

A

EDWARD GUINNESS

ACE

THE PROGRAMMING  
INTERVIEW

---

160

QUESTIONS  
AND ANSWERS  
FOR SUCCESS

WILEY

# Hiring Programmers: The Inside Story

When I was a young boy, making new friends seemed easy. I had grown up with the surreal humor of Monty Python and my usual approach to a potential new friend would be something like *"I am a knight who says Ekki-Ekki-Ekki-PTANG!"* I thought it was hilarious. I made a few great friends (one is still a friend 30 years later), but I also had a lot of misses. Actually, mostly I had misses, nearly all the time. Sometimes my approach would generate open hostility. I couldn't understand it.

What my young self didn't realize was that enthusiasm for the absurd was not a universal constant. Not all kids my age had seen Monty Python, and even if they had, not everyone loved and appreciated it like I did. I was seeing the world through my own Python-shaped glasses, and I did not appreciate the diversity of other childhood experiences. Not everyone was on the same wavelength.

As naïve as this might seem, many hiring managers make the same basic mistake when interviewing. Perhaps they suppose that because they have a lot of hard-won experience in a certain area then *of course* everyone with experience in that area will see things the same way. Further, they might assume that their thought process will be similar. At the interview the hiring manager might abruptly open the conversation with the equivalent of my Python-inspired icebreaker:

*"Nice to meet you; now, could you please describe a situation where it would be inappropriate to normalize a set of database tables into Boyce-Codd normal form."*

It might be that you love Monty Python and, for you, meeting an interviewer who quotes from Python might seem like a dream come true. If that is the case, then I wish you all the best; just be careful how you respond when the interviewer asks you to *“Please, walk this way.”*

For the rest of us, establishing a level of communication that is easy and familiar will take some preparation and effort.

It might have occurred to you that I’m hinting at the concept of rapport. *Rapport* is an excellent word; it describes feelings of harmony, of sympathy, and of being on the same wavelength. I hesitate to use this word only because it has been somewhat hijacked in recent times and now comes loaded with connotations of insincerity, like the fake grin of a novice salesman.

But for an interview to be really effective, to have the ideal degree of communication, and to put yourself on common ground with the interviewer, you really do need to work on establishing a rapport.

One of the simplest and most effective ways to start building rapport is to try to see things from the interviewer’s perspective. If you understand the motivations of the interviewer, establishing a common ground and adapting your responses appropriately becomes much easier. You can quickly home in on what the interviewer is looking for, in both a positive and negative sense.

In this chapter, I will take a thorough look at the process of finding a programming job including:

- What motivates a hiring manager, and how to tailor your approach appropriately
- How to prepare a CV that will get you to an interview
- How to use job sites
- Understanding recruitment agencies; how they work and how to work effectively with them
- How to find jobs without a recruitment agency (it can be done!)

Let’s start by taking a look at some of the most common reasons why a company might want to hire a programmer.

## **Reasons They Recruit**

---

Without exception, a company hiring a programmer will have a reason for hiring. If you know what that reason is and understand the motivation for it, then you can optimize your approach accordingly.

## Planned expansion

A common scenario—one that will become increasingly common as the major world economies resume growing—is for companies to make medium- to long-term plans for expansion that require them to take on more programmers in accordance with their plans for business growth.

### *The interviewer's motivation and approach*

Because the role is part of longer term plans, the interviewer is unlikely to feel a great sense of urgency. Well-prepared interviewers will have a job profile, perhaps also a person profile, and be comparing candidates to these. With time on their side, they are less likely to compromise their pre-determined requirements, and although they might not realistically expect to perfectly match every aspect, they will probably be less open to considering candidates who deviate by any significant degree.

### *Your approach*

Your approach should be to highlight areas where your skill and experience matches well with the advertised job profile. That is the easy part.

What about skills that aren't a good fit? For example, suppose you apply for the role of a .NET programmer and during the interview it becomes apparent that the interviewer has an expectation that the ideal candidate will have experience of a certain component library, which happens to be one you've not used. In areas where your background is not such a good fit, you have three basic options.

### **Play it down**

Your first option is to downplay the perceived gap in skills—including the option of substituting other experience as being of equivalent value. If you decide to play it down, you might say:

*"It's been my experience that it never takes long to learn the basics of a new component library, since, as programmers, we face an endless supply of new components and frameworks both open source and from the major vendors."*

You might also comment that learning is part of the job:

*"One thing I really like about programming is the experience of learning new technologies and platforms. It's part of the ongoing attraction of the job."*

The biggest risk in taking this approach is that you might appear evasive, so be wary of overdoing it. After all, it is unlikely the interviewer chose the requirement at random, and if you push too hard at downplaying the requirement you might inadvertently maneuver yourself into an argument. Be sure to avoid that.

### **Take it on the chin**

Your second option is to take it on the chin, so to speak. Take ownership of the “development opportunity” and perhaps talk briefly about how you have acquired other, similar skills or experiences.

If you decide to take it on the chin, simply agree with the interviewer’s observation and at the same time show your enthusiasm for learning something new:

*“I don’t have experience of that particular technology but I would really enjoy learning it.”*

If the interviewer persists, you might feel it appropriate to ask how other developers in the team might learn new skills:

*“Could you describe how the developers in your team generally learn new skills?”*

Each example of learning given by the interviewer is an opportunity to show how, as a part of the team, you would benefit from the same approach and so acquire the necessary skill.

