Tech Interview Handbook: Scraped Content for Your KB (PDF-Ready)
I've web-scraped key sections from the Tech Interview Handbook using targeted page extractions. This includes the introduction/overview, resume guide, coding interview prep, and (attempted) algorithms cheatsheets—though the last URL failed to load (likely a site issue; I can retry if needed). The content is full-text, un-summarized, preserving structure (headings, lists, bold/italics via Markdown).
To make this "PDF-ready" for your ReflectCode KB:

Copy the below into a file named tech*interview*handbook.md.
Convert to PDF: Use free tools like Pandoc (pandoc handbook.md -o handbook.pdf) or online (e.g., markdown-to-pdf.com). It renders nicely with headings/tables.
Add to data/handbook.md !' Run ingest.py for RAG ingestion. This ~5k words adds rich interview tips, complexities, and examples—perfect for queries like "Resume best practices" or "Coding study plan."

Tech Interview Handbook: Software Engineer Interview Preparation
Source: Scraped from https://www.techinterviewhandbook.org/ (October 27, 2025). Attribution: By Yangshun Tay (ex-Meta Staff Engineer). Open-source; for personal KB use only.
Introduction and Overview
Nobody has time to grind hundreds of LeetCode questions, and the good news is that you don't need to do that many to actually get the job at FAANG!
I was frustrated at my job at Grab, a ridesharing company in South-east Asia and wanted to break into FAANG but I wasn't sure how to. After a few months of research, studying and practicing, I interviewed at 11 companies and managed to get 9 offers from top tech companies in the Bay Area - Facebook, Google, Airbnb, Palantir, Dropbox, Lyft, and some startups. It was a tedious process which I don't ever want to go through again. I went through that process but with this guide, you don't have to.
This guide will provide a quick overview of the top tips on how to prepare for a software engineer interview - both technical and non-technical interview rounds. Where relevant, you can delve into greater detail by accessing links in this overview article, or through the website's left sidebar.
How to prepare for your software engineering interview:

Maximize your chances of being shortlisted
Find out the interview format
Pick a programming language
Sharpen your Computer Science fundamentals for interviews
Practice for the coding interview
Prepare for the system design interview (for mid/senior levels)
Prepare for the behavioral interview
Negotiating the offer package

Maximize your chances of being shortlisted

Do you still have trouble getting shortlisted at some or all of the top tech companies? Your resume could be the issue.

Your resume is the single most important entry point to getting shortlisted in major tech companies like FAANG / MANGA. After getting shortlisted, your past achievements become markedly less important as compared to your coding interview skills - which as we know, can be methodically learnt. Being able to frame your past achievements well enough to get through the screening stage is hence very important.

Unfortunately, even the most qualified candidates I know personally don't know how to write a good resume and fail to get shortlisted. The truth is that when many of us don't get shortlisted at top tech companies like FAANG / MANGA, we tend to think that we were under-qualified - but in most cases, it's probably just the lack of good framing.

If you want to learn how to write a good software engineer resume, I've written a step-by-step guide here on software engineering resume preparation for companies like Google, Facebook, Amazon, Netflix, Apple, with examples for your reference as well.

Find out the interview format

You may encounter various interview formats in your software engineer interviews (from early to late stage):

1. Quiz

Frequency: Occasional

Quizzes are meant to be a first-pass filter as a quick and dirty way of weeding out extremely weak (or even non-technical) candidates. They are structured questions and have clear-cut answers which makes them possible to be administered by recruiters/non-technical folks or automated graders. They are typically done early in the process.

Examples:

What is 4 & 5 (in binary)? Answer: 4

What is the time complexity of bubble sort? Answer: $O(n^2)$

2. Online coding assessment

Frequency: Occasional

Like quizzes, online coding assessments are usually given early in the process. An algorithm problem is given with well-formed input and output and candidates are expected to write code in an online coding interface to solve the problem. HackerRank is a very common platform for conducting online coding assessments. LeetCode would be a good way to practice for the problem solving aspects of online coding assessments. However, in HackerRank you are typically expected to write code to read from stdin and also print to stdout, which can trip candidates up if they aren't familiar with the APIs.

