一份从 0 到 1 的 Java 项目实践清单

2017-12-22 ImportNew

（点击上方公众号，可快速关注）

> 来源：等你归去来 ，
> www.cnblogs.com/yougewe/p/7749444.html

虽说工作就是简单的事情重复做，但不是所有简单的事你都能有机会做的。

我们平日工作里，大部分时候都是在做修修补补的工作，而这也是非常重要的。做好修补工作，做好优化工作，足够让你升职加薪！

但是如果有机会，去尝试些自己平日里少做的事，我觉得是一件值得庆幸的事。

前段时间，接了个新项目。只有一些idea在业务需求方脑海里，然后就开始需求讨论，然后就开始做事了。项目不复杂，但是由于是用JAVA语言实现（这相对来说是我的薄弱点），对我个人显得比较有意义。

总结下来，其实也就是一个项目清单。个人觉得还是有点意义吧，给没有一定全面实践的同学参考吧！

1. 项目规划

1.1 首先，你得彻底明白到底要做什么？

这个过程，可能是你要读需求一遍、两遍、三遍。。。 然后假设，你已经在使用这个产品了。

1.2 其次，明白需求后，就要进行整体框架的构思！

比如用什么呈现给用户，用什么来存储数据，需要些什么样的系统等。

在这个层次上，一般都会遵循公司的规定，然后再根据项目本身需求，做些相应的调整。

我们在这个项目里的整体框架为：前端使用 APP(ios&android)、H5进行用户界面呈现 =====>> 接入网关进行数据加解密，流控转发等 =====>> 第一层API服务，接受客户端请求，做简单业务检验组装 =====>> 第二层核心业务SERVICE服务，进行核心业务处理，如写库、调用第三方接口等 =====>> 最下层基础服务，提供单一的功能服务，如消息服务，订单服务。

前期只提供APP，因此不存在单独H5调用API服务的情况，但是H5的应用场景仍然存在，此时的H5地址，由服务接口提供地址返回到APP进行webview加载。

1.3 人员规划

项目整体框架出来后，得要有人去实施才行。

这里一般需要遵循一个最小原则，即划分出的人员，尽量做到能够独立完成自有的模块，而不是一定要依赖于另一方的实现才能进一步。比如 android,ios各一人，API与SERVICE可以多个人，但是都要让其有全部权限，因为API与SERVICE有强依赖，脱离一方，将无法独立完成。基础服务各自安排相关人员实现。最后进行联调即可。

1.4 时间规划

有了人员之后，也不能无限时间的去做事。肯定是要规划的，否则没有压力也没有动力。项目不知何时才能结束。订时间计划一定要去询问当事人，要多少时间，尽量站在专业的角度给出合理的建议和评估。促进项目的完成。

2. 框架规划及搭建

2.1 有了整体框架的构思后，就要细节到每个层次的实践了

因为都是应用的分层，所以，不可能有统一的描述，只能是针对每个应用层。做自己该做的事。如 android/ios 有自己的开发框架；h5有自己的开发框架(因为很多应用场景可能涉及到h5与app原生的交互，所以即使功能简单，也尽量利用一些已有的框架进行开发)。

而服务端，虽分为多层应用，但是应尽量使用同一门语言，利用同一套开发框架，自己公司有研发框架自然最好，没有也尽量利用统一的开源框架。这样做的好处是，当有人员变动时，可以立即熟悉其代码及应用场景，从而增加适应性和管理性。

针对服务端的框架，我觉得有必要多说点。因为整个应用运行的流畅性，可靠性，准确性，都是由服务端来决定的。虽然用户看到的是APP或者H5，但是可以说，服务端才是应用的核心。所以，服务端要做的事情自然很多了。

2.2 怎样搭建好一些服务端的框架呢？

首先，框架类的东西，自然是要提前学习的。但是，就目前市场行情来说，要想利用框架应该都是比较简单的，尤其是公司内部提供的框架，一定要有demo。这样，照着demo，一步步调试，直到整个应用接通；

删除不需要的模块，添加特别需要的模块，保证在具体开发过程中，能够想利用啥就有啥可利用；

充分了解框架需要的一些配置参数，知道事务从哪里来，到哪里去？这里，应有一个配置中心与之对应，但是自己得清楚。

使用一个顺手的IDE工具，不是说你技术不够牛逼，而是一个好的工具，能够让你事半功倍。（其实能够多背点套路，也不一定非要体现在正式项目上）

写出第一个可供使用的接口服务，可以说，第一个永远是比较重要的。因为，第一个的思路，就是你后续所有功能的方向，因此，写好第一个”hello, world.”；

3. 开发环境的搭建(服务端)

3.1 其实这项工作是及其重要的，之所以把它放在第三点，是因为，没有代码作铺垫，开发环境搭了也没用。

3.2 开发环境的搭建，主要也是服从于整体框架的构思。

主要包括，需要多少个服务，需要多少台服务器，需要多少个基础应用，需要多少个基础配置等等。

当然，开发环境本身就是一个很大的难题，一般还是交给专业运维几十年的老司机来完成了。自己就当作了解得了。

目前的项目开发，除一些小规模公司还在利用一套服务端代码，干完所有的事外，大部分应该都是多个应用的配合完成。而测试环境，不太可能利用多个服务器提供服务。因此，使用docker进行测试环境搭建尤佳。建立多个docker进行多个服务器模拟，也算是和线上环境保持一致了。

目前的主流技术得用上（当然关键还得看你的框架规划），zookeeper, dubbo, redis, mongo, mq, ...