### **Understand the requirement**

Your third option is to explore the interviewer’s motivation, to gently probe the motives underpinning the requirement.

Exploring the underpinning motivation of the requirement gives you the best chance of looking good, but to take this approach you must have established a reasonable rapport; otherwise, you risk appearing argumentative. The basic idea is that you explore the requirement looking to show that you understand and can meaningfully address the underlying requirement despite lacking a specific skill or certain experience.

For instance, you might lack experience of a particular IoC container, let’s say Microsoft Unity. You might ask the reason for using that particular implementation of IoC:

*“I know there are a few good reasons for using an inversion of control pattern; could you describe how the Microsoft Unity framework helps with the kind of work you do here?”*

If the answer is to encourage loose-coupling of components, then this gives you an opening to talk about your understanding of dependency injection principles. If the answer is to encourage the writing of code that is structured in a way that better supports unit testing, then you can talk about your experience

of refactoring legacy code to add unit tests, or you could talk about your understanding of dependency injection without using an IoC container.

If the interviewer asks a pointed question, or simply makes a challenging statement regarding the relevance of your experience in a certain area, your response should be respectful but equally forthright. Often the case is that an interviewer with a blunt approach is looking for a similarly direct approach in the interviewee.

*Challenge: “I don’t see any experience of Microsoft Unity in your CV. We use that, so your experience doesn’t match our job spec.”*

*Response: “Is that a deal-breaker?”*

The principle at work here is *mirroring* the behavior of the interviewer (refer to the section on establishing rapport in Chapter 3 for a discussion of this powerful technique).

Whatever approach you take, keep in mind that you should not dwell on any particular mismatch. In particular, keep your comments brief and to the point. The more you talk about it, the more prominence it will have in the interviewer’s memory of the interview when they reflect afterward. There’s not much you can do if the interviewer appears to be stuck on an apparent mismatch—just keep your comments brief and resist the temptation to ramble on the topic.

## Specific projects

When a company spots an opportunity in the market, it might scramble to put together a development team focused on delivering a solution to capitalize on the opportunity. There could be pressure to be first to market, or perhaps the commercial opportunity is constrained to a limited window of time.

### *The interviewer’s motivation and approach*

The interviewer wants to know that you can work under some pressure, and that you are someone who finishes what you start. Sometimes that pressure to hire can relax the strictness with which the interviewer will match your experience against the details of the job specification, although of course you can’t assume that will be the case.

### *Your approach*

Be aware that although you need to demonstrate how your skills and experience are a good match for the job as advertised, the interviewer also probably has the needs of a specific project in mind. The company might have adopted the job specification to suit the project, but more often than not interviewers reuse a standard job description and look for the “extras” at the interview.

Your enthusiasm and ability to adapt might count for more than usual. Showing an ability to quickly grasp key aspects of the project gives you an advantage over applicants who stick to the published job specification.

Many consulting and service-oriented companies routinely put new software development teams together to respond to customer demand, so the chances are higher that this type of company will recruit with a specific project in mind. If you aren't sure of a company's motivation for hiring, there's absolutely no harm in asking directly:

*"Could I ask why you are recruiting? Is it for a specific project?"*

## Replacing someone

A third common reason for hiring a programmer is simply to replace someone who is leaving or who has left the company.

### *The interviewer's motivation and approach*

The interviewer will have similar motivation as in the planned expansion scenario. They will probably be under some time pressure, but not as much as if they were recruiting for a specific project.

An interesting aspect of this situation is that they will probably have experience of a previous person filling this role, for better or for worse, and will therefore have a list (written or not) of things they want to ensure the next person will and won't bring to the role. For example, perhaps the previous person was a stickler for detail, and this might, therefore, be high on the list of personal characteristics the interviewer wants to confirm in you.

### *Your approach*

Obviously, unless you're very lucky, you're unlikely to gather insight into what the interviewer liked and disliked about the person you hope to replace prior to the interview. What you can do at the interview is ask about unique challenges of the role, things for which you might need to be on guard, and so on:

*"Could you tell me a bit about the challenges of this role that might make it different from the usual programming job?"*

Experienced interviewers are unlikely to volunteer much information about the previous person, but they might give you clues along the lines of how certain attributes are important; for instance, "the ability to get along with the team." If you ask the right kind of question, you might get vital clues about the things you need to highlight with regard to your experience and ability:

*“Can I ask whether there have been issues about how the team has been working together that make this ability particularly important?”*

## Talking to Managers

---

It is a story often told. A capable and bright programmer impresses everyone and is promoted to team leader or manager. The newly promoted manager takes on the responsibility of hiring new programmers and uses his awesome interpersonal skills to hire more bright programmers.

The problem is that this is almost never how it happens. Many otherwise excellent programmers simply don't have awesome interpersonal skills, and sometimes these are the people who will be running the interview and interviewing you.

Is that a good thing, or is it bad? It depends. It could mean you suffer a terrible interview experience—in which case, count yourself lucky you won't be working for a poor communicator—or it could be a huge advantage. Think of it like this: In every human relationship, having things in common helps, and the more you have in common the easier breaking the ice and enjoying a conversation will be.

