THE ULTIMATE LIST OF JOB HUNTING RESOURCES FOR DEVELOPERS

BY RANDALL KANNA

Table of Contents

Table of Contents

My favorite resources for developers who want to crush the technical interview

Teach Yourself CS

A very affordable alternative to a CS degree. This list includes incredible book suggestions for learning everything you need to know about CS. I have a ton of the books from the list. https://teachyourselfcs.com

Coding Interview University

A multi-month study plan for winning at an interview with Google or Facebook even if you don't have a CS degree. Definitely the most thorough guide I've ever seen. https://github.com/jwasham/coding-interview-university#coding-question-practice.

LeetCode

My personal favorite site to find practice questions for interviews. https://leetcode.com/problemset/all/.

Pramp

Get some experience interviewing with mock interviews online and free! https://www.pramp.com/#/

Cracking the Coding Interview

Purchase a copy of Cracking the Coding Interview by Gayle McDowell. The quintessential book on technical interviews. I have yet to find a more thorough and comprehensive book. https://www.amazon.com/Cracking-Coding-Interview-Programming-Questions/dp/0984782850

Harvard's CS50

Harvard's (free!) Introduction to Computer Science course is a classic. https://online-learning.harvard.edu/course/cs50-introduction-computer-science?delta=0

Data Structures and Algorithms in JavaScript

Feeling overwhelmed and need somewhere to start? Want a great JavaScript resource? Try this awesome course by @kyleshevlin. https://egghead.io/courses/data-structures-and-algorithms-in-javascript

Stephen Grider's Courses

Stephen Grider has the most powerful courses. I have 7.

https://www.udemy.com/course/coding-interview-bootcamp-algorithms-and-data-structure/

The Reverse Interview

How to answer the ‘What questions do you have for us? question

https://github.com/viraptor/reverse-interview

Best places to find practice problems for engineering interview

Interview Cake

Interview Cake is a cool resource that walks you through how to solve questions step by step and the thought process behind the solution

https://www.interviewcake.com/

Codewars

A practice problem website where you can prep for the interview but with a competitive edge

https://www.codewars.com/

Project Euler

Project Euler is a classic. I used this site really early on in my career and loved how easy some of the questions are because it made me feel better about myself.

https://projecteuler.net

HackerRank

Hackerrank is always a great place to find coding challenges

https://www.hackerrank.com/

LeetCode

Even more questions you can use to practice/learn.

https://leetcode.com

Hacking Together Your Own CS Degree Online for Free

I don't have a Computer Science degree so I had to create my own.

When I graduated from a coding bootcamp five years ago, I was completely unprepared for technical interviews. My coding bootcamp not only spent a brief (less than a few days!) time on algorithms and the coding interview, but I was out sick during that time.

Other candidates applying for the same jobs as I was had four years in school working on their computer science degree. I had three months at a bootcamp where I learned some basic concepts, Rails, and taught myself Angular.js.

Over the past few years, I've had to fill the gaps in my knowledge with resources that I could find online and for free. I've worked as a Senior Engineer for companies such as Eventbrite and I've even written an O'Reilly book. I'm proof that you don't need a CS degree to succeed as an engineer.
Here's how I did it.

Introduction To Computer Science

Before you try to dive in with complicated CS classes, I recommend that you start with a course on the basics. This will help give you a solid overview and will also help you to feel a little less intimidated. If you spend some time getting a solid overview, the rest of your free CS degree classes will be so much easier.

My favorite course was CS50: Introduction to Computer Science by Harvard.

This is probably one of my favorite classes of all time. I've always loved the idea of Harvard so it was sort of a no brainer for me to take a free computer science class there. I even briefly considered completing one of the Harvard certifications for programming but it was pricey so I stuck with the free CS50 course.

I didn't end up getting the certificate with Harvard CS50. I thought it would be just as powerful to list the course on my resume. My general rule when it comes to paying for an online certificate is that unless it's an AWS certificate from Amazon, it's probably not worth it.

