业务层相关的技术（一）

Web 层 + 业务层 + 持久层

业务层就是软件的核心，也是我们重点任务。Web应用开发也对业务层有相关的技术来提高业务层开发的效率。Java EE对业务层也采用容器+组件的架构，主要有两种：一种是基于经典的Java EE技术，即EJB技术，采用EJB容器+EJB组件，比如说：GlassFish（EJB容器）+EJB组件；另一种采用轻量级的技术，例如Spring容器+Bean组件（甚至可以是POJO），开发业务组件时，比较容易。

一、为什么要采用容器+组件的技术，它带来的优点是什么？

来看一个日常生活的例子。

比如说：一个国家的经济生活中，银行（金融）的作用，它是关系到国民经济的血液。

对于一个小型的社交范围

存款人1 借款人1 存款人2 借款人2

人们之间可以直接形成借贷关系，因为人员比较少，在管辖的范围内。

但是对于大型（巨型的）社交范围。

可能会形成复杂的借贷关系网络，不便于管理，所以一般在借贷之间产生一个中介（银行）

银行

同样，对于企业级的面向对象的软件系统，有很多的业务对象，这些业务对象是交互的。

所以，为了便于管理这些对象之间的交互，软件开发实践中，也采用一种中介（容器）来管理。

容器

云应用（私有云）的微服务

二、怎样实现业务层的容器+组件

比较流行的技术采用的 依赖注入 。

软件开发的基本原则，常说有五大原则：

SOLID

其中O- OCP原则

示例，简单的游戏开发。

设计方案：

采用名词/动词法（这是面向对象程序设计的一种方法）来设计

1.找出名词

**角色，怪物，HP，武器，木剑，铁剑，魔剑**

2.对这些名词进行分析，

（1）名词在系统中是否是独立，还是另一个名字的属性（状态）。这个示例中，可以看出

HP不是一个独立的概念，它是怪物的一个属性。

（2）尽量形成一个整体的概念体系

从这个示例中，可以看出**武器与木剑，铁剑，魔剑形成继承体系，也就是说武器是**

**木剑，铁剑，魔剑的抽象。**

更准确地说：可以形成这样的概念结构：

攻击服务（接口）

实现（提供）

**木剑， 铁剑， 魔剑**

经过分析之后，就可以定型系统中的类。

2. 找出动词

**攻击，损失，死亡，装备**

（1）分析这些动词。

是不是某个类的操作

某个操作是不是某些类的共同具有的。考虑是否把它定义为接口？