如何成为一名合格的程序员

1　方法/步骤

你是否觉得自己从学校毕业的时候只做过小玩具一样的程序？走入职场后哪怕没有什么经验也可以把以下这些课外练习走一遍（朋友的抱怨：学校课程总是从理论出发，作业项目都看不出有什么实际作用，不如从工作中的需求出发）

建议：

不要乱买书，不要乱追新技术新名词，基础的东西经过很长时间积累而且还会在未来至少 10 年通用。

回顾一下历史，看看历史上时间线上技术的发展，你才能明白明天会是什么样。

一定要动手，例子不管多么简单，建议至少自己手敲一遍看看是否理解了里头的细枝末节。

一定要学会思考，思考为什么要这样，而不是那样。还要举一反三地思考。

注：你也许会很奇怪为什么下面的东西很偏 Unix/Linux，这是因为我觉得 Windows 下的编程可能会在未来很没有前途，原因如下：

现在的用户界面几乎被两个东西主宰了，1）Web，2）移动设备 iOS 或 Android。Windows 的图形界面不吃香了。

越来越多的企业在用成本低性能高的 Linux 和各种开源技术来构架其系统，Windows 的成本太高了。

微软的东西变得太快了，很不持久，他们完全是在玩弄程序员。详情参见《Windows 编程革命史》

所以，我个人认为以后的趋势是前端是 Web+移动，后端是 Linux+开源。开发这边基本上没 Windows 什么事。

启蒙入门

1、 学习一门脚本语言，例如 Python/Ruby

可以让你摆脱对底层语言的恐惧感，脚本语言可以让你很快开发出能用得上的小程序。实践项目:

处理文本文件，或者 csv (关键词 python csv, python open, python sys) 读一个本地文件，逐行处理（例如 word count，或者处理 log）

遍历本地文件系统 (sys, os, path)，例如写一个程序统计一个目录下所有文件大小并按各种条件排序并保存结果

跟数据库打交道 (python sqlite)，写一个小脚本统计数据库里条目数量

学会用各种 print 之类简单粗暴的方式进行调试

学会用 Google (phrase, domain, use reader to follow tech blogs)

为什么要学脚本语言，因为他们实在是太方便了，很多时候我们需要写点小工具或是脚本来帮我们解决问题，你就会发现正规的编程语言太难用了。

2、 用熟一种程序员的编辑器(不是 IDE) 和一些基本工具

Vim / Emacs / Notepad++，学会如何配置代码补全，外观，外部命令等。

Source Insight (或 ctag)

使用这些东西不是为了 Cool，而是这些编辑器在查看、修改代码/配置文章/日志会更快更有效率。

3、 熟悉 Unix/Linux Shell 和常见的命令行

如果你用 windows，至少学会用虚拟机里的 linux， vmware player 是免费的，装个 Ubuntu 吧

一定要少用少用图形界面。

学会使用 man 来查看帮助

文件系统结构和基本操作 ls/chmod/chown/rm/find/ln/cat/mount/mkdir/tar/gzip ...

学会使用一些文本操作命令 sed/awk/grep/tail/less/more ...

学会使用一些管理命令 ps/top/lsof/netstat/kill/tcpdump/iptables/dd...

了解/etc 目录下的各种配置文章，学会查看/var/log 下的系统日志，以及/proc 下的系统运行信息

了解正则表达式，使用正则表达式来查找文件。

对于程序员来说 Unix/Linux 比 Windows 简单多了。（参看我四年前 CSDN 的博文《其实 Unix 很简单》）学会使用 Unix/Linux 你会发现图形界面在某些时候实在是太难用了，相当地相当地降低工作效率。

4、 学习 Web 基础（HTML/CSS/JS) + 服务器端技术 (LAMP)

未来必然是 Web 的世界，学习 WEB 基础的最佳网站是 W3School。

学习 HTML 基本语法

学习 CSS 如何选中 HTML 元素并应用一些基本样式（关键词：box model）

学会用 Firefox + Firebug 或 chrome 查看你觉得很炫的网页结构，并动态修改。

学习使用 Javascript 操纵 HTML 元件。理解 DOM 和动态网页
（http://oreilly.com/catalog/9780596527402) 网上有免费的章节，足够用了。或参看 DOM 。

学会用 Firefox + Firebug 或 chrome 调试 Javascript 代码（设置断点，查看变量，性能，控制台等）

在一台机器上配置 Apache 或 Nginx

学习 PHP，让后台 PHP 和前台 HTML 进行数据交互，对服务器相应浏览器请求形成初步认识。实现一个表单提交和反显的功能。

把 PHP 连接本地或者远程数据库 MySQL（MySQL 和 SQL 现学现用够了）

跟完一个名校的网络编程课程（例如：）不要觉得需要多于一学期时间，大学生是全职一学期选 3-5 门课，你业余时间一定可以跟上

学习一个 javascript 库（例如 jQuery 或 ExtJS）+ Ajax (异步读入一个服务器端图片或者数据库内容）+JSON 数据格式。

HTTP: The Definitive Guide 读完前 4 章你就明白你每天上网用浏览器的时候发生的事情了(proxy, gateway, browsers)