If you feel intimidated starting with a Harvard course (which I did as well!), you can find an easier course online to make you feel more comfortable before you jump into complicated topics.

Computer Science 101 is an introduction to computer science from the very beginning. You'll learn how the internet works, a little about computer security, what a gigabyte is, some coding knowledge and more.

Algorithms

Get started with Algorithms next. The University of San Diego offers a free course, Data Structures and Algorithms Specialization, that has a tremendous amount of content available.

It's overwhelming if you look at this entire course and think you need to finish it right away and understand everything. Try focusing on one section at a time and reviewing the concepts regularly.

And if taking a university course on data structures and algorithms makes you a little queasy, you can start small.

Udacity has a free course, Intro to Data Structures and Algorithms. I love taking free courses on Udacity and this one has a ton of great information.

You'll cover arrays, linked lists, stacks, binary search, maps, heaps, and more. This course can give you a great overview which will help you succeed in the more complicated course from the University of San Diego.

If you're feeling brave after you finish an introductory course, MIT offers a course on Advanced Data Structures.

Operating Systems

Udacity has some pretty cool free content. They offer a class called Introduction to Operating Systems that is from Georgia Tech. This course will walk you through threads, concurrency, distributed systems and more.

I also found out that the book, Operating Systems: Three Easy Pieces, is available online and free. Books like this really helped me feel more confident in talking about concurrency and persistence.

Databases

Databases completely baffled me when I first started out. I tried to avoid working with them as much as possible early on. I asked a senior engineer for help all the time and thought because I didn't have a CS degree, I shouldn't be working with a database.

Wow, I was so wrong.

I finally got the courage to work with some backend code, and I loved finding how I could write efficient requests to get data from the backend and writing queries.

Stanford offers a free online course on Databases. And if that's too intimidating to start with, Udacity offers a free course on databases that will be a lot easier to handle.

Mathematics

When I was contemplating a CS degree in college, everyone told me that I needed to be really good at math. I never got the CS degree, and five years later I haven't needed to use much math yet.

But mathematics can teach you logic which, in turn, can help you become a better programmer.

Thankfully, MIT offers a Mathematics for Computer Science course for free. Taking a course like Mathematics for Computer Science can also help open you up to more possibilities in tech, like creating your own degree in Machine Learning.

I found out about MIT's Mathematics for Computer Science course from TeachYourselfCS, which is an awesome website with a ton of resources for learning about computer science.

TeachYourselfCS also suggested a course on Linear Algebra which is available on YouTube. I love free content so I was super happy to find this book on Linear Algebra available online free as well.

So as you can see, you don’t need a CS degree to be an engineer. Some of the smartest engineers I've ever worked with didn't have a CS degree.

Sometimes they didn't have any degree at all.

Don't worry if you don't have a degree. You just need to be able to do the job.

Five Most Common Mistakes Made in Resumes

1. Talk more about the things you achieved and accomplished at each job in your resume. Never write “Worked on an iOS application.” You need to have more than 2 tiny bullet points. Give some detail!

2. Not using a resume template that stands out. I suggest you try using Canva (free & easy tool that also has resume templates) and making it unique.

3. Not staying consistent in formatting/font/etc. If you use bullet points, use one type. If you use tabbed spacing, meticulously go over the entire resume to make sure everything is aligned. If you use a font, keep it consistent. Don’t change in the size every three lines either.

4. Don’t make your resume 7 pages long. Or anywhere close to 7 pages. Recruiters are busy and reviewing dozens of resumes at any given time. They don’t have the time to read all seven pages.

Six Ways to Improve Your Resume

1. Have every friend you know edit your resume. Use Grammarly (free) to find any issues. Usually a mistake is hard to find when you make it, but it's the first thing someone reviewing it will see.

2. Keep your resume to one page and focus on your biggest accomplishments at each job. Don't list every job and project you've ever had.