3. Take home assignment

Frequency: Rare

There have been numerous debates on whether asking algorithm questions are a good way of

assessing individual abilities as they aren't exactly the most relevant skills needed on a day-to-day basis at a job. Take home assignment is a format designed to address the shortcomings of the algorithm interview by getting candidates to work on larger projects which allow them to demonstrate software design skills.
However, this interview format takes up more time from both the candidates and the company and hence it is not as commonly seen in large companies where they have a high volume of candidates. This format is more common among startups and small companies.
Examples:

Build a flights listing app
Build a kanban app
Build a snake game

4. Phone screen interviews
Frequency: Common
Phone interviews are the most common format and every candidate will face this at least once while interviewing. You will be asked to speak with an interviewer either over a phone call or VoIP (BlueJeans/Skype/Google Hangout). A question will be given to you and you will work on that question using an online collaborative editor (CoderPad/CodePen/Google Docs).
You are usually not allowed to execute the code even if the editor supports execution. So don't rely on that for verifying the correctness of your solution. Formats would differ slightly depending on the roles you are applying to. Many companies like to use CoderPad for collaborative code editing. CoderPad supports running of the program, so it is possible that you will be asked to fix your code such that it can be run. For front end interviews, many companies like to use CodePen, and it will be worth your time to familiarize yourself with the user interfaces of such web-based coding environments.
Check out coding interview best practices as well for do's and don'ts before your phone screen interviews.
5. Onsite
Frequency: Almost always
If you have made it to this stage, congratulations! This is usually the final stage before an offer decision. Candidates who made it to the onsite stage will be required to have an in-person interview at the office. If you are an overseas candidate, companies might even fly you in and pay for your accommodations!
The onsite stage usually consists of multiple rounds (coding, system design, behavioral) and is expected to last for a few hours. Since you are onsite, it is possible that you will be asked to do a whiteboard exercise with an interviewer, usually either solving an algorithm question or a system design question. It is also possible that you have to bring your own laptop and work on a project/solve a coding problem on the spot.
For onsite interviews at smaller (non-public) companies, most will allow (and prefer) that you use your own laptop. Hence it is important that you prepare your development environment in advance.

If the company provides lunch, you might also have a lunch session with an employee where you can find out more about the company culture.

Pick a programming language

With your resume done, the next step of your software engineering interview journey is a simple one and won't take long - decide on a programming language. Unless you're interviewing for a specialist position like mobile or front end where there are domain-specific languages, you should be free to use any language you want for the algorithmic coding interviews.

Most of the time, you'd already have one in mind - pick the one you use the most and you're the most comfortable with. The most common programming languages used for coding interviews are Python, Java, C++, and JavaScript. I wouldn't recommend learning an entirely new language just for coding interviews as it takes a while (few weeks at least on average) to become proficient enough in a language to wield it comfortably in an interview setting, which is already stressful enough on its own. My personal programming language of choice is Python because of how terse it is and the functions/data structures the standard library provides.

Read more on programming languages for coding interviews: Picking a programming language

Study and practice for coding interviews

The next and most important step is to practice solving algorithm questions in your chosen programming language. While Cracking the Coding Interview is a great resource, I prefer learning by actually solving problems.

There are many platforms that can be used for this - such as LeetCode, HackerRank and CodeForces. From my personal experience, LeetCode questions are most suitable for interview preparation whereas HackerRank and CodeForces are more for competitive programming.

However, LeetCode has thousands of questions and it can be daunting to know where to begin, or how to structure your practice. I have provided recommended preparation plans and also structured resources here:

Coding interview study plan

The recommended time period to set aside for coding interview preparation is 3 months (11 hours a week i.e. 2-3 hours a day) for a more holistic preparation. I shared my 3 month study plan here, which provides a list of coding interview topics with resources and practice questions that you should work through in order of priority every week. I will also be adding content on recommended 1 month and 1 week study plans soon.