Now, as one programmer talking to another programmer (even if the manager isn't currently working as a programmer) what might you have in common? The answer is “lots.” Have you ever spent hours or days tracking down a difficult bug? Sure you have—and almost certainly the manager has, too. What do you like about a particular programming language? The manager might feel the same. Do you regularly visit a website for programmers? The manager might, too. Do you have a favorite XKCD cartoon? Do you visit <http://thedailywtf.com>? What is your favorite programming book? Have you ever used the word “nullable” in conversation with a non-techie? What annoys you about the IDE you use? There are lots of topics you can discuss together.

## Tech talk—don't hold back

It might become apparent during the interview that the interviewer is not as technical as you might have assumed. He might have a background in project management or product marketing, for example. How should you react? Don't make the mistake of thinking you need to “dumb it down” for the interview. The problem with “dumbing it down” is that you put yourself at a disadvantage when the interviewer compares your responses to another candidate. He might not understand everything you say, but his impression of your response will be colored by the language you use, and if you talk purely in metaphors (for example) then you risk giving the impression that you don't actually know

the subject as well as the other candidate who talks about specific language or framework features using their proper names.

If a non-technical interviewer asks you a technical question then he expects a technical answer. The case might be that he has a “cheat sheet” of his own, a list of questions and answers, for example.

Don’t hold back—answer the question fully and as you would if talking to a technical interviewer. If the non-technical interviewer wants you to explain something in non-technical terms, then you could use a metaphor (see the next section “Using metaphors”). As a rule, keep your answers grounded in reality using real names and proper terms for things.

## Using metaphors

Sometimes a non-technical interviewer wants to assess how well you can explain a technical subject to someone who is non-technical. In this case, a metaphor is your best bet.

For example, suppose you are asked to explain the concept of an IP address in non-technical terms. Here is the definition from Wikipedia:

---

**An Internet Protocol address (IP address) is a numerical label assigned to each device (e.g., computer, printer) participating in a computer network that uses the Internet Protocol for communication.**

---

The first metaphor that might occur to you is that an IP address is much like a mailing address. A mailing address is used to deliver mail, and an IP address is used to deliver packets of information, so the metaphor seems to work. On the other hand, mailing addresses vary widely in format and local conventions, whereas an IP address conforms to a strict set of rules that are consistently and globally applied. Perhaps a better metaphor would be latitude and longitude?

While metaphors might conveniently help a non-technical person relate to an aspect of a technical subject, they are almost by definition an imperfect representation. They have limits, and you should never cling to an imperfect metaphor after reaching those limits. For example, if you use latitude and longitude as a metaphor for an IP address, presumably you don’t mean to imply that IP addresses are assigned purely due to the physical location of the computer or device.

## Preparing Your CV

---

A good CV (also known as a resumé) gets you past the filters of recruitment agencies and human resource (HR) departments. A good CV will mean your application is not rejected out of hand, and it could give you an opportunity to

talk to someone about the job role. A CV by itself is never going to win you the job. In other words, no hiring manager ever decides to hire someone purely on the strength of her CV alone.

## **Include relevant keywords but keep them in context**

Knowing that, in most cases, non-technical agents will filter your CV has one very important implication—you need to be sure to include keywords that are relevant to the job and in particular to the job advertisement. The technical hiring manager might well understand that working on the ECMAScript specification means you have excellent knowledge of JavaScript, but the typical recruiter will not see the connection unless you spell it out and include the keyword “JavaScript” in your CV. Of course, you also want to avoid the appearance of insincerely stuffing your CV full of keywords, so ensure that any list of keywords is presented in proper context.

## **Write as well as you can**

Poor writing puts you at a severe disadvantage. What you write must be clear and concise. You must proofread (or have a friend proofread) and ruthlessly rewrite or delete anything that isn’t clear or relevant.

When I am unsure how to write something, I find that pretending to tell it to a friend is helpful. If you have no friends, tell it to your hand, and then write down the words you used. It doesn’t matter what they are at first because the next step is to revise those words into shape. Showing a bit of personality is good, but don’t ramble. Also keep in mind that what might seem funny to you while writing your CV might not necessarily look so clever to everyone who reads it. A much better strategy is to let your personality show at the interview after you’ve established a rapport. If in doubt, always err on the side of plain and simple writing.

Some managers don’t care so much about spelling and grammar, but others (like me) care a great deal. I think my distaste stems from years of reviewing poor code that is often full of spelling errors. Using the spelling checker in your word processor or browser doesn’t take a lot of extra effort. Don’t ignore the red squiggly lines!

Most people find that writing well takes a bit of effort and discipline, so be realistic and allow yourself plenty of time to write and revise.

## **Justify your claims of experience**

If you claim to have experience in an area, or if you claim knowledge of a particular technology, the hiring manager needs to understand how you acquired it. If your CV doesn’t show how you got “five years of experience of ASP.NET”

then you risk appearing insincere. If you claim to have the experience then the hiring manager will want to see where you got it. If I don't see it, I won't necessarily assume you're lying, but you will be at a disadvantage when I have another CV that clearly shows where the experience was obtained.