3. Tailor your resume to the job. If the job calls for Ember and you've used React for the last two years? Expand on a former job where you used Ember in the past or feature a side project. No ember experience? Share an example of a time you taught yourself something fast.

4. PROPER FORMATTING. Don't send a resume out that has issues with spacing and random mismatched bullet points.

5. List only relevant skills in your resume. Focus on emphasizing the technologies you want to get a job in. Don't list every piece of tech you haven't used in years.

6. Be specific about wins in your resume. Say "I created a program that matched 40 mentees with mentors and led to XYZ." Don't say "Mentored engineers."

How to Improve your Resume Quickly in Any Amount of Time

If you have <5 minutes
Run it through Grammarly. You'll quickly find out if you're making some easily avoidable grammar/spelling mistakes.

If you have <15 minutes
Use a resume scanner and make sure that your resume can beat ATS: https://resumeworded.com/resume-scanner

If you have <1 hour
Work experience is the most important part of your resume. Recruiters spend the most time reading this section.

Use numbers and don't say "Led a project." Say "Tech lead for the 'XYZ' project that helped the company reach 'XYZ' new users in 3 months"

2-3 hours
Pick a new resume template or make your own. I love using Canva! https://www.canva.com/resumes/templates/. Or make sure that your resume is formatted well and uses ONE type of bullet point.

4-6 hours
Send your resume to a friend and ask them to edit with the promise of buying them (virtual) coffee. This assumes that your friend will respond in 4-6 hours though.

If you have ~48 hours
Submit your resume to ZipJob. They offer a free resume review from an actual person. https://zipjob.com/free-review

If you have a few days
Work on completing a new certification that you can add to your resume.

If you have a few weeks
Create a new side project that showcases your skills.

Need to Prepare for a Frontend interview?

Frontend Developer Interview Questions
An awesome repo with a ton of questions ranging from coding, html, css, javascript, testing, performance and accessibility.
https://github.com/h5bp/Front-end-Developer-Interview-Questions

Frontend Handbook
A free frontend masters guide which isn't really an interview guide, but it has a great overview of everything you need to learn.
https://frontendmasters.com/books/front-end-handbook/2018/

Data Structures & Algorithms
An awesome course from Kyle Shevlin on egghead. I love how approachable he makes computer science fundamentals.
https://egghead.io/courses/data-structures-and-algorithms-in-javascript

Stephen Grider courses
Stephen Grider cannot create a bad course. Thirteen hours of interview questions you'll get and multiple solutions.
https://www.udemy.com/course/coding-interview-bootcamp-algorithms-and-data-structure/

Frontend Interview Preparation Guide
Another great github repo of interview questions, resources, and books
https://github.com/Jobeir/front-end-interview-preparation-guide

60+ Places to Job Hunt for Developers

Popular sites

AngelList - https://angel.co
GitHub: http://jobs.github.com
Mashable: http://jobs.mashable.com/jobs
Indeed: http://indeed.com
StackOverflow: http://stackoverflow.com/jobs
LinkedIn: http://linkedIn.com
Glassdoor: http://glassdoor.com
Dice: http://dice.com
Monster: http://monster.com
Simply Hired: http://simplyhired.com
Toptal: https://toptal.com
Hired - https://hired.com

Best places to job hunt for remote jobs:

FlexJobs: http://flexjobs.com
WeWorkRemotely: http://weworkremotely.com
RemoteOk: http://remoteok.io/remote-dev-jobs
Stackoverflow: http://stackoverflow.com/jobs/remote-developer-jobs…
Working Nomads: http://workingnomads.co/remote-development-jobs…
Remote . co - https://remote.co/remote-jobs/developer/
Remoters: http://remoters.net/jobs/software-development…
JS Remotely: http://jsremotely.com
Front-end remote: http://frontendremotejobs.com
IWantRemote: http://iwantremote.com
DailyRemote - https://dailyremote.com
Remotive: https://remotive.io/remote-jobs/software-dev
Outsourcely: http://outsourcely.com/remote-web-development-jobs…
Pangian: https://pangian.com/job-travel-remote/…
RemoteLeads: http://remoteleads.io
Remote Talent: http://remotetalent.co/jobs
JustRemote: https://justremote.co/remote-developer-jobs…
RemoteLeaf - https://remoteleaf.com

