## 基础篇

1. java基础知识，这个太多了，自己随便找吧，例如《java语言程序设计》，知识点：基本语法，数据类型，类，面向对象,io，线程，网络等。
2. 数据库相关。知识点：表，字段，常用SQL语句。mysql,oracle的安装，使用程序连接DB，增删查改操作等。
3. 中间件，安装tomcat,配置启动tomcat
4. IDE,知道使用eclipse，创建，发布，启动工程。熟悉常用的快捷键，此链接可学习[http://www.open-open.com/bbs/view/1320934157953/[http://docs.alibaba-inc.com:8090/images/icons/linkext7.gif](http://www.open-open.com/bbs/view/1320934157953/)](http://www.open-open.com/bbs/view/1320934157953/)
5. linux操作系统的安装，熟悉常用的操作命令，可以尝试在linux安装mysql或者oracle等。
6. maven，安装配置，基本知识（编译，坐标，打包等），尤其要掌握怎样是用dependency解决jar包冲突的问题。
7. html,javascript,css，基本语法，写个程序在browser上跑跑。

## 进阶篇

1. jvm ，了解jvm的内存分区，jvm的内存模型，gc机制，类加载机制等，自己能够调整jvm参数，能够根据dump或者其它文件进行程序性能分析，重点推荐《深入学习java虚拟机》
2. 多线程，理解锁，并发，线程池，新的juc包里提供的一些线程类，原子类等，自己能够编写多线程程序，重点推荐《java并发编程》
3. NIO，熟悉nio模型，nio的基本原理，java中的nio的一些类，自己去写一个基于nio的server和client程序。了解mina或者netty。重点推荐《Netty权威指南》
4. 框架，优先学习spring,ibatis,其次是struts，了解spring 的aop,ioc的原理，spring,ibatis的配置和常用的用法，当然推荐看下spring源码（优先级低）
5. 数据库，数据库悲观乐观锁，数据库的主备分离的配置，[分库分表以及tddl的原理与使用[http://docs.alibaba-inc.com:8090/images/icons/linkext7.gif](http://gitlab.alibaba-inc.com/middleware/tddl5-wiki/wikis/home)](http://gitlab.alibaba-inc.com/middleware/tddl5-wiki/wikis/home)。数据库本身的性能优化，SQL性能优化（尤其要知道怎样看执行计划进行优化）
6. 远程调用，了解[dubbo](http://dubbo.io/)[[http://docs.alibaba-inc.com:8090/images/icons/linkext7.gif](http://dubbo.io/)](http://dubbo.io/)、[hsf](http://gitlab.alibaba-inc.com/middleware/hsf2-0/wikis/home)[[http://docs.alibaba-inc.com:8090/images/icons/linkext7.gif](http://gitlab.alibaba-inc.com/middleware/hsf2-0/wikis/home)](http://gitlab.alibaba-inc.com/middleware/hsf2-0/wikis/home)的实现方式和原理，写demo对比两者差别。
7. 异步消息，例如metaq，了解其原理，要知道怎样提高可靠的消息服务，以及消息风暴防止的问题，尤其要知道幂等性的控制（特别是在分布式系统中怎么做到幂等性的控制）
8. **数据库事物，事物的隔离级别，分布式事物，了解XTS，实现的原理以及使用。**
9. **Hbase,**[**tair[http://docs.alibaba-inc.com:8090/images/icons/linkext7.gif](http://baike.corp.taobao.com/index.php/CS_RD/tair)**](http://baike.corp.taobao.com/index.php/CS_RD/tair)**,本地缓存了解其实现的原理，然后在项目中能够运用起来。**
10. **测试分析，了解测试分析的过程以及常用方法论，自己可以去操练下，知道怎么写测试用例，推荐看下这个视频(**[http://atit.alipay.net/index.php?r=courseDetail/index&courseid=639[http://docs.alibaba-inc.com:8090/images/icons/linkext7.gif](http://atit.alipay.net/index.php?r=courseDetail/index&courseid=639)](http://atit.alipay.net/index.php?r=courseDetail/index&courseid=639)**)**
11. **junit测试，了解单元测试，并在项目中使用junit测试，使用mock测试技术（具体技术不限，推荐easymock），另外推荐jacoco工具，在eclipse查看实时查看覆盖率。**
12. **负载均衡的原理，**[**LVS篇[http://docs.alibaba-inc.com:8090/images/icons/linkext7.gif](http://www.atatech.org/articles/1437)**](http://www.atatech.org/articles/1437)
13. **数据层之**[**DRC[http://docs.alibaba-inc.com:8090/images/icons/linkext7.gif](http://tbdocs.alibaba-inc.com/display/DS/Home)**](http://tbdocs.alibaba-inc.com/display/DS/Home)
14. **更多：[阿里中间件列表](http://baike.corp.taobao.com/index.php/MiddleWare)[[http://docs.alibaba-inc.com:8090/images/icons/linkext7.gif](http://baike.corp.taobao.com/index.php/MiddleWare)](http://baike.corp.taobao.com/index.php/MiddleWare)，包括hsf、tddl、diamond等；**[**阿里中间件git地址[http://docs.alibaba-inc.com:8090/images/icons/linkext7.gif](http://gitlab.alibaba-inc.com/groups/middleware)**](http://gitlab.alibaba-inc.com/groups/middleware)

**进阶篇优先学习项目中用到的，其它的可以先留个映像，慢慢再学。**

# 内功篇

1. 系统设计，对系统设计有个初步的认识，怎样识别业务用例，然后怎样转变为系统用例。在大型系统中，怎样划分业务边界以及系统边界，大到一定程度的系统怎样运用领域设计的思想，设计常用工具的使用（推荐astah工具）。推荐看下鲁肃的视屏([http://atit.alipay.net/index.php?r=courseDetail/index&courseid=288[http://docs.alibaba-inc.com:8090/images/icons/linkext7.gif](http://atit.alipay.net/index.php?r=courseDetail/index&courseid=288)](http://atit.alipay.net/index.php?r=courseDetail/index&courseid=288))，最后看看他推荐的书籍
2. 设计模式，了解常用的设计模式，并可以在自己的代码中将这些模式应用进来，推荐《head first 设计模式》
3. 代码重构，学习代码重构常用的思想和方法论，推荐《重构 改善既有代码的设计》。