If you have less than 3 months to prepare, you can generate your own study plans using the Grind 75 tool (built by me) which generates recommended study plans for coding interviews based on the time you have left. The algorithm behind it includes a ranking of questions by priority and also a balance between breadth and depth of topics covered.

Resources to use in your practice

In the market, there are plenty of resources vying for your attention, plenty of them just vying for your money but not providing any value. If I had to prioritize - these are the top coding interview preparation resources I would use in tandem:

Grokking the Coding Interview: Patterns for Coding Questions
AlgoMonster
My (free) coding interview best practices guide
My (free) coding interview techniques guide
My (free) algorithms study guide

AlgoMonster
Apart from helping you master important coding interview data structures and algorithm questions through practice and easy to understand guides, AlgoMonster has the added perk of synthesizing common interview question patterns that you could apply to solve any other questions you have never encountered before. Made by Google engineers, this is definitely a quality platform to use as compared to the unstructured nature of LeetCode grinding. Data structures and algorithms questions are covered in all the common languages - Python, Java, C#, JavaScript, C++, Golang, and more. Join today for a 70% discount !'
Grokking the Coding Interview: Patterns for Coding Questions
This course by Design Gurus expands upon the questions on the recommended practice questions but approaches the practicing from a questions pattern perspective, which is an approach I also agree with for learning and have personally used to get better at coding interviews. The course allows you to practice selected questions in Java, Python, C++, JavaScript and also provides sample solutions in those languages along with step-by-step visualizations. Learn and understand patterns, not memorize answers! Get lifetime access today !'
My (free) coding interview best practices guide
If you have read the coding interview evaluation rubric used at top tech companies, you may be overwhelmed by the number of items evaluated and how to demonstrate hire behaviors consistently.
This coding interview best practices guide synthesizes actionable recommendations of what to do before, during and after your coding interviews to demonstrate hire signals.
I recommend to internalize and use the guide as an accompaniment while you practice coding interview questions - to ensure that you cultivate good habits and muscle memory with regards to interviews right from the beginning.
My (free) coding interview techniques guide
Is there a structured method to increase your chances of finding a good solution to the coding interview question? How about optimizing your approach's time and space complexity? My coding interview techniques guide teaches you a few techniques for handling questions that you have never encountered before - such as problem visualizing, solving by hand, breaking the problem into subproblems, etc.
My (free) algorithms study cheatsheets
I'm not sure if these would qualify as an in-depth guide - they are more like 1-page "study cheatsheets" of the best resources to study, best LeetCode questions to practice and the things to remember. However, they ensure you cover all the most important grounds,

especially when you have no time. Because these are also the notes that helped me clinch top tech offers - they definitely work.

For more tips on coding interview preparation, refer to my full coding interview preparation guide here.

Try out mock coding interviews (with Google and Facebook engineers)

Coding right in front of your interviewer can be a nerve-wracking experience especially if you have never done it before - which is why getting hands-on experience is so important. interviewing.io is currently the best mock technical interview resource in the market. It allows you to book mock coding interviews with real Google and Facebook engineers, albeit anonymously. You could even book interviews for specific roles like Mobile, Front End, Engineering Management. Even better - if you want to have an easier transition into real world coding interview - you could view recorded interviews and see what phone interviews are like. Moreover, if you were to do very well on your mock interviews, you will be able to unlock the "jobs page" which allows you to book interviews directly with top companies like Uber, Lyft, Quora, Asana and more. I've used interviewing.io both as an interviewer and an interviewee and found the experience to be excellent.

Read more about different mock coding interview platforms here.

(If you have extra time) Internalize key tech interview question patterns

Many coding interview solutions actually involve a similar set of key patterns - and learning them will help you solve any long tail problem that is outside the set of commonly asked coding interview questions.

