从啥也不会到可以胜任最基本的JavaWeb工作，推荐给新人的学习路线（二）

在上一节中，主要阐述了JavaScript方面的学习路线。先列举一下我朋友的经历，他去过培训机构，说是4个月后月薪过万，虽然他现在还未达到这个指标。

培训机构一般的套路是这样：先教JavaSE，什么都讲一讲，可是都讲不透，基础差一点的只有越听越糊涂，被面向对象的概念，接口，抽象类，搞得头都大了。一天到晚盯着几个干巴巴的案例看。

JavaSE大概持续讲两个月，最后会让你做一个控制台的小屁项目，可能是ATM管理系统，也可能是图书管理啥的。

然后，开始讲html、css，这两样东西一般就是1到2天，讲了等于没有讲，让他做一个小网站的前端页面根本不可能，他只会用div画几个五颜六色的框框。定位也不会，总是凭着老师课堂上讲的那几个例子，写写 position:absolute。然后郁闷得一塌糊涂，这个div怎么就移不过去呢，怎么就是不动呢？？为什么老师的代码可以，我写的代码就不行？？去问老师，老师说，我们是Java培训，这个是附带的，没关系，你们到时候也不会写这种代码，你们是写Java的。（呵呵了）

不管那么多了，接下来几天马上学习JavaScript，第一天，着重讲了讲var是什么，js的几种基础数据类型。然后讲function，alert怎么用。大概3到5天的样子吧，JavaScript就讲完了。

最后一天用来讲jQuery，发现好神奇啊，可是$是啥，为什么会这样，就不知道了。只会写写 $("#id")这样的代码。可是不管怎么样，jQuery已经讲完了。懂不懂是你的事。

我朋友直到毕业，都不知道json是啥，ajax是啥？就最后两个礼拜讲了SSH，带了一下。

JavaScript讲完后，立刻开始数据库，数据库一般还是教的Oracle。第一天，啥也不做，Oracle太大了，也不教你怎么安装，反正就告诉你老师的主机已经安装好了，那个培训机构的电脑可承受不起oracle。数据库教了1个礼拜，什么都涉及一点，包括触发器，但是没有存储过程。反正学完以后，我朋友只会 select * from student了。left join,right join 没看懂，只记得老师花了半天时间解释啥叫笛卡尔积，什么是三大范式？好像很厉害，然后... ...不懂。

接下来就是tomcat，JSP，这段时间对他来说就是噩梦了，反正最后差点弃坑。最后半个月，终于，万众瞩目，SSH闪亮登场。OK，培训结束，去月薪过万吧。

因为关系比较好，我和他经常保持联系，后来一段时间，他去了外包。因为太辛苦，后来又辞职了。

说了这么多，我在此提出质疑，个人认为，这种学习路线绝对有大问题。对于一个没有什么编程基础的人，这种学习路线肯定会害了他。我个人推崇的路线应该是这样：

第一步：不学任何后台语言，从Html + css开始。

这个阶段，你可以花费一到两周的时间，看各种视频外加各种操练。最起码，你得有一些绘制网页的能力吧。虽然如同easyui的框架非常流行，可是你起码也要有点这方面的基础吧。用户的需求千变万化，你不可能永远依赖easyui的。你不要跟我说什么bootstrap，如果你连div＋css都不懂，bootstrap你真的能用得溜吗？在我现在看来，bootstrap只是一种工具罢了，一些前辈把

很多经常用到的代码封装起来，方便自己和他人使用，仅此而已。你不要觉得有多么难，毫无疑问，这种工程，我想最开始的时候肯定是一个人做的，后来随着名气越来越大，可能会有一个团队来维护。但是，最早最早的时候，肯定来源于一个人的想法和思考。

第二步：深入学习JavaScript。

JavaScript轻巧，灵活，只需要一个浏览器，就能立即看到效果，没有环境搭建方面的疑虑。另外，JavaScript兼具面向对象语言的各种特点，再加上现在前端这么火，你学习JavaScript吃不了亏。当然，我说的学习，不是泛泛而学，而是有目的地去深入学习。仔细体会其中的乐趣和奥妙，最起码，你得达到能够自己封装一个弹窗组件的程度。