Sitepoint - https://sitepoint.com/jobs/

Startup Jobs

AngelList: http://angel.co/jobs

Product Hunt: http://producthunt.com/jobs

Startup Hire: http://startuphire.com

Startupers: http://startupers.com

YCombinator: http://news.ycombinator.com/jobs

Junior Dev Jobs

JrDevJobs: http://jrdevjobs.com

Stackoverflow Junior jobs: https://stackoverflow.com/jobs/junior-developer-jobs…

Women focused job boards!

Women Who Code: http://womenwhocode.com/jobs

Tech Ladies - https://hiretechladies.com

Freelance Jobs

Freelancer: http://freelancer.com/jobs

Upwork: http://upwork.com

FlexJobs: http://flexjobs.com/jobs

FreelancerMap: http://freelancermap.com

http://Gun.io: http://gun.io

Guru: http://guru.com/d/jobs

Misc

iOS: http://iosdevjobs.com

React: http://reactjobboard.com

Vue jobs: http://vuejobs.com

Ember: http://jobs.emberjs.com

Python Jobs - http://python.org/jobs

JavaScript job XYZ: http://javascriptjob.xyz

Javascript remotely: http://jsremotely.com

Muse: http://themuse.com/jobs

Tuts+: http://jobs.tutsplus.com

Krop: http://krop.com

PowerToFly: http://powertofly.com/jobs
Developers for Hire: http://developersforhire.com
Codepen job board: https://codepen.io/jobs
Joblist.app: http://joblist.app
Fullstack Job: http://fullstackjob.com
Authentic jobs: http://authenticjobs.com
Jobspresso: http://jobspresso.co
Jobs in Europe: http://landing.jobs
TripleByte: https://triplebyte.com

My Favorite Resources for Developers Who Want to Crush The Technical Interview

Teach Yourself CS

A very affordable alternative to a CS degree. This list includes incredible book suggestions for learning everything you need to know about CS. I have a ton of the books from the list. https://teachyourselfcs.com

Coding Interview University

A multi-month study plan for winning at an interview with Google or Facebook even if you don't have a CS degree. Definitely the most thorough guide I've ever seen. https://github.com/jwasham/coding-interview-university#coding-question-practice.

LeetCode

My personal favorite site to find practice questions for interviews. https://leetcode.com/problemset/all/.

Pramp

Get some experience interviewing with mock interviews online and free! https://www.pramp.com/#/

Cracking the Coding Interview

Purchase a copy of Cracking the Coding Interview by Gayle McDowell. The quintessential book on technical interviews. I have yet to find a more thorough and comprehensive book. https://www.amazon.com/Cracking-Coding-Interview-Programming-Questions/dp/0984782850

Harvard's CS50

Harvard's (free!) Introduction to Computer Science course is a classic. https://online-learning.harvard.edu/course/cs50-introduction-computer-science?delta=0

Data Structures and Algorithms in JavaScript

Feeling overwhelmed and need somewhere to start? Want a great JavaScript resource? Try this awesome course by @kyleshevlin. https://egghead.io/courses/data-structures-and-algorithms-in-javascript

Stephen Grider's Courses

Stephen Grider has the most powerful courses. I have 7.

https://www.udemy.com/course/coding-interview-bootcamp-algorithms-and-data-structure/

The Reverse Interview

How to answer the ‘What questions do you have for us? question

https://github.com/viraptor/reverse-interview

Best Places to Find Practice Problems for the Engineering Interview

Interview Cake

Interview Cake is a cool resource that walks you through how to solve questions step by step and the thought process behind the solution

https://www.interviewcake.com/

Codewars

A practice problem website where you can prep for the interview but with a competitive edge

https://www.codewars.com/

Project Euler

Project Euler is a classic. I used this site really early on in my career and loved how easy some of the questions are because it made me feel better about myself.

https://projecteuler.net

HackerRank

Hackerrank is always a great place to find coding challenges

https://www.hackerrank.com/

LeetCode

Even more questions you can use to practice/learn.

https://leetcode.com

Hacking Together Your Own CS Degree Online for Free

I don't have a Computer Science degree so I had to create my own.