**内功篇是需要不断的投入时间去学习和研究的，有些东西刚开始不理解，过了一段时间之后，会慢慢明白的。**

怎样做测试

A.重视投入产出比

1.功能点的风险：更多的站在业务角度来思考

2.功能点的复杂度：

3.对于不同的功能点，可以制订采用不同的测试设计方法，边界值/等价类/判定树/功能交互分析/测试类型分析

B. Don’t repeat your self

DRY原则

C.不断的学习和发现被测对象

D.过程改进/实践

一位蛋疼人士说过，一个合格的tester应该是名业余QA

知道并使用不同的测试方法，比如基于模型的测试，探索式测试，用户界面测试(UI Testing)；

在一些领域有深入的测试知识，比如性能测试，网页(web)，手机应用测试，安全性测试（或一个安全专家），云测试；

熟悉一些测试工具，比如JUnit，NUnit，MSTest，Selenium或商业QA工具，如HP负载测试工具(HP Load Running)，或VSTS负载测试工具；

广泛(Enrichment)，偏向(Lateral)，垂直(Vertical)，跨职能(Cross-functional)，重新调整(Realignment)，探索(Exploratory)，执行

(Peform)，和其他的追求(Other Pursuits)。

请记住编码是一个重要的技能，你应该去提高它。

session/cookie、cache：存储、尺寸、失效时长、辅助工具

web测试：测试设计（TC、测试策略）、接口、功能、性能、安全、回归、发布验证

itest(junit)接口测试、svn diff、对比；http Mock；业务测试、浏览器、性能、安全

等价类划分方法\*边界值分析方法\*错误推测方法\*场景设计方法\*因果图方法

http://kelude.taobao.net/faq/query\_file.htm?typeId=12

http://kelude.taobao.net/pm/

GUI 自动化测试框架？

Automanx(webDriver)+ Appium

手工+自动化--协议、参数独立性/上下限、异常模拟、内部业务逻辑

http://act.mei.tmall.com/go/market/mei/wppk.php?spm=a221z.7076649.1002.1.8pR2EJ&img=http://www.baidu.com/img/1.png

性能测试：

工具：Grinder、Jmeter

性能诊断：VisualVM、Btrace、Jtop

OS层：Top/iostat/sar/perf ...

自动化测试回归体系

页面JS错误、加载--jstracker

页面兼容性--猫须

PC页面逻辑--automanx

H5页面逻辑--Appium

前后端分离--if-test

对外接口--itest

业务逻辑--itest

java web

servlet--> Spring mvc + velocity + kissy

分布式 Session:tbsession

CDN + memcached + tomcat

ali java web

webx--http://openwebx.org

Webx Turbine Webx Framework SpringExt

性能测试名词

1.Transaction事务

2.TPS

3.PV

4.Peak PV

5.Concurrency并发

6.性能场景

7.RT(Response Time)

8.监控资源中常见的指标

性能测试需求评估

在实施性能测试之前，需要对被测项目做相应的评估。 实施前的评估，主要目的是明确是否需要做性能测试和确立性能点，明 确该测什么、期望值是多少。测试期望值也会根据情况评估，要求被测系 统能满足将来一定时间段的压力。性能测试评估分为测试前的评估和测试 后的评估。这里重点阐述测试前的评估，测试后的评估在“性能测试通过 标准”章节进行描述。 主要从以下5个维度进行测试前的评估： 1. 业务逻辑 2. 日PV量 3. 逻辑复杂度4.运营推广计划（产品未来发展）5.逻辑改动复杂度

监控指标

性能测试通常需要监控的指标包括： 1. 服务器：Linux应用服务器。 具体包括CPU、Memory、Load、I/O、Disk等。 2. 数据库：1.Mysql 2.Oracle。 具体包括缓存命中、索引、单条SQL性能、数据库线程数、数据池连 接数等。 3. 中间件：1.Jboss 2. Apache/nginx 具体包括线程数、连接数、日志输出等。 4. 网络。 具体包括防火墙、网卡、网线、吞吐量、吞吐率等。 5. 应用服务。 具体包括JVM内存使用和回收、JAVA内存使用、Full GC频率、JAVA类装入和 卸载、日志、线程运行状态（阻塞、等待、正常运行）等。 6. 监控工具（PAP）。 具体包括用户执行情况、场景状态、事务响应时间、TPS、Load、CPU分析图表等。 7. 测试机资源。 具体包括CPU、Memory、网络、日志输出、磁盘空间、负载生成器评估等

常见安全漏洞

XSS：恶意攻击者往Web页面里插入恶意html代码，当用户浏览该页之时，嵌入其中Web里面的html代码会被执行，从而达到恶意用户的特殊目的。

CSRF：跨站请求伪造，是一种对网站的恶意利用。尽管听起来像跨站脚本（XSS）， 但它与XSS非常不同，并且攻击方式几乎相左。XSS利用站点内的信任用户，而CSRF则通过伪装来自受信任用户的请求来利用受信任的网站。查询类的请求禁止带TOKEN，同时也禁止TOKEN直接返回在content内容中

SQL Injection：由于程序员的水平及经验也参差不齐，相当大一部分程序员在编写代码的时 候，没有对用户输入数据的合法性进行判断，使应用程序存在安全隐患。用户可以提交一段数据库查询代码，根据程序返回的结果，获得某些他想得知的数据，这就 是所谓的SQL Injection，即SQL注入。

敏感数据：途径A：在外部站点泄露员工信息等敏感数据。途径B：在内部应用泄露用户信息等敏感数据。

文件上传：

File upload

图T1a.lbXixMXXb1upjX.jpg

url redirection

http://www.fvck.com/url.php?url=http://www.baidu.com

http://www.fvck.com/url.php?url=%0A%0DLocation:%20http://www.baidu.com%0A%0D

http://www.fvck.com/url1.php?url=javascript:alert%281%29

Phishing

黑盒测试准备

HackBar 1.6.2 a toolbar that helps you find and test SQL injections

HttpFox 0.8.14 An HTTP analyzer addon for Firefox

访问“安全测试平台”：http://stc.taobao.net

安全框架搭建

请参考http://twiki.corp.taobao.com/bin/view/SRE/Taobao\_Security/JavaCodingRules

测试设计：

促使测试人员深入了解需求

有利于测试的组织；避免盲目测试并且提高测试效率。

确保功能不被遗漏；

有利于测试的重复执行，减少对个人的依赖-；

测试过程可量化的

在少数高风险的测试中，必须证明确实执行了计划执行的测试；

使软件测试的实施重点突出、目的明确

1.对前面评审后的需求进行进一步的评审

2.对于大的需求做一些分解和加深理解

3.产出所有具体需求的所有测试思路和异常分支

4.对后续的用例编写根本性的指导

5.对整个被测系统进行测试模型的建立

6.为了对被测接口进行接口功能的评审和检查(接口)