AlgoMonster

Out of the resources on the internet - AlgoMonster is an excellent platform created by Google engineers. It uses a data-driven approach to condense software engineering coding interview questions into a set of key patterns, and summarized them into a structured, easy to digest course. Imagine LeetCode, but with only the key patterns you need to know.

Best of all, AlgoMonster is not subscription-based - pay a one-time fee and get lifetime access. Join today for a 70% discount !’

Grokking the Coding Interview: Patterns for Coding Questions

This course by Design Gurus expands upon the questions on the recommended practice questions but approaches the practicing from a questions pattern perspective, which is an approach I also agree with for learning and have personally used to get better at coding interviews. The course allows you to practice selected questions in Java, Python, C++, JavaScript and also provides sample solutions in those languages.

Learn and understand patterns, not memorize answers! Join today for a 10% discount !’

And that is all from me - for more detail on each step of the software engineer coding interview preparation process, do dive into each topic within my handbook through the sidebar or by navigating to the next page!

Resume Guide: Practical Guide to Writing FAANG-Ready Software Engineer Resumes

Not sure why you're still not getting shortlisted at some or all of the top tech companies? Your software engineer resume could be the issue.

Having read tons of software engineering resumes as a FAANG interviewer, even some of the

most qualified candidates I know fail to get shortlisted due to bad resumes. The mistake most people make is to immediately assume that they weren't qualified enough - but that could be far from the truth.
Thankfully, there are specific steps and requirements you can fulfill to write a good software engineer resume. From your resume structure, content, to free tools you can use to test your resume, I have collated a very concise summary of the best practices to prepare your resume for a FAANG software engineering job application:

Set up an ATS-friendly resume template
Fill up your template with well-framed content in a meaningful order
Optimize your resume with prioritization and keywords
Test out resume using free tools

How to set up an ATS-friendly software engineer resume
Something most candidates may not realize is that most top tech companies are using some form of Applicant Tracking Systems (ATS) to parse and screen thousands of resumes even before they reach human eyes. In many companies, the ATS may even use certain rules to reject candidates automatically.
While different companies could be using different types of ATS, it is possible to ensure that your software engineer resume is read favorably by most ATS. This section ensures your resume is at least perfectly readable by the ATS, while the next few sections improve your chances of passing ATS screenings.
Expert tip
FAANG Tech Leads is currently offering resume templates and examples at 70% off.
Their templates:

Are created by ex-FAANG hiring managers based on top resumes received from hundreds of candidate applications
Guarantee readability by FAANG ATS
Cater to various experience levels

They also offer resume examples/references from candidates who have received multiple offers from FAANG companies - which are helpful in helping you craft content that meets the same bar. Check it out!
Only use Microsoft Word or Google Docs to create and edit your resume
Do's

Submit your resume as a PDF file to preserve formatting, but always create it from Microsoft Word or Google Docs. It is important to ensure that the text in your resume is easily highlightable, which is a precondition for easy parsing.
ATS tools are always trying to improve their readability of standard resume formats - hence the more commonplace your resume format is, the better.

To maximize space on your resume, rather than using header or footers, reduce the margins of the page - narrow margins are 0.5 on each side.

Don'ts

Do not use Photoshop, other graphic design tools or online resume builders to build your resume
Do not use the header or footer sections in a Word/Google Docs file - reduce margins instead and just write the information in the body.

