The Google Tech Dev Guide is great, but I liked the simplicity of the previous version. So I retrieved the Aug 2017 version of the Google Guide to Technical Development from the Internet Archive, and reformatted it for Github. I plan to make minor updates, but will try to keep it as simple as the original.

Technical Development Guide

This guide provides tips and resources to help you develop your technical skills (academically and non-academically) through self-paced, hands-on learning.

This guide is intended for Computer Science students seeking an internship or university grad role at Google.

What this guide is for

- You can use this guide to determine which courses to take, but be sure stay on track with your courses required for your major to graduate.
- We encourage you to learn more outside of this guide. The more you know, the better!
- The online resources we've cited aren't meant to replace courses available at your university, but they may help supplement your education or provide an introduction to a topic.
- The information and recommendations in this guide were gathered through our work with students and candidates in the field. It is a work-in-progress, a living document, so be sure to periodically check back for updates.

Note: Following the recommendations in the guide does not guarantee a job at Google.

How to use this guide

- The guide lists topics and resources in a rough progression, from possible places to begin if you have little or no technical skills, to resources for those with increasing skills, to ways to gain exposure in the Computer Sciences field.
- You can use any of the resources you want, in any order.

Recommendations and Resources

Take an "Introduction to CS" course	Focus on basic coding instructions. Online resources: • CS101 - Intro to Computer Science, Udacity • CS50x - Introduction to Computer Science, Harvard, edX
Learn to code in (at least) one object-oriented programming language (C++, Java, Python)	Online resources for beginning programmers: • Learn to Program: The Fundamentals (Python), University of Toronto, Coursera • Google's Python Class • Introduction to Interactive Programming in Python, Rice University, Coursera • Java Programming: An Introduction to Software, Duke University, Coursera • Introduction to Programming in Java, MIT OpenCourseWare Online resources for more experienced programmers: • Java Programming: Data Structures and Beyond, University of California San Diego, specialization on Coursera • Design of Computer Programs (Python), Udacity • Learn to Program: Crafting Quality Code (Python), University of Toronto, Coursera • Introduction to Programming Languages, Brown University
Learn other programming languages	Add to your repertoire: • JavaScript • CSS & HTML • Ruby • PHP • C • Perl • Shell script • Lisp • Scheme Online resources: • Codecademy
Test your code	Learn how to catch bugs, create tests, and break your software. Online resources: • Software Testing, Udacity • Software Debugging, Udacity
Develop logical reasoning and knowledge of discrete math	Online resources: • Mathematics for Computer Science, MIT OpenCourseWare

	• If Mathematics for Computer Science is too challenging, try taking the Discrete Mathematics Specialization first. • Introduction to Mathematical Thinking, Stanford, Coursera • Effective thinking through mathematics, University of Texas at Austin, edX • Probabilistic Graphical Models, Stanford, Coursera • Game Theory, Stanford and University of British Columbia, Coursera
	Learn about fundamental data types (stack, queues, and bags), sorting algorithms (quicksort, mergesort, heapsort), data structures (binary search trees, red-black trees, hash tables), and Big O.
Develop a strong understanding of algorithms and data structures	Online resources: • Introduction to Algorithms, MIT OpenCourseWare • Algorithms Part 1 & Algorithms Part 2, Princeton, Coursera • CS61B - Data Structures (and video lectures), UC Berkeley • List of Algorithms, Wikipedia • List of Data Structures Wikipedia • Book: The Algorithm Design Manual, Steven S. Skiena
Develop a strong knowledge of operating systems	Online resources: • CS162 - Operating Systems and Systems Programming, UC Berkeley, YouTube
Learn artificial intelligence and machine learning	Online resources: • Machine Learning Engineer nanodegree, Udacity • Deep Learning, Udacity • Introduction to Robotics, Stanford University • Machine Learning, Stanford University
Learn Android development	Online resources: • Google Developer Training for Android
Learn web development	Online resources: • Google Developer Training for Web • freeCodeCamp - Learn to Code and Help Nonprofits
Learn other developer skills	Online resources: • Google Developer Training site
Learn cryptography	Online resources: • Cryptography, Stanford, Coursera • Applied Cryptography, Udacity
Work on projects outside of the classroom	Create and maintain a website, build your own server, or build a robot. Online resources: • Capstone project: Analyzing (Social) Network Data - scroll down to bottom of page, UCSD, Coursera • Capstone project: Java Programming: A DIY Version of Netflix and Amazon Recommendation Engines, Duke University, Coursera • Project Directory, Apache • Google Summer of Code Project Archive
Work on a small piece of a large system (codebase), read and understand existing code, track down documentation, and debug	GitHub is a great way to read other people's code or contribute to a project. Online resources: • How to contribute to Open Source • Hacktoberfest • GitHub • GitLab
Work on projects with other programmers	This will help you improve your ability to work well in a team and enable you to learn from others.
Practice your algorithmic knowledge and coding skills	Practice your algorithmic knowledge through coding competitions like Code Jam or ACM's International Collegiate Programming Contest. Online resources: • Code Jam • Kickstart, a Code Jam competition, is for university students looking to develop their coding skills and pursue a Google career • ACM ICPC
Become a teaching assistant	Helping to teach other students will help enhance your knowledge of the subject matter.
Gain internship experience in software engineering	Find Google's internships in Engineering and Technology on our Students site.
Prepare for the interview	Online resource to prepare to interview for software engineering positions, including for internships:

- Mastering the Software Engineering Interview, UCSD, Coursera
- Book: Cracking the Coding Interview
- Book: Programming Interviews Exposed

Articles

There was no article section in the original guide, but these are relevant, recent and excellent.

- How to land a top notch tech job as a student.
- Why I studied full-time for 8 months for a Google interview.