7.为了对性能测试点进行规划和场景流程制作(性能)

测试用例设计的依据

考虑使用Clover覆盖率工具和mockito mock工具来进行接口测试的边界测试，观察多个库多个表的数据改变情况

优化一下sql

select a.sn,a.is\_lock from resource\_armory\_info a left join idc\_asset\_list s on s.sn=a.sn where s.order\_id=6492

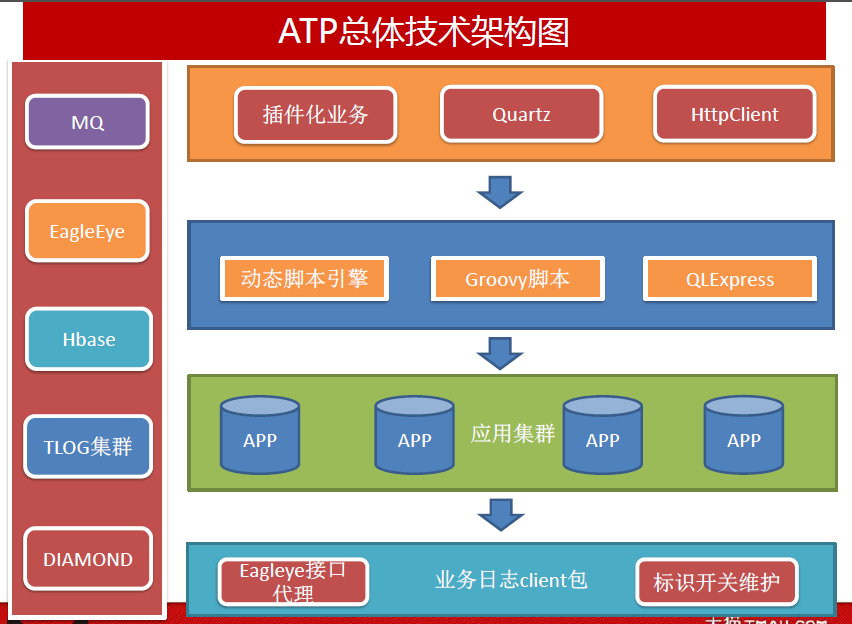
常见性能瓶颈



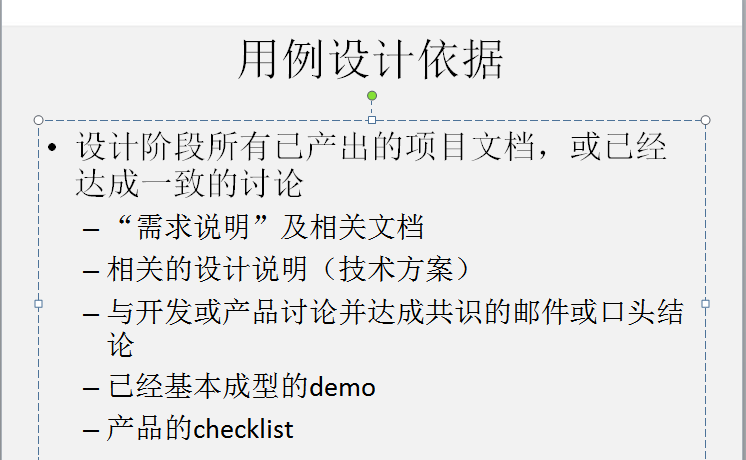
性能优化思路（简）



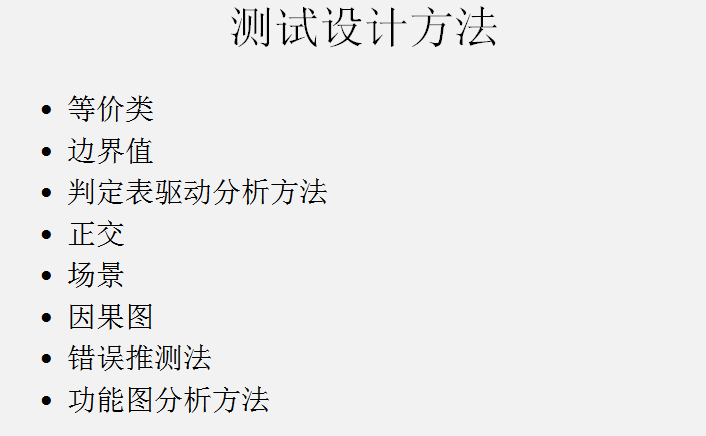
ATP总体技术架构图









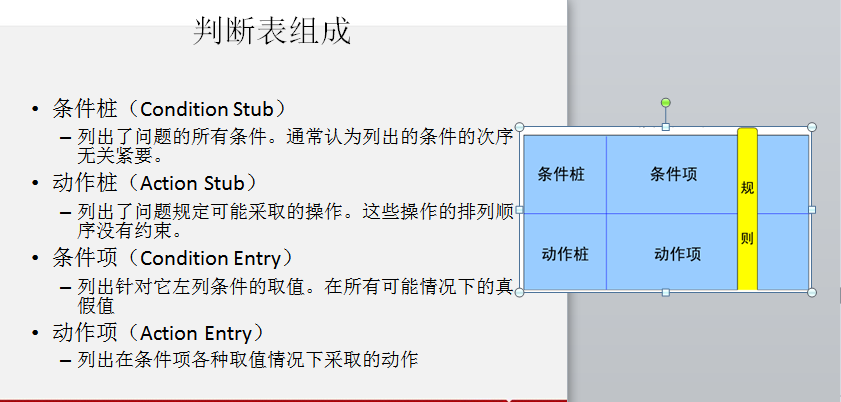


场景法一般包含基本流和备用流，从一个流程开始，通过描述经过的路径来确定的过程，经过遍历所有的基本流和备用流来完成整个场景。场景主要包括4种主要的类型：正常的[用例](http://baike.baidu.com/view/706238.htm)场景，备选的用例场景，异常的用例场景，假定推测的场景。

因果图法是一种适合于描述对于多种输入条件组合的[测试](http://baike.baidu.com/view/1619.htm)方法，根据输入条件的组合、约束关系和输出条件的因果关系，分析输入条件的各种组合情况

列举出程序中所有可能有的错误和容易发生错误的特殊情况,根据他们选择测试用例.

功能图模型由状态迁移图和逻辑功能模型构成.