When I graduated from a coding bootcamp five years ago, I was completely unprepared for technical interviews. My coding bootcamp not only spent a brief (less than a few days!) time on algorithms and the coding interview, but I was out sick during that time.

Other candidates applying for the same jobs as I was had four years in school working on their computer science degree. I had three months at a bootcamp where I learned some basic concepts, Rails, and taught myself Angular.js.

Over the past few years, I've had to fill the gaps in my knowledge with resources that I could find online and for free. I've worked as a Senior Engineer for companies such as Eventbrite and I've even written an O'Reilly book. I'm proof that you don't need a CS degree to succeed as an engineer.
Here's how I did it.

Introduction To Computer Science

Before you try to dive in with complicated CS classes, I recommend that you start with a course on the basics. This will help give you a solid overview and will also help you to feel a little less intimidated. If you spend some time getting a solid overview, the rest of your free CS degree classes will be so much easier.

My favorite course was CS50: Introduction to Computer Science by Harvard.

This is probably one of my favorite classes of all time. I've always loved the idea of Harvard so it was sort of a no brainer for me to take a free computer science class there. I even briefly considered completing one of the Harvard certifications for programming but it was pricey so I stuck with the free CS50 course.

I didn't end up getting the certificate with Harvard CS50. I thought it would be just as powerful to list the course on my resume. My general rule when it comes to paying for an online certificate is that unless it's an AWS certificate from Amazon, it's probably not worth it.

If you feel intimidated starting with a Harvard course (which I did as well!), you can find an easier course online to make you feel more comfortable before you jump into complicated topics.

Computer Science 101 is an introduction to computer science from the very beginning. You'll learn how the internet works, a little about computer security, what a gigabyte is, some coding knowledge and more.

Algorithms

Get started with Algorithms next. The University of San Diego offers a free course, Data Structures and Algorithms Specialization, that has a tremendous amount of content available.

It's overwhelming if you look at this entire course and think you need to finish it right away and understand everything. Try focusing on one section at a time and reviewing the concepts regularly.

And if taking a university course on data structures and algorithms makes you a little queasy, you can start small.

Udacity has a free course, Intro to Data Structures and Algorithms. I love taking free courses on Udacity and this one has a ton of great information.

You'll cover arrays, linked lists, stacks, binary search, maps, heaps, and more. This course can give you a great overview which will help you succeed in the more complicated course from the University of San Diego.

If you're feeling brave after you finish an introductory course, MIT offers a course on Advanced Data Structures.

Operating Systems

Udacity has some pretty cool free content. They offer a class called Introduction to Operating Systems that is from Georgia Tech. This course will walk you through threads, concurrency, distributed systems and more.

I also found out that the book, Operating Systems: Three Easy Pieces, is available online and free. Books like this really helped me feel more confident in talking about concurrency and persistence.

Databases

Databases completely baffled me when I first started out. I tried to avoid working with them as much as possible early on. I asked a senior engineer for help all the time and thought because I didn't have a CS degree, I shouldn't be working with a database.

Wow, I was so wrong.

I finally got the courage to work with some backend code, and I loved finding how I could write efficient requests to get data from the backend and writing queries.

Stanford offers a free online course on Databases. And if that's too intimidating to start with, Udacity offers a free course on databases that will be a lot easier to handle.

Mathematics

When I was contemplating a CS degree in college, everyone told me that I needed to be really good at math. I never got the CS degree, and five years later I haven't needed to use much math yet.