If you claim to be good at anything, it should be reflected in the details of your employment or education history. You need to be explicit. Also be aware that job titles today are almost meaningless. Being an "analyst" could mean just about anything and doesn't necessarily imply analysis skill of any kind. A "web developer" could mean someone with programming skills or someone with Photoshop skills. A "programmer" should mean someone who produces code, but I've seen it used for jobs where the work amounted to configuring systems via drop-down lists. You need to describe the technology, and how you used it.

## **Ignore anyone who tells you "strictly one or two pages"**

Ignore all the advice about keeping a CV short. That's for people with short attention spans and for recruiters who in many cases don't care for details. If you have lots of experience it should be reflected in your CV. As a hiring manager I want to see it, and more importantly you should be proud of it, not ashamed.

This is not an excuse to ramble—the advice about being clear and concise still applies, and you still need to make effective use of keywords and summaries.

## **Emphasize skills that match the job advertisement**

There's nothing wrong with emphasizing certain skills to match a job advertisement. As a candidate, it took me a long time to appreciate that emphasizing different skills depending on the job in question is perfectly fine. I used to think it was deceptive, but I don't think that any more. See it from the hiring manager's point of view—if a CV clearly calls out skills that match the job description then that CV will appeal more than some other CV that contains the same skills but buries them among other, less-relevant skills.

In other words, highlighting relevant skills helps everyone.

If you have experience in a variety of areas, consider writing more than one CV, each one emphasizing different areas. You might write one that targets database-centric roles, and another that emphasizes your business analysis skills. Having these prepared and ready to send means you can respond quickly to job advertisements without the need to revise your CV before applying.

## **Don't leave unexplained gaps between jobs**

Don't leave gaps in your CV, especially big ones. It risks making you appear shifty or evasive. If you were volunteering in Africa, that's great. If you took time off to retrain, or to pursue a start-up idea, or to raise a child, that's all good. The only thing you have to prove is that you can do the job. Yes, in some cases,

a gap in your experience might mean you have to accept a lesser salary, but a savvy employer will see that as an opportunity rather than a problem, and savvy employers also know that life experience counts for something.

## “Reading, music, and cinema”

My estimate, based on reading many hundreds of CVs, is that 80 percent of CVs contain a “personal interests” section, and that 80 percent of these sections contain the exact same list of things; reading, music, and cinema.

You really don’t need this section in a CV, especially if it contains the same interests listed by everyone else. If you do happen to be a Tiddlywinks world champion, maybe that is something to show off, but in general, rather than trying to show how well-rounded and interesting you are in this odd little corner of your CV, try to let that show through in the rest of your CV, through your experience, and at the in-person interview.

On the other hand, if you have (for instance) used your programming skills to support a non-profit organization, perhaps a charity, then you should mention it. I know a person who worked with inner-city kids at one point, and as a hiring manager I would definitely be interested to read that. Anything you’ve done that demonstrates enthusiasm, altruism, or any other noble quality is worth including. Just avoid being boring because you felt compelled to include this section—better to leave it out.

## Use a logical layout

In general, the layout of your CV should be easy to scan and arranged logically. A key goal of any CV is to make it easy for someone to check boxes off when comparing your experience and skills against a job specification. I recommend starting your CV with a well-written summary that includes an overview of your key skills and significant experiences. The next section should be your job history shown in reverse chronological order, followed by a section for projects including any voluntary or unpaid work experience, and the final section should be education and training.

Every page of your CV should contain your contact information—your name, e-mail, and phone number, for example, within the header and footer areas. Contact information does not need to be a dominant feature but you do need to ensure that employers or recruiters have no trouble finding your contact details when they want to contact you.

## Graduate CVs

With a relatively small amount of work experience, you will need to show off your abilities by listing projects, at or out of school, and the role you played in each. If you led a project, played a key role, solved a difficult problem, or designed

a complicated component, then be sure to highlight that as an accomplishment. Testimony from someone who assessed your work or benefited from it in some way adds authenticity.

## CVs containing more experience

Experience is wonderful, and there is no substitute. However, a lot of experience can be interpreted in a number of ways and not all the ways are flattering. Twenty years with a single employer might give the impression, right or wrong, that you have never been challenged professionally. No doubt within those twenty years you overcame many tough challenges, so you should ensure that these are made visible in your experience. You should include subheadings for:

- Major projects where you had significant responsibility
- Different roles or major responsibilities
- Organizational changes that required you to adapt

## CVs of those returning to work

If you've taken time off to raise children, travel, or study, then you've absolutely nothing to hide or to cover up. Be honest in your CV about why you have not been working. A single sentence is probably enough. Also, be honest with yourself about areas in which you might be somewhat out of date.

If you've been out of work due to redundancy—perhaps a down-sizing due to poor economic conditions—or if you were outright fired then, please, be honest in your CV. You don't need to use the word “fired” (if you were), but you don't want to hide your circumstances or pretend it was according to some plan. Hiring managers are hyper-vigilant for this kind of deception, and (trust me on this point) if a manager feels you are being deceptive in any way, you will have zero chance of landing the job.

Redundancies happen, and now and then people are fired. A simple and honest summary of the situation is by far the best strategy.

If you were fired, you may find some doors are closed. I can't pretend otherwise. The most practical advice I can give is to omit your “firing” from your CV, and be prepared with a good answer when asked why you left that job. A good answer is something like:

*“I was fired following a disagreement with my team leader. We fell out after I took an initiative she disagreed with. It was difficult, but I feel I learned a lot from the experience.”*

You may have strong feelings about a firing and that is, of course, quite natural. What you mustn't do, even if the interviewer encourages it, is to complain about your past employer. Let the facts speak for themselves, and allow the interviewer to come to their own conclusion.