### 知乎用户，**有几年的学习经历，同时保留了自己的思维…**

[糖果超人](http://www.zhihu.com/people/town-nelson)、[pardLeon](http://www.zhihu.com/people/pard-leon)、知乎用户 [等人赞同](javascript:;)

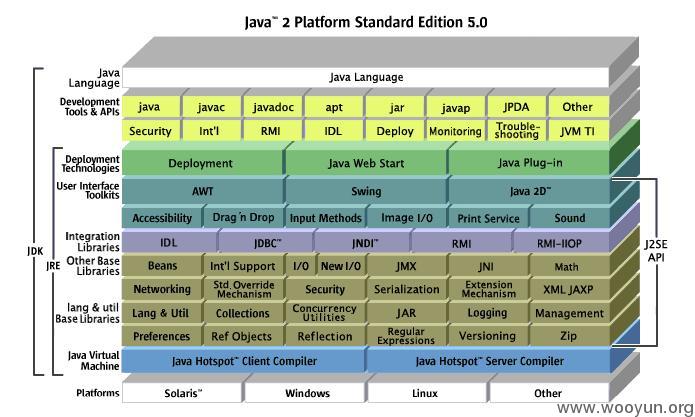
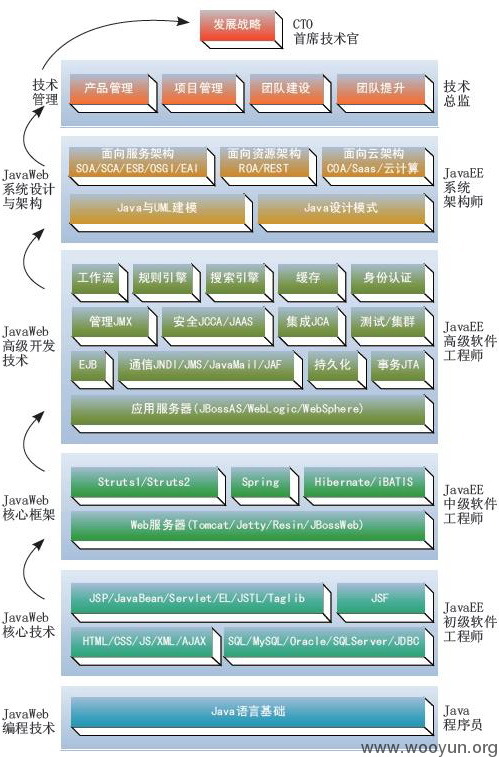
这会有空，给一些初学者一些我认为比较靠谱的建议，仅供参考。  
1，不一定要把java学深入。有多少人搞程序是喜欢这个的，搞程序一半是因为钱多，一半是因为晋升公平。那么问题就来了，如果是为了钱，那为什么不直接选择一个挣钱多的方向呢？  
2，初学者学java至少要脱产半年的时间才能算了解了，才适合被人带，因为至少半年的时间，才知道遇到问题如何解决，有的人需要更长的时间。现在有很多人已经有5年左右的经验了，在他们的压制下，java初学者很难有逆袭的机会了。  
3，行业环境，现在数据存储，后台打包的方案的创业公司已经有苗头，也就是说，后台开发已经被集中解决，日后会越来越成熟，这就是意味着，java招聘需要有经验的人，也就是说，必须熬几年后才能有突破。  
4，java离市场较远，所以呢？java开发人员很难有一个大的突破。  
5，程序里面有变量所以解决了一系列类似的问题，所以注定了程序必须提供大批服务时才能有高收益，不知道大家明白这句的意思不。如果不理解，就思考下云服务器怎么才能赚钱。  
6，如果是想挣钱，如果是要公平。那就放弃追求高薪的想发，java只是一门语言，现在辉煌，以后就会凋零；java工程师也只是一个岗位，今天这个岗位很重要，明天就不一定了。任何一个岗位的薪资不是由能力决定的，是市场决定。今天java稀缺，明天如果出来一大批，价位马上下来。我本来想建议一下还没入门，想做java的同学转去学html5，学设计，但是觉得这也不就是一门语言，和java类似。  
啰嗦了这么久，我只是了想告诉大家，不要被java限死了，任何时候老板都不会在乎你用什么技术，老板需要的是低成本，高收益，只有这样他们才能活下去，这不怪他们。今天为什么java热，因为成本低，php热，因为成本低，站在老板的角度，站在用户的角度，他们会在乎你用java用php吗 这个思想我提过几遍了，但是还有好些人好些没看见。  
  
上知乎的人，有一部分有自己的见解，有一部分是来解决困惑的。为什么现在的大学生有困惑，因为老师没有教如何解决困惑，由于惯性，很多学生也就没有培养这方面的能力，现在问，是学java还是学php，非常简单的解决方法就是问自己的学语言的目前，为了钱就去看招聘网站 那个给的钱多学那个，有的说，php难，java简单 。怎么有这么多问题，想要钱多，还不想出力，给钱多说明这个门槛高，不行就不上。有时候很多人讨论这些问题就和女的讨论去哪吃饭一样，能不能有点主见，干什么都要问，那还干毛啊。不要用战术上的勤奋遮盖战略上的懒惰，说的就是那些不思考的人。别人怎么知道答案，别人开始的时候就没想过，直接干了。其实这个问题也是，怎么入门，你说怎么入门，你是中国人，你学英语怎么入门的，学java的时候就怎么入门。  
又有人说了， 因为不会所以来问，那ok，正常。有的人，不会就去学，有的人，不会就去问然后去学，而有的人，不会去问然后就不了了之了。人和人的差距是不是就在这里  
  