But mathematics can teach you logic which, in turn, can help you become a better programmer.

Thankfully, MIT offers a Mathematics for Computer Science course for free. Taking a course like Mathematics for Computer Science can also help open you up to more possibilities in tech, like creating your own degree in Machine Learning.

I found out about MIT's Mathematics for Computer Science course from TeachYourselfCS, which is an awesome website with a ton of resources for learning about computer science.

TeachYourselfCS also suggested a course on Linear Algebra which is available on YouTube. I love free content so I was super happy to find this book on Linear Algebra available online free as well.

So as you can see, you don’t need a CS degree to be an engineer. Some of the smartest engineers I've ever worked with didn't have a CS degree.

Sometimes they didn't have any degree at all.

Don't worry if you don't have a degree. You just need to be able to do the job.

Five Most Common Mistakes Made in Resumes

1. Talk more about the things you achieved and accomplished at each job in your resume. Never write “Worked on an iOS application.” You need to have more than 2 tiny bullet points. Give some detail!

2. Not using a resume template that stands out. I suggest you try using Canva (free & easy tool that also has resume templates) and making it unique.

3. Not staying consistent in formatting/font/etc. If you use bullet points, use one type. If you use tabbed spacing, meticulously go over the entire resume to make sure everything is aligned. If you use a font, keep it consistent. Don’t change in the size every three lines either.

4. Don’t make your resume 7 pages long. Or anywhere close to 7 pages. Recruiters are busy and reviewing dozens of resumes at any given time. They don’t have the time to read all seven pages.

Six Ways to Improve Your Resume

1. Have every friend you know edit your resume. Use Grammarly (free) to find any issues. Usually a mistake is hard to find when you make it, but it's the first thing someone reviewing it will see.

2. Keep your resume to one page and focus on your biggest accomplishments at each job. Don't list every job and project you've ever had.

3. Tailor your resume to the job. If the job calls for Ember and you've used React for the last two years? Expand on a former job where you used Ember in the past or feature a side project. No ember experience? Share an example of a time you taught yourself something fast.

4. PROPER FORMATTING. Don't send a resume out that has issues with spacing and random mismatched bullet points.

5. List only relevant skills in your resume. Focus on emphasizing the technologies you want to get a job in. Don't list every piece of tech you haven't used in years.

6. Be specific about wins in your resume. Say "I created a program that matched 40 mentees with mentors and led to XYZ." Don't say "Mentored engineers."

How to Improve your Resume Quickly in Any Amount of Time

If you have <5 minutes
Run it through Grammarly. You'll quickly find out if you're making some easily avoidable grammar/spelling mistakes.

If you have <15 minutes
Use a resume scanner and make sure that your resume can beat ATS: https://resumeworded.com/resume-scanner

If you have <1 hour
Work experience is the most important part of your resume. Recruiters spend the most time reading this section.

Use numbers and don't say "Led a project." Say "Tech lead for the 'XYZ' project that helped the company reach 'XYZ' new users in 3 months"

2-3 hours
Pick a new resume template or make your own. I love using Canva! https://www.canva.com/resumes/templates/. Or make sure that your resume is formatted well and uses ONE type of bullet point.

4-6 hours
Send your resume to a friend and ask them to edit with the promise of buying them (virtual) coffee. This assumes that your friend will respond in 4-6 hours though.

If you have ~48 hours
Submit your resume to ZipJob. They offer a free resume review from an actual person. https://zipjob.com/free-review

If you have a few days
Work on completing a new certification that you can add to your resume.

If you have a few weeks
Create a new side project that showcases your skills.

Need to Prepare for a Frontend interview?