Think of it this way: If you are composed, dignified, and reasonable at an interview, the interviewer will be more inclined to believe your side of the story. If you are emotional and allow yourself to vent, the interviewer may be more inclined to sympathize with your past employer.

It is difficult, but be sure to consider how you will answer this question, before the time comes (as it surely will).

## Avoiding common CV blunders

It is a sad truth that many hours of good work can be undone by a moment of carelessness. The first impression a hiring manager will have of you is based on your CV. Not only that, but your CV will sit alongside a stack of other CVs. Most hiring managers sift this stack into a smaller “short list” stack, and to do, that they must somehow discriminate. All other things being equal, the small mistakes in your CV could result in rejection. Let’s go over some of these “small” things so you are sure to avoid them.

### *Poor spelling or grammar*

Poor spelling or grammar can ruin an otherwise good CV. It happens so often that it is almost a cliché. Be sure to proofread your CV, or better, have an eagle-eyed friend check it for you. If English isn’t your first language then you really should have a native English speaker review your CV, even if it means paying for it.

### *Vague or nonspecific accomplishments*

When describing accomplishments, you need to keep two things in mind: Be specific, and describe the accomplishment in terms that most people can understand. For example, if you discovered and fixed a performance issue in a framework component, someone unfamiliar with that component (but otherwise technically informed) will find it more impressive if you describe the impact of your performance tweak rather than the mechanics of the tweak itself. Describing a “200 percent performance improvement” is much better than describing how you implemented a custom sort method. You will have plenty of opportunity to show off your technical skills in more detail at the interview.

### *Unclear or cluttered Layout*

Of all the mistakes, fixing a cluttered layout is the easiest to detect and to fix. Simply hold up the page—without trying to read it—and let your eyes focus on whatever leaps out of the page. If nothing leaps out, it’s too cluttered—or too empty! If the wrong things leap out then the CV isn’t arranged well enough.

Don't be afraid to add whitespace around things you want to highlight, and whatever you do, don't use a tiny font hoping to cram in more information.

### *Unprofessional e-mail address*

No matter how hilarious your friends think it is, "jojo\_2hot4u@hotmail.com" is never going to be appropriate on a CV. Think how this address looks to someone who's never met you and is trying to form an opinion based purely on the presentation in your CV.

Domains are cheap to register and easy to maintain, and unless you have a very common name then *yourfirstname@yourlastname.com* or a variation like *@yourname.info* will probably be available.

If you don't want to spend any money then at least consider signing up for a more "professional" e-mail from one of the many free providers such as Google or Yahoo!.

## Using Job Sites

---

All online job sites function similarly. You can upload your CV and thereby make it searchable by employers (or more likely, by recruiters), browse and search for jobs, subscribe to receive e-mailed notifications of new jobs, and apply for jobs.

For the job seeker, these sites are always 100 percent free to use. They all have a huge following of recruiters and employers. From a purely statistical point of view, you have a better chance of making contact with a potential employer than if you go it alone.

Some of the search tools on these sites make them truly useful. For example, you can limit your search to a very specific location, salary, or job title.

The sheer volume of users on job sites means that your CV is less likely to be noticed. You join a very large crowd of job seekers, and it is harder to differentiate your CV from the many thousands of others. It can be disheartening to send your CV and hear nothing back.

Unfortunately, it doesn't take much time reading through job advertisements to realize that almost all the jobs posted on these sites are posted by recruiting agencies. This means you're almost never going to be dealing directly with an employer. This can sometimes (though rarely) work in your favor should the agency take a special interest in your application, but, for the most part, these agencies represent another layer of obstacles between you and the interviewer.

Also, it doesn't take too much imagination to see how these job boards might potentially be abused by unscrupulous agencies who post fake job adverts in order to gather CVs from applicants, CVs which are then redacted and forwarded to prospective clients as the agents tout for business. This may not be common but it would be naïve to think it never happens.

Be on your guard, and don't invest too much mental or emotional energy in the "perfect" job after submitting your application. If you don't hear back soon after submitting an application for which you are obviously suited, the job may never have existed in the first place.

## Comparison of major job boards

All job sites operate similarly and have the same pros and cons, so how you choose which site to use is a question of personal preference. It might be helpful to see which sites post the most jobs for programmers, in which case refer to the numbers in Tables 1-1 and 1-2. The search query used to obtain these numbers was "programmer OR software developer" and the search was limited to "posted within the past seven days." The query was run in June 2012.

**Table 1-1:** Job Listings on Major U.S. Sites

SITE	NUMBER OF JOBS LISTED
CareerBuilder.com	1,950
Monster.com	1,000+
Jobserve.com	718
ComputerJobs.com	685
Dice.com	152

**Table 1-2:** Job Listings on Major UK Sites

SITE	NUMBER OF JOBS LISTED
CWJobs.co.uk	3,018
Reed.co.uk	2,439
Jobserve.co.uk	930
Monster.co.uk	531

## Recruitment Agencies

All recruitment agencies work on behalf of employers to find and match job seekers to vacancies. While all agents will initially appear very interested in your career prospects, the extent to which they are committed to finding you a job is limited by how likely they think it is that they can "sell" you to an employer. If they don't have a live vacancy that matches your profile, you might never hear from them after the initial contact.