2015-01-13  
===================我是快乐的分割线======================  
应一些知友的要求，我建了一个群，供大家交流交流，328994417。  
2015-01-11  
===================我是快乐的分割线======================  
我说一下，文章的顺序哈。越是上面的内容，越是最近更新的。今天有点空，我想分享一些工具给大家。(我只是做web的开发，所以介绍的内容有限)  
jdk,tomcat,mysql，这三个就不用多说了，必须的。继续，如果ide 我推荐intellij 不要使用eclipse，myeclipse这些了,intellij 是java开发最好的ide，没有之一。如果想用轻文本编辑器，可以考虑notepad,或者是sublime,刚刚开始学的时候，我就是用notepad写，用jdk编译运行。选好ide后，就花一点时间学一学他们，工欲善其事，必先利其器嘛。学一段时间，需要管理项目的时候，了解一些插件，譬如maven，svn,git等等 这些东西能给你们剩下很多时间专心在业务或者是钻研上。哦，对了 还有，搞程序 必须还会翻墙，这里我推荐 chrome+goagent+SwitchySharp，稳定，免费，速度也能接受。  
2015-01-03  
===================我是快乐的分割线======================  
非常感谢大家的支持，真的非常感谢，我个人现在的压力比较大，今年大学毕业，今年也就辞职了，也面临了生活的压力，但是好在有一技傍身，不至于饿死，我已经开始规划自己以后的路了，有自己的打算，所以以后就不太可能更新这篇文章了，我的文笔很差，有一句说一句，望读者减量。  
最后我想说一点自己的总结，希望对大家有用。  
1，程序没有一个入门和不入门的概念，大家不要扣这些字眼；  
2，无论是哪种语言开发，程序都是运行在计算机上面的，所以计算机的知识必须了解；  
3，程序开发只是一门工作，现在很热，以后说不定就冷了，这只是一份职业，只是工资高，赚钱不赚名，想清楚自己的规划再做决定；  
4，坚持与实践，程序是必须动手才能有产出的，只动脑是 不！可！能！的。  
5，祝大家开心的工作，生活，这才是最重要的！  
谢谢大家.  
===================我是快乐的分割线======================  
周一来了个大早，补一点：  
程序的说道底，就是程序设计+算法； 如果觉得自己不适合搞算法，那程序设计总得学吧，不然怎么活下去。那么问题来了？学程序设计哪家强！：）。 不闹了 。程序设计，就是组织代码，让你的代码更容易复用，让你的代码精炼，即 多加一个字符嫌多，少加一个就跑不起的地步。   
好了 ，现不说了 ，某总来了。  
===================我是快乐的分割线======================  
非常感激大家的支持。下面的东西的都是非常基础的 编程久了就会面临更大的困难 举个栗子 多。这个太重要了，随着项目的复杂加深，java文件多，数据库表多，配置文件多，写程序的时候就需要控制这些了。把下面的学会了 就赶紧做项目，做多了就看到自己的渺小了。哈哈 感触很深。大家一起加油，努力奋斗吧！  
===================我是快乐的分割线======================  
刚刚捡到一张图，貌似很有用的样子，so,look look   
  
===================我是快乐的分割线======================  
昨天晚上看到这个题目，当时要睡觉了就没回。趁上班偷闲时间，好好回答下。  
本人从自学java开始，已经两年有余。现在到达的程度就是，敢接10W以内的项目，2个月内，再加一个美工就能把活完成得漂漂亮亮。  
以上属于废话，接下来才是重点，完整的把学习路线给你说下。  
1，HelloWord!首先，学习如何把这个单词打印在控制台。这点成功能给你点信心和乐趣。当然，如果你不会，怎么办。网上找教程。<马士兵>的教程就很不错，**边看边学**，这是必须的，切记。  
他的教程会有一个完整路线，主要先把J2SE的看完。按照顺序看，一定要认真的看，并且自己把视频归类：理论部分，操作部分。这部分看完得2个月左右吧，按照每天10小时，每周6天的进度，刚刚开始会比较慢，不要怕，我那个时候就是在暑假的时候看的，没有女朋友，只能撸电脑了。哈哈！  
2，俄罗斯方块！上面的看完马上干这个，网上有教程，按照他上面的做，当然，自己改下颜色，造型什么的。大概3周左右吧!  
  
===================我是快乐的分割线======================  
好，通过以上大概3个月的时间，恭喜楼主，您已经入门了，但仍然是屌丝，不过你马上就要被你们班的女神仰望了。接下来的时间，看你分类的理论视频，边看边思考，这是什么，那是什么，这有什么作用，基本就是4W1H，哈哈高中学的这个词，不知道是5W还是4W了，不管了。  
这个时候你大概懂了jdk，类型，变量，函数，IO，泛型，swing，反射，接口，实现，等等一堆"专业"词汇，但是这是不够的，让你的女神给你跪舔，你还要学习一门利器，利器——J2EE，哈哈，好像很高级的样子是吧！放心吧，少年，你的高富帅哥哥们(譬如我辈等，已经给你制造了很多轮子，你只需要坐上来就能策马奔腾了)什么是J2EE，举个栗子，淘宝网就一个J2EE实现。虽然你的女神天天上淘宝，但是很快你就能上女神了。接下来告诉你怎么玩转J2EE。  
===================我是快乐的分割线======================  
  
1，简单说下，就是让你做一个网站出来，这个网站的内容是动态生成的，什么跟什么啊，我不解释，你自己上网查。  
2，网上一样有教程，我这里给你说下，你要注意的一些事，也就是说，我只会给你interface，不会给你具体的implemention。如下  
a,tomcat，你要知道这是什么，干什么用的，怎么用。  
b,数据库(mysql)，如上这是什么，干什么用的，怎么用。  
c,jsp,..................................  
d,servlet...........................  
好了，就4个，老老实实学会我说的，看API，读reference。当你读到一半的是时候，就能秒杀一大片的同学基友了。  
之后就学习，**设计模式**:工厂模式，责任链模式等等，不要问这是什么，你就看，看完之后，你就知道如何写代码了。“我特么都会web了 还不知道怎么写代码？”此刻你肯定有这种想法。为什么让你看，因为以前你一直看着视频写代码，脱离了视频，你根本都不会写，就像有拐棍你知道怎么走，没了你根本不会走。这部分一定要看，不然你写的代码，别人根本看不懂。我这里指的别人都说高手大牛哦！  
好了，哥没有坑你。  
接下来学习点框架，见见世面。  
持久层：Hibernate,Mybatis  
控制层：Spring MVC,Struts2  
视图层：JSP，Jstl , EL，bootstrap,easy UI, Jquery。  
通信：JSON，XML  
然后你就发现世界是多么的美好，生活是多么的和谐，等你学会了 ，就接个万把块钱的项目练练手吧，顺便搂在你的女神过个情人节，然后啦啦啦啦 你懂啦。  
===================我是快乐的分割线======================  
  
亲，您想月薪过万吗？您想出国XX游吗？如果想就接着学点东西吧！  
记住几个专业词先，分布式，集群，高并发，高吞吐，等等。  
既然你选择了java就渐渐忘记算法，数学之类的吧。思考思考下，淘宝网天天被那么多人上，是如何支撑到现在的。这里的那么多是以百万为单位的哦。如果你能解决一个问题，轻轻松松进入BAT，然后在别人像狗一样找工作的时候，拿着月薪过万的OFFER陪你女神毕业之旅吧！  
这部分我也不擅长，所以不多说了。我现在也在这条路上。有时候想得多了，做得反而就少了，我也是有这个毛病。楼主千万别以代码太容易就不写了，一定要动手，只有动手了才会发现问题，才有自己的思考。简单的码代码是没有意义的(这里指的的是重复的写类似的功能的时候)，只有你能走上设计代码的路线上，你才有价值，譬如设计数据库表，设计系统框架，设计工具等等，等你到一定深度的时候就明白了。说多了容易把你弄迷糊，暂且你先开动起手来，开始做东西，加油楼主。女神在像你招手哦！  
  