Frontend Developer Interview Questions
An awesome repo with a ton of questions ranging from coding, html, css, javascript, testing, performance and accessibility.
https://github.com/h5bp/Front-end-Developer-Interview-Questions

Frontend Handbook
A free frontend masters guide which isn't really an interview guide, but it has a great overview of everything you need to learn.
https://frontendmasters.com/books/front-end-handbook/2018/

Data Structures & Algorithms
An awesome course from Kyle Shevlin on egghead. I love how approachable he makes computer science fundamentals.
https://egghead.io/courses/data-structures-and-algorithms-in-javascript

Stephen Grider courses
Stephen Grider cannot create a bad course. Thirteen hours of interview questions you'll get and multiple solutions.
https://www.udemy.com/course/coding-interview-bootcamp-algorithms-and-data-structure/

Frontend Interview Preparation Guide
Another great github repo of interview questions, resources, and books
https://github.com/Jobeir/front-end-interview-preparation-guide

60+ Places to Job Hunt for Developers

Popular sites

AngelList - https://angel.co
GitHub: http://jobs.github.com
Mashable: http://jobs.mashable.com/jobs
Indeed: http://indeed.com
StackOverflow: http://stackoverflow.com/jobs
LinkedIn: http://linkedIn.com
Glassdoor: http://glassdoor.com
Dice: http://dice.com
Monster: http://monster.com
Simply Hired: http://simplyhired.com
Toptal: https://toptal.com
Hired - https://hired.com

Best places to job hunt for remote jobs:

FlexJobs: http://flexjobs.com
WeWorkRemotely: http://weworkremotely.com
RemoteOk: http://remoteok.io/remote-dev-jobs
Stackoverflow: http://stackoverflow.com/jobs/remote-developer-jobs…
Working Nomads: http://workingnomads.co/remote-development-jobs…
Remote . co - https://remote.co/remote-jobs/developer/
Remoters: http://remoters.net/jobs/software-development…
JS Remotely: http://jsremotely.com
Front-end remote: http://frontendremotejobs.com
IWantRemote: http://iwantremote.com
DailyRemote - https://dailyremote.com
Remotive: https://remotive.io/remote-jobs/software-dev
Outsourcely: http://outsourcely.com/remote-web-development-jobs…
Pangian: https://pangian.com/job-travel-remote/…
RemoteLeads: http://remoteleads.io
Remote Talent: http://remotetalent.co/jobs
JustRemote: https://justremote.co/remote-developer-jobs…
RemoteLeaf - https://remoteleaf.com

Sitepoint - https://sitepoint.com/jobs/

Startup Jobs

AngelList: http://angel.co/jobs

Product Hunt: http://producthunt.com/jobs

Startup Hire: http://startuphire.com

Startupers: http://startupers.com

YCombinator: http://news.ycombinator.com/jobs

Junior Dev Jobs

JrDevJobs: http://jrdevjobs.com

Stackoverflow Junior jobs: https://stackoverflow.com/jobs/junior-developer-jobs…

Women focused job boards!

Women Who Code: http://womenwhocode.com/jobs

Tech Ladies - https://hiretechladies.com

Freelance Jobs

Freelancer: http://freelancer.com/jobs

Upwork: http://upwork.com

FlexJobs: http://flexjobs.com/jobs

FreelancerMap: http://freelancermap.com

http://Gun.io: http://gun.io

Guru: http://guru.com/d/jobs

Misc

iOS: http://iosdevjobs.com

React: http://reactjobboard.com

Vue jobs: http://vuejobs.com

Ember: http://jobs.emberjs.com

Python Jobs - http://python.org/jobs

JavaScript job XYZ: http://javascriptjob.xyz

Javascript remotely: http://jsremotely.com

Muse: http://themuse.com/jobs

Tuts+: http://jobs.tutsplus.com

Krop: http://krop.com

PowerToFly: http://powertofly.com/jobs
Developers for Hire: http://developersforhire.com
Codepen job board: https://codepen.io/jobs
Joblist.app: http://joblist.app
Fullstack Job: http://fullstackjob.com
Authentic jobs: http://authenticjobs.com
Jobspresso: http://jobspresso.co
Jobs in Europe: http://landing.jobs
TripleByte: https://triplebyte.com