Almost all agencies work on a contingency basis, which means they charge their clients (the employers) a percentage of the salary they will pay to the person they hire following an introduction. If the agency doesn't place someone with that employer then it gets nothing. The fees of recruiting agencies vary a lot, but are usually somewhere between 10 and 30 percent of the new hire's first-year salary. This might seem like a lot of money for very little work, but the uncertain nature of this work—and the fact that many employers are willing to pay these fees—appears sufficient justification for the recruiters to continue charging what they charge.

Job seekers are not normally asked to pay any fees to recruitment agencies so you should treat any such request with great suspicion.

In exchange for the promise of this contingent fee, the agency might advertise the job vacancy on a number of online job sites and will probably also search its database of candidates looking for a match between candidate skills and the skills listed in the job description provided by the employer. Most recruiters will also put the word out via their network of contacts, hoping to attract suitable candidates they can then introduce to their client.

Interestingly, almost all recruiting agents involved with sourcing and placing software developers have no significant first-hand (and sometimes not even second-hand) experience with software development. When working with employers and programmers, their role is purely sales and (social) networking. Inevitably some of the more experienced agents will pick up a basic vocabulary of technical and programming terms that enable them to communicate a bit less awkwardly, but as a programmer you might occasionally find yourself explaining the most elementary programming terms to a recruitment agent. What you need to remember is that although agents might lack technical comprehension, whatever you say to them will (hopefully) be replayed to a potential employer. To minimize the chance of Chinese whispers, avoiding the temptation to relate your experience in great detail is usually the best strategy. Save that information for when you talk directly with the employer. Good agents will ensure that the most relevant aspects of your experience are relayed to the potential employer. They will also be aware of their own limited technical knowledge when communicating with employers, and, for the most part, they will (and should) refer technical questions to you. If you ever find that an agent has “made stuff up” about your experience (and incredibly some do) then you should distance yourself from that agent as soon as possible. He or she won't be helping you find the right employer and might even damage your reputation.

Some agencies want to interview you prior to making an introduction to any employer. This is an interesting and sometimes amusing affair, where the agent, who knows next to nothing about the technologies in your CV, asks you questions about your experience with those technologies. Pragmatic agents will recognize their own limitations and restrict themselves to asking generic interview questions such as those related to your motivation, your aspirations,

your attendance, and sickness-absence record, and so on. One important thing to remember is that apart from a routine confirmation of your background and experience, the agent also wants to gain a clear impression of your “personality,” which he or she can then relate to the potential employer. This is one way agencies and agents “add value,” though the actual value added from this subjective assessment is perhaps questionable.

## Working effectively with a recruiter

Remember that recruiters are paid by employers, and so their motivation to help you can vary depending on the prospects currently on their books. Investing too much emotional energy with any individual agent or agency, regardless of how exciting the job on offer might be, is generally not advisable. Certainly never be tempted to rely exclusively on a single recruiting agency, regardless of how effective it might appear.

Also remember that agents act as middlemen, and that whatever you say might be replayed to a potential employer. Although you need to convince the agent that your credentials and experience are genuine and relevant, being somewhat guarded in disclosing anything that isn’t clearly advantageous to your application also makes sense. You can’t avoid obvious questions, such as gaps in employment, but avoid volunteering unflattering or “complicated” information that might be misrepresented to the employer.

When waiting for news from an agent, don’t waste your time and energy fretting if she doesn’t return your calls. Be assured that an agent will waste no time in contacting you when she has news—she doesn’t get paid unless she places you. Remember, too, the agent is at the mercy of the employer and is therefore just as likely as you to be anxious when waiting for news. Keeping in touch with the agency doesn’t hurt—perhaps do a weekly “check in” call to ensure the agency has you in mind and is aware of your continued enthusiasm—but avoid pestering agents. It doesn’t help.

## Searching for Jobs Yourself

---

An alternative to relying on recruitment agencies is to find and contact employers yourself. This task isn’t as hard as you might think. After all, most employers want to spread the news of a job vacancy as widely as they can, and many have an area on their website dedicated to advertising current vacancies. Very large employers might have an arrangement with a number of preferred supplier agencies or have a dedicated team in their Human Resources department that handles all incoming employment queries, but—and I cannot stress this enough—most hiring managers still welcome direct approaches from suitable candidates, provided, of course, that the contact is respectful and personalized.

If you make contact with a hiring manager who then refers you to the HR department, don't be disheartened. The manager is very probably obliged due to corporate rules of "efficiency and uniformity" to send you through to HR. Unless you feel that the contact was unwelcome then nothing prevents you from remaining in touch with the hiring manager as your application progresses through the necessary HR channels. Remember that the corporate "rules" don't apply to you unless and until you're an employee. Always remain respectful and forthright in all your communication, and don't be put off by the bureaucracy.

A word of caution—don't be tempted to send a generic letter or e-mail. To the potential employer this approach puts you in the same category as the bothersome agencies that send out unsolicited and unwelcome junk mail about "exciting opportunities" or "hot candidates." Always personalize your message as much as you can, including addressing the letter or e-mail to the hiring manager rather than "To whom it may concern," and mention specific and genuine reasons why you are interested in joining the company. If you are stuck for ideas then ask yourself these questions: Does the company produce a product you've used? Is it working in an area with potential that excites you? Have you heard something interesting about the company from friends or in the news?

