱成.安全文化通论[M]. 北京: 化学工业出版社, 2005.10
2. 克鲁克洪[美]. 文化与个人[M]. ×××译. 杭州: 浙江人民出版社, 1986
3. Herskovits M J. Cultural Anthropology[M]. New York: Knopf, 1955

**2 连续出版物（期刊）**

【格式】序号 析出责任者（第1作者, 第2作者, 第3作者等）. 析出题名. 刊名, 出版年, 卷号（期号）: 起止页码.

1. 徐德蜀.安全文化、安全科技与科学安全生产观[J]. 中国安全科学学报, 2006, 16(3): 71-82.
2. W.K. Chow. Simulation of tunnel fires using a zone model[J]. Tunneling and Underground Space Technology, 1996, 11(2):221-236.
3. H.Graf\*, H.Schmidt-Traub. Early hazard identification of chemical plants with state chart modeling techniques[J]. Safety Science, 2000, 36:49-67.

**3 会议论文集**

【格式】序号 析出责任者. 析出题名[文献类型A]. 文集编者. 文集名[文献类型C]. (供选项: 会议名, 会址, 开会年.) 出版地: 出版者, 出版年: 起止页码.

1. 李融融. 依法促进中小企业健康发展[A]. 中国中小企业发展年鉴[C]. 北京:中国经济出版社, 2004: 1～24.
2. 徐德蜀, 汪国华, 张爱军. 浅谈"安全生产五要素"与安全科技[A]. 第十四届海峡两岸及香港、澳门地区职业安全健康学术研讨会暨中国职业安全健康协会2006年学术年会论文集[C], 2006. 5:64-71.
3. R. E. Briter. Recent research on the dispersion of hazardous materials [A]. International Conference and Workshop on Modeling the Consequences of Accidental Releases of Hazardous Materials[C]. San Francisco, California: 1999:197-230。

**4 学位论文**

【格式】序号 责任者. 题名[文献类型D]. 学位授予地: 学位授予单位[硕士或博士论文], 年份.

1. 韩艳. 地震作用下高速铁路桥梁的动力响应及行车安全性研究[D]. 北京: 北京交通大学[博士论文], 2005.
2. 徐茂波. 考虑施工期间人为错误的结构安全分析与控制[D]. 北京: 清华大学[博士论文], 1998.
3. 田水承. 第三类危险源辨识与控制研究[D]. 北京: 北京理工大学[博士论文], 2001.7: 35-39.
4. 易云兵. 基于风险评价的输气管线设计方法研究[D]. 成都: 西南石油大学[硕士论文], 2005.6.

**5 专利**

【格式】序号 专利申请者. 专利题名[文献类型P]. 专利国别(或地区). 专利号， 发布日期.

1. Samuel P Schillaci. Chlorine gas filtering material suitable for use in a chemical oxygen generator[P]. USP, 4687640, 1987.01.01.
2. 王庆禄, 李汝南. 陶瓷超微孔臭氧曝气装置[P]. 中国. ZL00231153.4, 2001.

**6 报纸**

【格式】序号 责任者. 题目[文献类型N]. 报纸名, 出版日期(版次).

1. 吴宗之. WHO安全社区建设标准简介[N]. 中国安全生产报, 2004-07-08(8).
2. 王云. 俄罗斯紧急情况部[N]. 环球时报, 2006-01-26(5).

**7 科技报告**

【格式】序号 责任者. 报告题目[文献类型R]. 报告代码及编号. 地名: 责任单位, 年份.

1. 安全工程学科教学指导委员会. 关于在《授予博士、硕士学位和培养研究生的学科、专业目录》中设立“安全科学与工程”一级学科的论证报告[R], 2005.8.
2. 煤矿安全会诊专家组第三组. 煤矿安全技术会诊技术报告[R], 2005.
3. 国务院核电办人因工程研究专家组. 核电厂人因工程研究[R]. "八五"科技成果报告, 1991: 1-50.

**8 标准**

【格式】序号 责任者. 标准名[文献类型S]. 标准编号, 出版年.

1. 汽车行驶记录仪[S]. GB/T19056-2003, 2003.
2. 中华人民共和国国家标准. 光气及光气化产品生产安全规程[S]. GB19041-2003, 2003.

**9 电子文献**

【格式】序号 责任者. 文献题名[文献类型EB/OL]. 出处或可获得地址, 发表或更新日期/引用日期.

1. 王东军. 反"三违", 保安全[EB]. http://www.hdgl.gov.cn/, 2006-07-27.
2. 我国今年来发生94起特大安全事故2074人死亡[OL]. 中国新闻网. http://www.sina.con.cn, 2005-10-25.