我学习JavaScript后，再学JavaSE，真的就是一个水到渠成的过程，每学到一个地方，脑海中立马就闪现 “这不就是JavaScript中的XXX吗？”的念头。最好的学习方法就是类比。记得牢，而且可以融会贯通。就好像玄幻小说里面描写的那样，当你对某一个技能或者法则达到了很高深的境界，那么对于其他任何流派就都能融会贯通了。

第三步：突袭JavaSE

为什么说突袭JavaSE？因为我现在明白过来，对于JavaWeb开发而言，没必要对JavaSE做太多的深入。如果你的JS功底足够，那么JavaSE对你来说就太简单了，因为JS比Java难入门得多。以下的学习清单是我整理所得：

理解什么是类？

你不要去管什么面向对象还是不面向对象了，我认为面向对象这种概念很难用语言描述清楚，更多的是当你代码写多了，产生的一种直观的觉悟。你是去查资料也好，看视频也罢，先理解什么是类。然后自己尝试着写几个类。你不要去写什么Person类，Animal类了，有什么意思呢？随便打开一个网页，比如百度贴吧，你有百度账号的话，你就想想，一个百度账号会有哪些属性？用户名，密码，头像，是否可用，等等。用这些信息来写Java类。

一个Java类，无非就是属性和方法，大部分情况下，方法无非就是用来给属性赋值的。属性是干嘛用的，不就是用来存储数据的吗？你说对不对呢？Java有八种基本数据类型：分别为整型 int，短整型 short，长整型 long，字节型 byte，布尔型 boolean，字符型 char，单精度浮点数 float，双精度浮点数 double。

就比如说int，Java编译器规定int占四个字节，也就是4个byte，一个字节占8位，一个位就是一个bit。bit是计算机中最小的单位，它只有0和1两种状态。我们常说一个文件有多少兆，这个兆就是MB，1MB有1024KB，1KB有1024个字节。当你定义了一个int类型的变量，在运行的时候就会在Java虚拟机中申请一个4个字节的空间。1MB有1024KB，1KB有1024个字节。所以说1MB可以存放 （1024 * 1024 / 4） = 262144 个int变量。Java虚拟机的默认内存是64MB，所以最多应该能存放16777216个int类型的变量。当你定义一个int类型的变量，那么运行的时候，虚拟机的剩余内存就会被减掉4个字节。所以，属性是干嘛用的，我们在写Java类的时候，为什么要定义属性。我觉得没有别的含义了，定义属性就是为了存储数据的嘛。

我们写一个

private int a;

Java虚拟机（JVM）跑起来，一旦我们new了这个对象。这个a变量就会被放到JVM的内存中，然后JVM就会专门开辟一个空间，来装载这个数据。然后，我们才可以在计算机中操作这些个数据。你总不可能说，我有一个数字100，就要计算机对这个数字进行加减乘除的运算吧。计算机怎么知道这个事情呢？你是不是必须要告诉计算机有一个数字100，它才会知道？

为了装载这些数据，所以才有了八种基本数据类型，每一个数据类型就好比一个篮子，有的篮子大一点，比如long类型，可以放好长好长的数字。有的篮子小一点，比如byte类型，只能放一点点大的数字。Java类，我的理解就是一个模板，因为我学过JS，所以我会类比。但是这些类比，仅限于我个人，在此就不多说了。

Java类，，我更愿意把它称为一个 数据模板。它只是一个模板而已，不是一个实实在在的对象，这一点首先要确定。就好像工厂生产一个产品，首先是不是要有一个模板和设计图纸，这个模板决定了你这个产品是一种怎样的形状，以及可能会具备哪些功能？图纸则决定了功能的具体实现。比如生成一部手机，模板开出来就是一个扁平的长方体的样子，可是光有模板还不行，你还得规定它的一些具体细节。这些细节就好比是Java类的构造方法，以及其他的一些方法实现。

