《手册》泰山版新增了错误码相关规约，比如：“错误码的制定原则：快速溯源、简单易记、沟通标准化”；“错误码不体现版本号和错误登记信息”；“全部正常，但不得不填充错误码时返回五个零”；“错误码使用者避免随意定义新的错误码”；“错误码不能直接输出给用户作为提示信息使用”；“错误码的后三位编码与 HTTP 状态码没有任何关系”；“错误码即人性，感性认知 + 口口相传，使用纯数字来进行错误码编排不利于感性记忆和分类” 等。

那么这些规约对我们有啥启发？

2.1 2w1h

我们学习某个知识的时候可以尝试使用 2w1h 这种学习和思考方式。

学习错误码，要思考为什么要使用错误码，即错误码的目的。要清楚错误码是什么，即错误码的本质。

搞清楚了是什么和为什么，那么怎么做就容易多了。

注释是为了帮助阅读源码的人快速、清晰、准确地理解源码。注释的规约都是围绕这个目的展开的。

错误码则是为了 “快速溯源、沟通标准化”。错误码的其他规约都是围绕着这个目的而设计。

2.2 以终为始

我们做事情、学编程等都要抱着 “以终为始” 的理念。

规约中规定 “错误码不能直接输出给用户作为提示信息使用”，那么为什么要有这样的规定呢？

错误码一般都是数字或者数字加字母的形式，对编程人员排查问题非常方便，但是对于用户而言非常不友好。

因此即使要给最终用户展示，也要展示最终用户可以理解的文字形式。

2.3 Do More

不知道大家是否思考过这些问题：**《手册》中给出了错误码的诸多建议，那么已经非常完善了吗？已经解决了所有痛点了吗？**

比如有些公司错误码的注册和使用是分离的，即注册时在页面上填充错误码的各种属性，使用时要手动定义错误码枚举。那么是否可以将定义和使用关联在一起呢？

我们是不是可以定义好错误码之后可以提供自动生成 jar 包的功能，而不需手动定义？

比如当你拿到错误码时，我们是否可以直接根据错误码来搜索其相关联的服务、常见原因，关联过的故障等信息呢？

书中提到 “错误码使用者避免随意定义新的错误码”，建议尽可能在原有错误码附表中找出相同或者相近的错误码。可是实际使用时又该如何落地？

我们是不是可以设计一个根据错误码的描述智能匹配出一些可复用错误码的系统呢？

本文重点讲解新增的错误码规约给我们带来的启示。包括专栏中一直提倡的 2W1H，以终为始和 Do More。

希望大家在未来的学习和工作中，可以实践本专栏所传达的学习方式，能够从更宏观更本质的角度来思考和解决问题。

1、你们公司的错误码的格式是怎样的？有什么优点和缺点？

2、回忆下你从专栏中学到了哪些有用的学习方法?

}