===================我是快乐的分割线======================  
后记，虽然写得不是很好，好歹看客们给个赞，让我也涨涨信心吧！  
  
===================我是快乐的分割线======================  
非常感谢大家的支持，我现在也是在工作， 也在学习新的知识，现在的技术发展的很快，很多新东西必须学习才能提高开发效率，但是java必须得学，基础的东西一定要打牢，其他的东西学起来就很快了，再次感谢大家的支持，有你们，很愉快！

HttpServlet

Jsp（java server page）隐含对象：9个

<% 中使用：PageContext、（httpServletRequest）request、response、（HttpSession）session、（ServletContext）application、（ServletConfig）config、（JspWriter）out、（Object）page、Exception %>





舒适圈

压力圈

正则表达式：

[:alnum:]代表英文大小写字母及数字

[:alpha:]代表英文大小写字母

[:blank:]代表空格和 tab 键

[:cntrl:]键盘上的控制按键，如 CR,LF,TAB,DEL

[:digit:]代表数字

[:graph:]代表空白字符以外的其他

[:lower:]小写字母

[:print:]可以被打印出来的任何字符

[:punct:]代表标点符号

[:upper:]代表大写字符

[:space:]任何会产生空白的字符如空格，tab,CR 等

[:xdigit:]代表 16 进位的数字类型

* + 完成的定义
    - 比如写个POC
      * 1. 搞懂了目标Web应用漏洞的原理
      * 2. 熟练运用Python各相关模块与机制
      * 3. 熟练了解了HTTP协议
        + HTTP请求
        + HTTP响应
      * 4. 代码写得够规范，让人看起来就是爽
      * 5. 程序经过足够的测试
        + 黑测试
        + 白测试
      * 6. 及时反馈进度
        + 我遇到困难了
        + 我搞定了
      * 7. 更新相关文档，沉淀
  + messagebox_warning

熟练的定义

* + - 比如熟练SQL注入
      * SQL语句这门“语言”能脱离文档顺手写出
      * 主流数据库的SQL特有函数、存储过程、机制我都了如指掌
        + MySQL
        + MSSQL
        + Oracel
        + Postgre
        + Access
        + SQLite
        + ...
      * 牛逼的工具我不仅用的顺其自然，源码还读过几遍，我能修改
        + sqlmap
        + ...
      * 我具备创造性，而不仅仅是跟在大牛身后
        + 研究出了几个不错的技巧
        + 发了几篇不错的Paper
        + 对外会议/沙龙等进行了几次分享
        + 写出了自己的相关工具，爽
      * 我实战了N回，遇到了很多奇葩环境，我有足够的信心绕过
      * 以上这些之后，这才叫熟练！其他同理
  + 好书推荐
    - 推荐理由
      * 打通任督二脉的书，怎能不看？
      * 任何科学研究最终必须至少到哲学层面，触碰到上帝的脚
      * 具体技术类书籍请见“专业技能”相关部分
    - 鸡汤类
      * 黑客与画家
        + 印象深刻：设计者的品味

好设计是简单的设计

抓住本质

好设计是永不过时的设计

如果解决方法是丑陋的，那就肯定还有更好的解决方法，只是还没有发现而已

好设计是解决主要问题的设计

好设计是启发性的设计

好设计通常是有点趣味性的设计

好设计是艰苦的设计

好设计是看似容易的设计

好设计是对称的设计

好设计是模仿大自然的设计

好设计是一种再设计

好设计是能够复制的设计

好设计往往是奇特的设计

好设计是成批出现的

好设计常常是大胆的设计

* + - * 浪潮之巅
        + 感受IT帝国的崛起与没落，我们现在站在又一个互联网浪潮之巅
    - 洁癖类
      * 重构
      * 代码整洁之道
      * 代码大全2
      * 数学之美
    - 敏捷类
      * Rework中文版
        + 37signals团队的敏捷经验
      * 高效程序员的45个习惯
    - 产品类
      * 人人都是产品经理
      * 结网
    - 神书
      * 自私的基因
      * 失控
      * 万物由来
    - ...
* 专业技能
  + idea

原则

* + - 至少完整看完与练习好一本书
    - 至少过一遍官方文档
  + bookmark

基础必备

* + - HTTP抓包与调试
      * Firefox插件
        + Firebug

抓包与各种调试

* + - * + Tamper Data

拦截修改

* + - * + Live Http Header

重放功能

* + - * + Hackbar

编码解码/POST提交

* + - * + Modify Headers

修改头部

* + - * Fiddler
        + 浏览器代理神器
        + 拦截请求或响应
        + 抓包
        + 重放
        + 模拟请求
        + 编码解码
        + 第三方扩展

Watcher

Web前端安全的自动审计工具

* + - * Wireshark
        + 各种强大的过滤器语法
      * Tcpdump
        + 命令行的类Wireshark抓包神器
      * Python
        + urllib2

打开请求响应调试

编辑urllib2的do\_open里的h.set\_debuglevel

改为h.set\_debuglevel(1)，这时可以清晰看到请求响应数据，包括https

* + - 什么是跳转
      * 服务端跳转
        + 302

<?php header("Location: 3.php"); ?>

* + - * + 301

<?php header("HTTP/1.1 301 Moved Permanently"); header("Location: 2.php"); ?>

* + - * + u=urllib2.urlopen(url)后，u.url能得到服务端跳转后的地址

urllib2自己的特性

所谓的会跟进去

* + - * 客户端跳转
        + <meta http-equiv="refresh" content="0; url=http://www.evilcos.me" />

htmlparse解析就行了

* + - * + location.href="http://evilcos.me";

正则（弱），js引擎（王道）

* + - Python编码规范
      * attach

[PythonCodingRule.pdf](http://blog.knownsec.com/Knownsec_RD_Checklist/PythonCodingRule.pdf) [User Link](http://blog.knownsec.com/Knownsec_RD_Checklist/PythonCodingRule.pdf)