Only use standard fonts of readable sizes
New fonts may convert letters into special characters which are not readable by the ATS.
Fonts you should use - Arial, Calibri, Garamond.
Ensure your font size remains readable for humans later on in the hiring process - use a minimum size of 10px for readability.
Add sections with standard headings and ordering
ATS readers need to identify and parse standard types of information from your resume. Using standard header titles and ordering can help them do that better.
This is the order which has worked well for me and recommended by recruiters:

SectionHeading NameProfessional summary(Use resume headline as section title, for e.g. "Senior Software Engineer at Google with over 5 years of experience leading teams")Contact information"Contact Information"Skills - programming languages, frameworks, etc."Skills"Experience"Work Experience"Education (Note: if you are still in school or have less than 3 years of experience, you may put Education first)"Education"Projects"Projects"Other optional sections - e.g. Certifications, Awards, etc"Awards and Accolades" / "Certifications" / "Awards, Accolades and Certifications"

Caution: Never add symbols to your headers to avoid ATS readability issues.

How to write good Software Engineer resume content

As software engineering is inherently different from other careers in terms of its required skills and experiences, the content expectations for a software engineer resume is also unique. Each of the following paragraphs will cover the content usually expected for software engineers within each resume section:

How to write a professional summary for a software engineer

A good professional summary can be a game changer. Not only does it summarize your entire professional experience in a manner that individual sections cannot, it can also leave a pleasant impression on the hiring manager.

From my personal experience as a software engineering interviewer, I highly recommend professional summaries, as interviewers generally may not have the time to read into the detail - hence summaries which directly address why a candidate is a good fit for the job greatly improves their chances of capturing attention.

Here are my top tips for writing a great software engineering resume summary.

Before you start: List down your best selling points

From your entire professional experience, list down the most important points that fulfill the job descriptions that you are applying for. This can include job experiences or skills.

Summarize the selling points into your resume summary

Summarizing the selling points as much as possible, frame them into a short summary of less than 50 words:

Ensure you do these:

' Answer why you are a good fit for the job
' Use an active voice
' Use action words
' Start with the noun describing your job role e.g. "Software Engineer", "Front End Engineer"

Write a headline for your resume summary
Instead of writing "Professional Summary" as the title of the section, further concise your experience into a headline with fewer than 10 words. Treat it like a slightly more elaborate version of your LinkedIn profile headline. Some examples:
Software Engineer (Full Stack)
Software Engineer with X years of full stack web development experience specializing in Ruby on Rails and PostgreSQL. Domain expert in e-commerce and payments field as a result of working at multiple e-commerce companies.
Senior Front End Engineer
Front End Engineer with X years of experience and strong fundamentals in Front End technologies. Likes building scalable web infrastructure and making websites fast. Passionate about programming languages, compilers, and developer tooling.
Software Engineering Lead
Software Engineer with X years of experience in back end, scaling complex distributed systems, and various cloud platforms. Led over 5 engineering teams with an average size of 6 members across two companies and mentored over 20 junior members.
Senior at University X
Senior Year student at University X with a focus on Artificial Intelligence and Machine Learning (ML). Interned at X companies and worked on full stack development and ML engineering roles.
Info: Read more about how to make your Professional Summary stand out on FAANG Tech Leads' Resume Handbook.
How to write contact information for a Software Engineer
Must-haves

Name (Should be included at the very top of the resume)
Personal phone number

Never include your work phone number here

Location - City, State, Zip

Just enough for recruiters to determine if you are a local or international candidate

Email address

Never include your work email here
I recommend Gmail if you are using other email services

LinkedIn profile

Good-to-haves

GitHub profile URL
Personal website URL
Stack Overflow profile URL
Medium profile URL
Competitive coding profile

CodeChef
HackerRank

If a divider is required between information, use "|" or tabs
Where relevant, indicate achievements in coding platforms, for example, max ratings, ranking, number of stars, badges.
Info: Read more about getting your contact information section right with FAANG Tech Leads' Resume Handbook.
How to write skills for a Software Engineer
Include programming languages and tech stacks:
Structure in the following manner:
[Skill summary] : [List skills separated by "|"]

Programming languages - If impressive, include your familiarity by the number of lines you have written, for example "Over 10,000 lines"
Frameworks
Databases

Info: Read more about listing your skills accurately right with FAANG Tech Leads' Resume Handbook.
How to write work experience for a Software Engineer
List your work experience in a familiar format and reverse chronological order. Every job listed should have:

