DOC

3.25 future/promise

3.25.1 设计模式含义

future/promise模式,用于解决一般多线程编程难以返回一个返回值的问题.

undercook中,食材工程需要多线程的发送食材给初始,future/promise模式让食材工厂在完成对全部厨师发送食材后,可以返回食材的名称.

3.25.2 API描述

3.25.2.1 call()

3.25.2.1.1 功能:多线程执行从食材工厂发送食材给厨师

3.25.2.1.2 参数:

无,食材的名称在初始号食材工厂时就已经决定

3.25.2.1.3 返回值:

食材工厂生产的食材名字

3.25.2.1.4 使用方法:

```
1 FutureTask<String> ft = new FutureTask<>(gen);
2 new Thread(ft).start();
```

gen为一个食材工厂对象.

3.26 immutable

3.26.1 设计模式含义

在多线程编程中,保持对象在一时间内只能被一个进程读取非常重要.如果一个对象在其生命周期中不需要变化,那么设置其为不可变是一个保证并发安全性的一个有效方法.

3.26.2 API描述

3.26.2.1 Generator(String name)

3.26.2.1 功能:

该函数其实是食材工厂的构造函数, 其中name赋值之后是一个final String, 一经定义就不可修改,保证了并发的安全性.

3.26.2.2 参数:

name:食材工厂生产的食材的名字

3.26.2.3 返回值:

无

3.32 memento

3.32.1 设计模式含义

备忘录模式常用于保存一个对象的一种属性的状态。

在undercook中，需要保存一些游戏的设置使得这些数据长久保存于本地的配置文件中。比如，Menu类保存游戏的音量设置。

3.32.2 API描述

3.32.2.1 Save()

3.32.2.1.1 功能:

保存此时的音量信息.

3.32.2.1.2 参数:

无

3.32.2.1.3 返回值:

无

3.32.2.2 Volume(int delta)

3.32.2.2.1 功能:

根据delta的值改变音量,如果delta小于零,为减少.如果delta的值大于零,为增加.

3.32.2.2.2 参数:

delta:音量的变化

3.32.2.2.3 返回值:

无

3.40 publish subscribe

3.40.1 设计模式含义

发布订阅模式让发布者发布的消息可以传达到订阅者手中

undercook中,食材工厂一旦生产出食材,就要告知所有的厨师,让他们做好准备

3.40.2 API描述

3.40.2.1 addCook(Cook cook)

3.40.2.1.1 功能:

添加一个厨师到食材工厂维护的厨师列表中

3.40.2.1.2 参数:

cook:一个厨师对象实例

3.40.2.1.3 返回值:

无

3.40.2.2 deleteCook(Cook cook)

3.40.2.2.1 功能:

从维护的厨师列表中删除一个厨师

3.40.2.2.2 参数:

cook:待删除的厨师实例

3.40.2.2.3 返回值:

无

3.40.2.3 notifyAll(int max)

3.40.2.3.1 功能:

通知所有的厨师,生产一个食材

3.40.2.3.2 参数:

max:使用的最大线程数

3.40.2.3.3 返回值:

无

3.47 thead pool

3.47.1 设计模式含义

基础的多线程编程存在很多难以解决的问题,其中一个是控制最大并发数.

undercook在食材工厂通知厨师方面,可以选择最大的并发数

3.47.2 API描述

3.47.2.1 notifyALL(int max)

3.47.2.1.1 功能:

通知所有的厨师,一个食材被生产出来了

3.47.2.1.2 参数:

max: 限制的最大并发数

3.47.2.1.3 返回值:

无