如果我们有了maven之后，程序员的生活会是下面这样的。。。。。。

1、每天过来上班，先写代码，然后写单元测试，接着运行一个maven命令，只见哗啦，哗啦，哗啦，成百上千个单元测试就这么自动运行了，自动就出来一份完整的单元测试覆盖率的报告，以及单元测试运行错误的一份报告

2、如果一个系统有多个工程，那就用maven来把多个工程集成在一起

3、对于依赖，不用自己手工管理，简单用maven配置一下依赖，比如说你需要spring mvc，spring，mybatis几种依赖。然后用一个统一的父工程约束所有工程的依赖版本，所有的依赖下载、版本调解、版本升级等繁琐的事情全部由maven自动搞定

比如说你的某一个依赖，spring还依赖了一个log4j，maven会自动给你再下载一个log4j，不用你自己去管，还缺失了哪些依赖

不再需要自己各种手工上网下载jar包，放到lib目录下，解决各种jar缺失，jar包冲突

4、编译+打包+发布？不用自己手工搞了，简单运行一下maven的命令，只见maven哗啦，哗啦，哗啦，就给你编译好代码，按照规定的格式打成发布包，然后直接连接到对应环境的web容器，给你自动发布了上去

5、如果要对数十个工程统一进行编译、打包和整合，发布，一个命令，所有工程自动化给你搞，自动化编译、打包，所有的包给你整合在一起，最终的一个发布包就一键部署

简单来说，用了maven之后，在依赖管理、构建管理、模块化拆分管理，全部自动化完成！

maven的前身

1、make：最原始的构建工具，不能跨平台

2、ant：曾经没有maven的时候，流行过一段时间，但是手工配置的语法繁琐，而且需要一次又一次的重复，另外依赖管理还要借助ivy来完成，工作量还是有点大。。。

3、maven：自动化。。。，目前是最有影响力的工程管理工具

4、gradle：google发布，不再基于xml来进行配置，而是基于DSL语言（领域特定语言）来进行构建管理，语法功能更加强大，android这块用的较多，同时国外一些开源项目，比如spring也开始用gradle来管理，国内也有少数公司在尝试使用

5、说句实话：未来也许是maven和gradle并存的一个事情，但是近几年，主要还是maven

用轻松的语言给大家描述了一下，实际上maven里面包含的东西是很多的，接下来，就用完全符合企业实战应用的场景模式，来给大家一一讲解maven的各种功能，每个功能都会带出来在公司里是在什么场景下使用的。

提一点，很多视频课程，完全参照着书本方式去组织课程内容和讲解的

语言，文字

文字，不需要语言

文字的意义在于说，可以用比较少的信息量，用比较精炼的方式，以一个结构化的思维把一个知识给你组织清楚，告诉你怎么回事

写书，看书，看书，有局限性，没有办法说真正生动的，甚至是按照实战场景的方式给你去讲解是怎么回事

导致很多人看书没有兴趣看下去，枯燥，看不太懂

视频，语言，有情感，有情绪，我调动一下大家的积极性

然后有综合性的展现方式，可以打字，说话，画图，写代码，讲解，360度立体式的这么一个东西853769620