3.3 只有开发环境搭建好了，才能让后面的流程无忧。搭建的过程一定是，又搭建，又改代码，又排错...

4. 进度的同步

4.1 及时向领导同步项目进度

对于一个新项目，有些地方行动缓慢是很正常的。而部分开发同学（比如我自己），就喜欢沉浸在自己的小世界里纠结，走不出来，从而忘却向领导汇报工作。而作为一个有点同理心的领导来

说，他又不愿意实时都来盯着你做事，因为也怕你遇到困难，想多给你点时间解决。但是，这种情况，开发同学自己其实是要吃亏的，因为，给外人的感觉就是，你啥都没做。所以，解决问题的同时，也不忘向领导汇报。

4.2 有处理不了的问题，及时向大牛们或者领导请教

独立解决问题是好事，但是千万别过了头，实在解决不了，就要及时请教。否则，浪费的是时间。进步最快的方式，莫过于向比自己牛逼的人请教。知之为知之，不知为不知！

4.3 尽量将问题分摊下去

问题肯定是有的，而且会很多。千万不要把所有的事情都压在自己这儿，那样自己会累死的，而且项目进度也会因此变得缓慢。要多利用小组成员的各自优点，适当多让其搞点事情。

工作永远都不是单一的一件事，肯定还会有其他的事情插入进来，观察事情的重要性解决。如果能够让其他同学解决的，尽量让其他同学处理，这点也得与领导同步。否则分心过于利害，受阻的只有项目进度，延期可不是自己一人的事情了。

需求也不可能一下就是完善的，在做的过程中，才可能发现一些潜在的问题，这时及时与需求方沟通，保持高效的状态。当然，后期的跟进，也是尽量做到不要一人大包大揽，而是相应的人就去负责相应事情的跟进。其他人只要知道结果就行。

5. 功能模块的完成

5.1 说到具体的业务实现，个人觉得，已经不那么难了。不过就是，先尽力提出的一个初稿，然后发现问题解决问题，发现问题，解决问题的过程。

5.2 各自系统能做的事情完成后，就是联调各系统间的调用关系，保持高效的沟通，让问题在短时间内解决，尤为重要。在这种时候，我觉得，一个小黑屋也许也是个不错的选择。

5.3 联调的过程，其实就是一个自测的过程，应把尽可能多情况给考虑到位。

5.4 代码检查，自己开发的代码，基本上很难发现其中的问题，即时找到相应人帮忙检查代码，是比较好的解决代码问题的方案。其实，在给别人检查的时候，也是自己检查的时候，相当于自己再一次的开发，也能及时发现问题。

6. 多轮的测试验证

6.1 测试同学，其实在开发快结束的时候，已经把测试用例给到大家。这也是另一个角度的开发，因此，参考测试用例进行相应开发修改也是很有必要的。

6.2 第一轮测试，可能主要是大功能的验证，小功能的检查，挡板环境即可，无需真实环境。

6.3 第二轮测试，则是要把之前的测试及各种配置，全部清空，以一个全新的项目来对待，重新进行相应环境搭建，代码部署，然后再进行测试，确保问题解决后，做好了相应的处理方案备份。这时，就需要用到真实的应用环境了。对之前一些暂未解决的问题进行重新测试。确保无问题。

6.4 第三轮测试，应该是一个灰度发布的环境，也可以认为是预上线。将所有环境当作是线上来处理，如果运行ok,即可准备发布上线了。

6.5 在测试过程中，因测试人员只是人工的处理，有时不一定能捕获所有的问题，开发在这时，也应站在测试的角度，发现问题，即时监控，即时处理。

6.6 自动化测试，这个其实应该是靠后的处理，但是如果能做到这些的话，也能够快速的重现问题。

6.7 压力测试，应对线上环境，需有一定的能力评估，不然，只瞎猜，恐怕也不是好事。随时准备横向扩展，也只是出现问题后的解决方案。做好压测，发现代码中存在的问题，即时处理掉。

7. 外围处理(上线前)

7.1 上线前，肯定是有很多事务要处理的。

- 测试环境中的各种基础数据，随时导出备份，到线上时，直接插入使用；
- 服务器，在架构评审过程中进行数量评估；
- 域名，对外网提供服务一定是要域名的；
- 权限，比如上线后，出现了问题，谁有权限来处理问题，一定提前给到；
- 验收，这是关键的一点，功能完成后，及时验收，如果上线有些小问题，尽量协商，不要在线上频繁改动。

如此！

整个项目就完工了。

其实发现，一个项目真正的功能实现，并没有占多大的比例，而是一些前期的准备及后续的处理，反而占了更多的时间。

第一个版本上线后，可能接着就是迅速迭代了。（如果运营还可以的话！）

以上，就是一整个项目的流程清单，以一步一个脚印的经历总结，不涉及具体语言代码，但是思路都是相通的，希望对你有帮助！