做个小网站（例如：一个小的留言板，支持用户登录，Cookie/Session，增、删、改、查，上传图片附件，分页显示）

买个域名，租个空间，做个自己的网站。

进阶加深

1、 C 语言和操作系统调用

重新学 C 语言，理解指针和内存模型，用 C 语言实现一下各种经典的算法和数据结构。推荐《计算机程序设计艺术》、《算法导论》和《编程珠玑》。

学习（麻省理工免费课程）计算机科学和编程导论

学习（麻省理工免费课程）C 语言内存管理

学习 Unix/Linux 系统调用（Unix 高级环境编程），，了解系统层面的东西。

用这些系统知识操作一下文件系统，用户（实现一个可以拷贝目录树的小程序）

用 fork/wait/waitpid 写一个多进程的程序，用 pthread 写一个多线程带同步或互斥的程序。多进程多进程购票的程序。

用 signal/kill/raise/alarm/pause/sigprocmask 实现一个多进程间的信号量通信的程序。

学会使用 gcc 和 gdb 来编程和调试程序（参看我的《用 gdb 调试程序》）

学会使用 makefile 来编译程序。（参看我的《跟我一起写 makefile》）

IPC 和 Socket 的东西可以放到高级中来实践。

学习 Windows SDK 编程（Windows 程序设计 ，MFC 程序设计）

写一个窗口，了解 WinMain/WinProcedure，以及 Windows 的消息机制。

写一些程序来操作 Windows SDK 中的资源文件或是各种图形控件，以及作图的编程。

学习如何使用 MSDN 查看相关的 SDK 函数，各种 WM_消息以及一些例程。

这本书中有很多例程，在实践中请不要照抄，试着自己写一个自己的例程。

不用太多于精通这些东西，因为 GUI 正在被 Web 取代，主要是了解一下 Windows 图形界面的编程。@virushuo 说：“我觉得 GUI 确实不那么热门了，但充分理解 GUI 工作原理是很重要的。包括移动设备开发，如果没有基础知识仍然很吃力。或者说移动设备开发必须理解 GUI 工作，或者在 win 那边学，或者在 mac/iOS 上学”。

2、学习 Java

Java 的学习主要是看经典的 Core Java 《Java 核心技术编程》和《Java 编程思想》（有两卷，我仅链了第一卷，足够了，因为 Java 的图形界面了解就可以了）

学习 JDK，学会查阅 Java API Doc

了解一下 Java 这种虚拟机语言和 C 和 Python 语言在编译和执行上的差别。从 C、Java、Python 思考一下“跨平台”这种技术。

学会使用 IDE Eclipse，使用 Eclipse 编译，调试和开发 Java 程序。

建一个 Tomcat 的网站，尝试一下 JSP/Servlet/JDBC/MySQL 的 Web 开发。把前面所说的那个 PHP 的小项目试着用 JSP 和 Servlet 实现一下。

3、Web 的安全与架构

学习 HTML5，网上有很多很多教程，以前酷壳也介绍过很多，我在这里就不罗列了。

学习 Web 开发的安全问题（参考新浪微博被攻击的这个事，以及 Ruby 的这篇文章）

学习 HTTP Server 的 rewrite 机制，Nginx 的反向代理机制，fast-cgi（如：PHP-FPM）

学习 Web 的静态页面缓存技术。

学习 Web 的异步工作流处理，数据 Cache，数据分区，负载均衡，水平扩展的构架。

实践任务：

使用 HTML5 的 canvas 制作一些 Web 动画。

尝试在前面开发过的那个 Web 应用中进行 SQL 注入，JS 注入，以及 XSS 攻击。

把前面开发过的那个 Web 应用改成构造在 Nginx + PHP-FPM + 静态页面缓存的网站

4、学习关系型数据库

你可以安装 MSSQLServer 或 MySQL 来学习数据库。

学习教科书里数据库设计的那几个范式，1NF，2NF，3NF，……

学习数据库的存过，触发器，视图，建索引，游标等。

学习 SQL 语句，明白表连接的各种概念（参看《SQL Join 的图示》）

学习如何优化数据库查询（参看《MySQL 的优化》）

实践任务：设计一个论坛的数据库，至少满足 3NF，使用 SQL 语句查询本周，本月的最新文章，评论最多的文章，最活跃用户。

5、一些开发工具

学会使用 SVN 或 Git 来管理程序版本。

学会使用 JUnit 来对 Java 进行单元测试。

学习 C 语言和 Java 语言的 coding standard 或 coding guideline。（我 N 年前写过一篇关 C 语言非常简单的文章——《编程修养》，这样的东西你可以上网查一下，一大堆）。

推荐阅读《代码大全》《重构》《代码整洁之道》

标签： 学会, 程序员
分类： 数码&电脑
时间： 2015-02-03