但是，你光给客户模板和图纸行吗？一般来说是不行的。

至于静态方法，我们知道，我们调用静态方法的时候，不需要先生成一个实例，可以通过类名直接调用。这就相当于，在弄模版的时候，这些功能就已经定制在里面了。你买手机的时候，里面不是经常有一些内置的应用吗？有些删都删不掉，这不就相当于静态方法吗？

（我只是举一个例子啊，你不要非得较真说我可以ROOT一下啊）

如果模板里面已经有了一些做好的功能，今后任何根据这个模具生成出来的产品也自带了这些功能。如果模板里面已经做好了一些功能，那么我的确可以使用这个模板，而不需要真正拿到一个产品。比如生产一部手机，它的模板里面已经做好了一个手电筒的功能，那么，你即便不给我一个真正的产品，仅仅给我一个模板，我是不是也可以用它的手电筒功能呢？

这就是静态方法。所以我们常说，静态方法和静态属性为所有实例共用，不就是这个道理吗？

所以，正常情况下，我们调用一个类的非静态方法，是不是必须要先new一个对象？

2.基础知识

这部分包括：包的知识，java中数据类型的分类（值类型，引用类型），继承以及继承的特点等。快速过一遍就行了，不要去深究，就算你深究了其实也没啥用。以后代码写多了，自然就会烂熟于心。

3.数组和其他一些常用类

数组的话，会用即可，不要深究。其他常用类，比如Date类，最好都去学一下，但是同样不要深究。

4.集合

集合框架，只需要掌握两个，一个是ArrayList，一个是HashMap，知道怎么用就行了。其他集合都可以不去管它，你不要老是觉得，我学这些会不会不够啊？等你以后成为了老司机，还不是学什么会什么?现在功利一些也无妨。有余力的，去研究源码，没有余力的话，不研究也没事。

对于ArrayList，我写过两篇文章，还差一篇才完结，以后补上。

5.泛型

能看懂 List< Map< String,Object > > 是啥意思即可，其他都不要管。你又不去写框架，就算真有这种雄心壮志，也是若干年后的事情了，现在你的当务之急还是找一份工作。

6.设计模式

不要管，最多去学一下单例和工厂。其他的暂时不需要考虑，刚工作那会，就是做一些增删改查，先找到工作，稳定下来再去提升。

7.IO流

其实这部分真无所谓，还不如直接学文件上传。即便是文件上传，一般公司都有现成的类给你调用，也不需要你自己去写。所以，IO流，不学都没关系。

8.Awt，Swing

有多远，扔多远，不要问我为什么。有精力学这个不如去学SWT，虽然99%你学了还是用不到。Swing有啥用，我能想到的只有去网上接一些课程设计啥的私活了。

9.反射

不需要，你又不去写框架，暂时不必关心。

10.新特性

不需要，不是你初学者该考虑的。等你以后工作稳定了再说吧。

11.JDBC

这个，你仅仅需要知道如果导入驱动包，在项目里面成功拿到链接就行了。其他都不用管，最原始的JDBC1.0规范，不学也没事。还不如重点去学一下Mybatis。关键在于，你要学会配置文件怎么配。至于SQL，顺带着学一下就行了。

再说一遍，直接去学MyBatis之类的现成框架，不要再JDBC上浪费太多时间。

我相关的文章：

http://www.jianshu.com/p/8944864f5161

JavaSE，我认为初学的话，能做到这样就可以了。毕业标准就是自己能够做一个完整地增删改查出来。你会测试就行，比如用JUnit测试，不要去写那种什么控制台项目了，有什么意思呢？直接写DAO层，然后体会一下其中的流程即可。

学完JavaSE，就是JSP、Servlet，这一块要相当用心地去学。它是基础，我个人认为比框架更重要。这些内容会在下一篇中讲到。