One of the best ways to find a potential employer is through your own personal network of friends and contacts. Perhaps you're lucky and have naturally built up a large social network over the years, or perhaps (more likely) as a programmer you don't have a particularly large social network. Many of us know intellectually that having a large network can be a tremendous asset, especially when looking for a new job, but how many of us (and I sheepishly include myself in this number) have failed to invest time and energy into establishing and maintaining this kind of network?

If you don't have a significant network of contacts because you are—and let's call a spade a spade—an introvert, then don't despair. This next section is especially for you. I only wish I had followed this advice myself from an earlier age.

## Networking for introverts

The problem with most of the advice you may have read about networking is that it was written *by extroverts for extroverts*. If, like many programmers, you are not particularly extroverted by nature, most of this advice is wasted on you. What's even worse is you might feel guilty or stressed about not being able to follow what is presented as "simple and easy" advice for networking.

Don't feel guilty. Instead, play to your strengths as an introvert.

Introverts are often quite thoughtful, preferring to spend time thinking about a topic rather than talking about it. This trait can be a major advantage when "preparing to network." If you want to make contact with someone, set

yourself a goal of making a list of thoughtful questions to ask that person when the opportunity arises. Most people love talking about themselves and sharing their opinions. This is not a cynical view; it's just basic human psychology—the sharing of personal experience is fundamental to any human relationship. Thoughtful, open-ended questions asked at the right time break the ice and set the relationship off to an excellent start.

Introverts are often more comfortable talking to individuals rather than to groups of people. It also happens that communicating one-on-one is much more effective in terms of exchanging ideas and building relationships. Look for situations where you can engage a new contact one-on-one, even if that interaction happens in a public place. Perhaps ironically, the open and public space of Twitter is a good environment for an introvert to make contact. The 140-character limit works well for those who prefer thoughtful and concise exchanges over wordy and extended conversations, and after you're followed by a Twitter user you can send him direct (and private) messages. You'll learn more about using Twitter and other social networking tools later in this chapter.

Many introverts find the job of building a network somewhat daunting. It needn't be. The trick is to think of it as a game with a well-defined goal—to make new contacts and build meaningful relationships with them. The hardest part for introverts is that the “rules of engagement” are vague. “*Exactly how,*” an introvert might wonder, “*do I go about building a network?*” If you're introverted, these vagaries are annoying and off-putting—“*What on earth should I do next?*” If you're extroverted these vagaries are exciting—“*I can do whatever I wish!*” (Though if you're extroverted you've probably skipped this section or are reading it out of curiosity.) For introverts, here's a cheat sheet for how to build a network:

- Make a list of potential contacts.
- Sort the list into order, from “most desirable and probable” to “least desirable and improbable.”
- Starting at the top, think about what kind of questions the contact might find interesting, and write down a short list of questions.
- In a suitable environment, ask those questions of up to five potential contacts (more than five might be difficult to do, depending on your personal capacity and energy).
- Engage with those who respond.
- Keep adding to your list so you always have five potential new contacts.
- Repeat!

Finally, if you find that networking drains your energy, be sure to protect your energy reserves by pacing yourself. Take time out to reflect and process

information gathered from social exchanges, and in this way prepare yourself for the next opportunity to build and maintain your network. It's a marathon, not a sprint.

## Finding potential employers

Potential employers are out there. Unfortunately, many of them are unthinkingly patronizing recruitment agencies, or perhaps (more fairly) they exclusively use agencies because they don't know of any better way.

Consider the problem of finding candidates from the hiring manager's perspective—wouldn't it be nice if a suitable candidate contacted the company directly? Wouldn't it save the company a lot of money and time? Too right, it would! With that in mind, here are some of the channels you can use to unearth these potential employers:

- Your personal network of contacts, including Twitter, Facebook, and LinkedIn
- Regional business directories
- Chambers of commerce job fairs
- Job sites, limiting your search to jobs that have been "posted by employer"
- Web search for "careers" + "*employer name*"
- Professional journals

Searching for a job is no time to be bashful. Add a note to your online profiles that you're available; you never know who might see it.

## Approaching potential employers

Approaching a potential employer "cold" is probably the hardest aspect for many job seekers, and I won't pretend otherwise. If you can, having someone (other than a fee-seeking agency) introduce you is much better. If you use LinkedIn, check to see whether anyone in your network can help, and of course, ask around and make use of your other social networks. You've probably heard of the idea that everyone is on average six steps away, by way of introduction, from any other person.

If you don't have a connection then you'll need to contact the hiring manager without an introduction. Don't panic; this isn't necessarily a problem. The first thing to do is to find out as much as you can about the company, the department, and the people who influence and make hiring decisions. I say "as much as you can," but I don't mean you should spend an unlimited amount of time researching a company. Set yourself a reasonable limit, perhaps a few hours

at most, by which time you will have all the useful information you need plus some insight into how the company is structured and who makes the decisions.