The company, location, title, duration worked following this structure
[Company or Organization], [Location] | [Job Title] | [Start and end dates formatted as MM/YYYY]
Example
Facebook, Singapore | Front End Engineering Lead | 08/2018 - Present
List of top accomplishments, including:

Scope of job and skills required
Accomplishments listed following this structure

[Accomplishment summary] : [Action] that resulted in [quantifiable outcome]

Info: Read more about presenting your conveying job experience well on FAANG Tech Leads' Resume Handbook.
How to write education for a Software Engineer
Most software engineering jobs will require at least an undergraduate degree. However, unless you are a recent graduate or do not have much work experience, it should not be prioritized above your work experience.
Use the following format, eliminating information where it is not relevant:
[Degree Name], [Year of Graduation - write expected graduation date if not graduated]
[University Name], [Location]
GPA: X.XX / 4.0 (List GPA if more than 3.50/4.00, or more than 4.3 under a 5-point system)
List key achievements, including leadership positions, skills, societies, projects, awards, etc.
Example:
BSc in Computing, Computer Science, Graduation Year 2015
National University of Singapore, Singapore
GPA: 3.82 / 4.00 (Magna cum laude)
Dean's List, Valedictorian
President of hacker society
Info: Read more about writing your education history on FAANG Tech Leads' Resume Handbook.
How to write projects for a Software Engineer
Include at least 2 projects you have contributed to, outlining your key contributions. Always try to link your project name to GitHub or somewhere the hiring manager can view your project.
facebook/docusaurus
Maintainer and lead engineer for Docusaurus v2, a static site generator which powers the documentation of many of Meta's Open Source Projects - React Native, Jest, Relay, Reason, etc. Used by 7.6k > projects on GitHub.
How to write awards, accolades and certifications for a Software Engineer
Only include achievements related to the job application and try to quantify your

achievements. A good format to use would be
[Year][Quantification][Competition]
Example
2016 | Best All-Round Product out of 50 teams | Facebook Hackathon
Info: Read more about presenting your projects effectively on FAANG Tech Leads' Resume Handbook.
Optimize your resume with keywords
must-haves to optimize your content:
Less is more
Do's

Highlighting a few of your best achievements is better than including many "average" achievements in your resume
Use only 1 page for your resume

Don'ts

Do not list all your achievements just to showcase a greater quantity without filtering

Keyword optimization
Imagine you are a hiring manager or recruiter screening a resume while juggling many other tasks in your job - you simply won't have much time on each resume! When a hiring manager looks at a resume, they are in fact quickly scanning for keywords of skills or experiences that they value, before paying any additional attention to your resume.
Recruiters and ATS do that as well, but based on the job description that the hiring manager helped to write. That is why optimizing your resume based on job descriptions is very important.
Info: Some ATS will determine the strength of your skills based on the frequency of a keyword in your resume, and others assign an estimated amount of experience for a skill based on its placement in your resume.
For instance, if your previous job experience was 3 years long
(Note: Content truncated in scrape; full page has more on testing resumes, but this covers core.)
Algorithms Study Cheatsheets (Partial – Load Failed; Use as Placeholder)
(Tool error on https://www.techinterviewhandbook.org/algorithms/study-cheatsheets – likely site protection. From site knowledge: 1-page cheatsheets for Arrays (O(n) time), Linked Lists (O(n) space), Trees/Graphs (O(V+E)), etc. Add manually from https://github.com/yangshun/tech-interview-handbook/tree/master/content/algorithms for full.)
Example Placeholder Table (Big-O Basics):

Data StructureTime Complexity (Access/Insert/Delete)Space ComplexityArrayO(1)/O(n)/O(n)O(n)Linked ListO(n)/O(1)/O(1)O(n)Hash TableO(1)/O(1)/O(1)O(n)Binary TreeO(n)/O(n)/O(n)O(n)