* + - * 入门书
        + Python核心编程2

第4章 Python对象

完整熟练

6.8 Unicode

完整熟练

8.11 迭代器和iter()函数

完整熟练

第9章 文件的输入和输出

完整熟练

第10章 错误和异常

完整熟练

第11章 函数和函数式编程

完整熟练

第12章 模块

完整熟练

第14章 执行环境

完整熟练

第15章 正则表达式

idea

完整熟练

第18章 多线程编程

完整熟练

20.2 使用Python进行Web应用：创建一个简单的Web客户端

完整熟练

* + - Office能力
      * Word文档编写，看去要专业，尤其对外的
      * Excel里面大量的统计、图表功能，需要善于使用
      * PPT演讲、培训等必备，如何做好PPT？百度一下……
      * ksmiletris

进一步

* + - * + yEd
        + Visio
        + FreeMind

本技能表就是这个制作

* + - 熟练VIM
      * [实战至少3回合：http://coolshell.cn/articles/5426.html](http://coolshell.cn/articles/5426.html) [User Link](http://coolshell.cn/articles/5426.html)
    - 算法
      * 快排
      * 二分
    - 正则表达式
      * 调试工具
        + ksmiletris

Kodos

* + - * + idea

RegexBuddy

支持多种语言

支持调试优化

* + - * + ksmiletris

<http://www.regexper.com/> [User Link](http://www.regexper.com/)

正则图解

* + - * [正则表达式30分钟入门教程：http://deerchao.net/tutorials/regex/regex.htm](http://deerchao.net/tutorials/regex/regex.htm) [User Link](http://deerchao.net/tutorials/regex/regex.htm)
      * [http://wiki.ubuntu.org.cn/Python正则表达式操作指南](http://wiki.ubuntu.org.cn/Python%E6%AD%A3%E5%88%99%E8%A1%A8%E8%BE%BE%E5%BC%8F%E6%93%8D%E4%BD%9C%E6%8C%87%E5%8D%97) [User Link](http://wiki.ubuntu.org.cn/Python%E6%AD%A3%E5%88%99%E8%A1%A8%E8%BE%BE%E5%BC%8F%E6%93%8D%E4%BD%9C%E6%8C%87%E5%8D%97)
      * attach

[regex/regularexpressions.pptx](http://blog.knownsec.com/Knownsec_RD_Checklist/regex/regularexpressions.pptx) [User Link](http://blog.knownsec.com/Knownsec_RD_Checklist/regex/regularexpressions.pptx)

* + - * attach

[regex/正则表达式引擎浅析.txt](http://blog.knownsec.com/Knownsec_RD_Checklist/regex/about_regx_engine.txt) [User Link](http://blog.knownsec.com/Knownsec_RD_Checklist/regex/about_regx_engine.txt)

* + - 研发能力
      * 瀑布模型
        + 需求->需求分析->设计->开发->测试->上线->运维/运营
      * idea

需求分析能力

* + - * + 给你一个需求，如何给出一个优美的执行思路——方法论
        + 这个能力非常非常非常的关键
      * 调试能力
        + 只要定位出，就没有解决不了的Bugs
        + 肉眼看到的都是假象

一定要专业的工具与经验配合

* + - * + Bugs在哪出现，最终就在哪进行真实模拟调试
        + 缩小范围

构建自己的测试样例

排除网络复杂未知情况

关联模块一个个排除

Python单步调试

import pdb;pdb.set\_trace()

在需要单步调试的地方加上面这句，运行程序后中断在此，然后h查看指令进行一步步细细调试

粗暴调试：print

* + - * 敏捷思想
        + 快速迭代
        + 任务拆细
        + v1原则：定义好v1的目标，快速完成v1为优先
        + 习惯Wiki记录，利于沉淀与分享
    - 翻墙
      * idea

<http://code.google.com/p/goagent/> [User Link](http://code.google.com/p/goagent/)

* + - * SSH隧道
        + <http://www.ibm.com/developerworks/cn/linux/l-cn-sshforward/index.html> [User Link](http://www.ibm.com/developerworks/cn/linux/l-cn-sshforward/index.html)
        + 本地转发

ssh -L <local port>:<remote host>:<remote port> <SSH hostname>

* + - * + 远程转发

反弹

ssh -R <local port>:<remote host>:<remote port> <SSH hostname>

* + - * + 动态转发

ssh -D <local port> <SSH Server>

* + Web安全
    - Web服务组件
      * [8+1：一图胜千言哎:)](http://blog.knownsec.com/Knownsec_RD_Checklist/web_component.png) [User Link](http://blog.knownsec.com/Knownsec_RD_Checklist/web_component.png)
      * 钟馗之眼
        + 网络空间搜索引擎
        + 大数据，懂的人懂，不懂的人不懂
        + [http://www.zoomeye.org](http://www.zoomeye.org/) [User Link](http://www.zoomeye.org/)
      * 组件具有影响面，越底层的组件影响面可能越大
    - 安全维度
      * 漏洞
      * 风险
      * 事件
    - Web安全标准
      * OWASP
      * WASC
      * 我们内部Wiki
    - 实战环境
      * XSS
        + ks-xsslab\_open（内部虚拟机）

可以搞通

XSS

CSRF

ClickJacking

* + - * + <http://xss-quiz.int21h.jp/> [User Link](http://xss-quiz.int21h.jp/)

attach

[答案：xss/xss\_quiz.txt](http://blog.knownsec.com/Knownsec_RD_Checklist/xss/xss_quiz.txt) [User Link](http://blog.knownsec.com/Knownsec_RD_Checklist/xss/xss_quiz.txt)

* + - * SQL
        + <https://github.com/Audi-1/sqli-labs> [User Link](https://github.com/Audi-1/sqli-labs)

SQLI-LABS is a platform to learn SQLI

* + - * 500多个WSL靶场
      * 渗透虚拟机/BT5/Kali
        + 海量各类型黑客工具
    - 书
      * 黑客攻防技术宝典（Web实战篇）
      * 白帽子讲Web安全
      * Web前端黑客技术揭秘
        + 我和xisigr自己出品
      * SQL注入攻击与防御
    - papers
      * <http://www.exploit-db.com/papers/> [User Link](http://www.exploit-db.com/papers/)
      * BlackHat/Defcon/国内各安全沙龙等Papers需要持续跟进
  + 研发清单
    - 编码环境
      * pip
      * Vagrant
      * tmux/screen
      * vim
      * zsh + oh-my-zsh
      * Python2.7
      * >Django1.4
        + <http://djangobook.py3k.cn/2.0/> [User Link](http://djangobook.py3k.cn/2.0/)
      * web.py
      * node.js
      * Ubuntu/Gentoo/Centos
      * ipython
      * 版本控制
        + git/svn
        + gitlab
      * Nginx+uWSGI
    - Python
      * 官方手册
        + 至少过一遍，这都没过一遍，视野会局限
        + 行之说：「我没看过Python的书，却熟读官方手册……」
    - Linux
      * 书
        + attach

[Bash新手指南.pdf](http://blog.knownsec.com/Knownsec_RD_Checklist/linux/bash_freshman.pdf) [User Link](http://blog.knownsec.com/Knownsec_RD_Checklist/linux/bash_freshman.pdf)

* + - * + attach

[高级Bash脚本编程.pdf](http://blog.knownsec.com/Knownsec_RD_Checklist/linux/advanced_bash.pdf) [User Link](http://blog.knownsec.com/Knownsec_RD_Checklist/linux/advanced_bash.pdf)

* + - * attach

[Bash快捷操作.txt](http://blog.knownsec.com/Knownsec_RD_Checklist/linux/bash_shortcut.txt) [User Link](http://blog.knownsec.com/Knownsec_RD_Checklist/linux/bash_shortcut.txt)

* + - * attach

[screen最佳实践.pdf](http://blog.knownsec.com/Knownsec_RD_Checklist/linux/screen.pdf) [User Link](http://blog.knownsec.com/Knownsec_RD_Checklist/linux/screen.pdf)

* + - * attach

[crontab格式详解.pdf](http://blog.knownsec.com/Knownsec_RD_Checklist/linux/crontab.pdf) [User Link](http://blog.knownsec.com/Knownsec_RD_Checklist/linux/crontab.pdf)

* + - 前端
      * 书
        + JavaScript DOM编程艺术
      * 了解DOM
        + 这同样是搞好前端安全的必要基础
      * 库
        + jQuery

优秀的插件应该体验一遍，并做些尝试

官方文档得过一遍

* + - * + ECharts

来自百度

* + - * + Google API
        + ZoomEye Map组件

ZoomEye团队自己基于开源的打造

* + - * + AngularJS

Google出品的颠覆性前端框架

* + - * + Bootstrap

应该使用一遍

* + - 爬虫进阶
      * 代理池
        + 爬虫「稳定」需要
      * 网络请求
        + wget/curl
        + urllib2/httplib2/requests
        + idea

scrapy

* + - * 验证码破解
        + pytesser
    - 调度
      * crontab是最原生的定时调度
      * 基于redis实现的分布式调度
      * 基于rpyc实现的分布式调度
      * celery/gearman等调度框架
    - 并发
      * 线程池
        + 进程内优美的并发方案
      * 协程
        + 进程内另一种优美的并发方案
      * 多进程
        + os.fork
        + idea

multiprocessing

* + - 数据结构
      * JSON
      * cPickle
      * protobuf
    - 数据库
      * MySQL
      * MongoDB
      * Cassandra
      * Hadoop体系
      * Redis
      * Sqlite
      * bsddb
    - DevOps
      * SSH证书
      * Fabric
      * SaltStack
      * puppet
      * pssh/dsh
    - 调试
      * pdb
      * logging
      * Sentry
      * strace/ltrace
      * lsof
      * 性能
        + Python内

timeit

cProfile

[Python性能分析指南：http://www.oschina.net/translate/python-performance-analysis](http://www.oschina.net/translate/python-performance-analysis) [User Link](http://www.oschina.net/translate/python-performance-analysis)

* + - * + Python外

top/htop/free/iostat/vmstat/ifconfig/iftop...

* + - 算法
      * 分词
      * 贝叶斯
        + attach

[algorithm/贝叶斯.txt](http://blog.knownsec.com/Knownsec_RD_Checklist/algorithm/bayes.txt) [User Link](http://blog.knownsec.com/Knownsec_RD_Checklist/algorithm/bayes.txt)

* + - * 神经元
      * 遗传算法
      * 聚类/分类
      * ...
    - 持续集成
      * 自测试
        + nose
      * Jenkins
    - 协作
      * 类似Trello的在线协同平台
      * 微信
      * 立会
  + 设计思想
    - 人人都是架构师：具备架构思想是一件多酷的事
    - 实战出真知
    - 如何设计
      * attach

[任务架构设计变迁.pdf](http://blog.knownsec.com/Knownsec_RD_Checklist/arch_design_evolution.pdf) [User Link](http://blog.knownsec.com/Knownsec_RD_Checklist/arch_design_evolution.pdf)

* + - * 松耦合、紧内聚
      * 单元与单元属性
      * 生产者与消费者
      * 结构
        + 队列
        + LRU
      * 分布式
        + 存储
        + 计算
      * 资源考虑
        + CPU
        + 内存
        + 带宽
      * 粗暴美学/暴力美学
        + 大数据，先考虑run it，然后才能知道规律在哪
        + 「run it优先」能快速打通整体，洞察问题
        + 「run it优先」能摆脱细节（繁枝末节）的束缚
        + 「run it优先」能快速迭代出伟大的v1
      * 一个字总结
        + 美
  + 优质资源
    - [知乎周刊：http://zhuanlan.zhihu.com/Weekly](http://zhuanlan.zhihu.com/Weekly) [User Link](http://zhuanlan.zhihu.com/Weekly)
    - [码农周刊：http://weekly.manong.io/](http://weekly.manong.io/) [User Link](http://weekly.manong.io/)
    - [Pycoder's Weekly：http://pycoders.com/archive/](http://pycoders.com/archive/) [User Link](http://pycoders.com/archive/)
    - [Hacker News：https://news.ycombinator.com/](https://news.ycombinator.com/) [User Link](https://news.ycombinator.com/)
    - [Startup News：http://news.dbanotes.net/](http://news.dbanotes.net/) [User Link](http://news.dbanotes.net/)
    - [极客头条：http://geek.csdn.net/](http://geek.csdn.net/) [User Link](http://geek.csdn.net/)
    - [InfoQ：http://www.infoq.com/cn](http://www.infoq.com/cn) [User Link](http://www.infoq.com/cn)
    - [Stack Overflow：http://stackoverflow.com/](http://stackoverflow.com/) [User Link](http://stackoverflow.com/)
    - [GitHub：https://github.com/](https://github.com/) [User Link](https://github.com/)
    - [FreeBuf：http://www.freebuf.com/](http://www.freebuf.com/) [User Link](http://www.freebuf.com/)
    - [WooYun：http://drops.wooyun.org/](http://drops.wooyun.org/) [User Link](http://drops.wooyun.org/)
  + 牛人1,2,3
    - 1研究：研究东西，有足够洞察力，研究水准不错
    - 2研发：hack idea自己有魄力实现，不懂研发的黑客如同不会游泳的海盗
    - 3工程：研发出来的需要实战、需要工程化，否则只是玩具，而不能成为真的武器