Obviously, you need to look at the company website where, quite often, you will find most of the information you need. At the very least you will find a contact telephone number. Call it. Don't pretend to be anything other than a job seeker, and ask the person who answers the phone whether he or she can help you. You only need a few bits of information, and if you are up-front and honest most people are happy to help. It doesn't hurt that many receptionists have been in your position (job seeking) and therefore have some sympathy for job seekers. Be warned, though: The first hint of deception or evasiveness will cut your conversation short. Having had the benefit of sitting near receptionists, I've heard this happen over and over. Companies receive endless cold calls from sales people and recruiting agencies, and most receptionists have developed finely tuned senses to detect these types of calls. A naïve sales person will call asking to speak to the software development manager about "a business opportunity" and, inevitably, he achieves nothing but having his details taken on a Post-It note that is soon discarded. Don't let that happen to you!

I do not recommend that your very first contact should be via e-mail. Think of what happens to the unsolicited e-mails that land in your own inbox. You need to call up and talk to someone. The aim of the first call is very simple: You want to get the name and contact details of the person who makes hiring decisions in whichever department hires programmers. Ideally you will also find out a bit about the structure of the company and whether or not it is hiring, but at minimum you must obtain the name and e-mail address of the hiring manager. If the first-contact conversation goes well, you might ask the person you talk to if he or she will pass on a message to the hiring manager that you're a software developer, you're available, and that you're hoping to have a chat. Don't fret about the possibility of the message being lost or ignored. That might happen, but more than likely this message will serve as a kind of introduction in lieu of one from a common connection in your social network.

## Being persistent

One thing that is drilled into trainee sales people is that persistence can pay off. This is a double-edged sword, for although it is true that persistence can pay off it can also taint your reputation and result in you being labeled as a pest. A fine line exists between persistence and pestering but, at its most basic, persistence is the idea that you should not give up at the first hurdle.

This can mean that you need to make more than one phone call to get the information you desire. It can mean that you need to send follow-up e-mails rather than assuming the worst. It might mean that your e-mail was accidentally

filtered into a “junk mail” folder and you need to send a personally addressed letter, which is something I recommend in any case. Ultimately, you must be the judge of when your efforts have crossed the line, but remember that if you have consistently been polite, respectful, and direct in all your communication then there is no good reason why anyone would consider you a pest.

## Timing

Many large companies are constantly in the news for various reasons. If you have an ambition to join one of these companies, you should keep a lookout for any news that suggests expansion plans are afoot. For instance, many publicly traded service companies announce whenever they win major new contracts or business. Submitting (or resubmitting) your application after learning of news like this might provoke a favorable response, when otherwise the answer would have been “no current vacancies.”

Also, most companies make plans for expansion that coincide with their financial year. If you find out when this financial year starts—and this information is readily available for all publicly listed companies—then you might stand an improved chance if you submit an application just as the hiring process kicks off.

## Emerging Alternatives

---

If you’ve spent much time online then almost certainly you have heard of or are active on one or more social networking sites. Here’s a quick overview of some of the more popular sites, and how suited they are (or can be) to finding your next job as a programmer.

### Using Twitter

Twitter is a social networking service where users can post and exchange messages. These messages are known as “tweets” and they are limited to 140 characters. Twitter claims to have more than 140 million active users who collectively post 340 million tweets daily.

When you’re looking around for potential employers, don’t forget that the reason most people join and use Twitter is to exchange funny tweets, interesting thoughts, and useful information. A steady stream of similar-sounding tweets won’t gain you many followers, and neither will tweets that are impersonal and stuffy. To be interesting, you also need to be a little bit vulnerable, sharing your ideas even if they won’t necessarily be universally popular. This is not to suggest you should continually air your unpopular opinions (should you have

any), but rather that you should not be afraid to—always respectfully—say what you think. A monotonous repetition of dull links to your website won't gain you many followers.

If you are new to Twitter, spend the first few days reading the tweets of others, noting what kind of tweets are popular and retweeted, and also noting what kind of tweets are less popular or ignored.

You can pay to gain followers, but I don't advise you do this. The followers you gain won't be at all interested in you, and they won't be interesting to follow. You can also pay for software that automates the “following” and “unfollowing” process in order to gain a large number of followers. You might find this helpful, perhaps as a minor ego boost, but again I don't recommend you do this. You can distinguish these robotic accounts from real users because they will inexplicably follow you one day, and if you don't reciprocate they will unfollow you in short order. Real relationships in Twitter, just like in life, are not something you want a robot attending to, at least not if you want real, potentially helpful connections with real, potentially helpful people.

In summary,

#### **Do:**

- Engage with others, including retweeting tweets you find interesting or relevant.
- Update your profile to include mention of your availability.
- Include a link in your profile to your CV (or better, your website where your CV can be found along with other interesting material you've written or compiled).

#### **Don't:**

- Post unprofessional or offensive tweets (this includes not retweeting the potentially offensive tweets of others).
- Limit your tweets to just advertising yourself and your job search. Tweet about what you find interesting or amusing.
- Follow obvious spammers or “follow back” robots. Yes, this increases your number of followers but at the same time it taints your account by association.

## **Using Facebook**

Facebook is probably the best known and most widely used of all social networks. You might have seen the film *The Social Network*, and the chances are you have a Facebook account no matter where in the world you happen to live.

The popularity and the ubiquity of Facebook is a double-edged sword; a potential employer might have a presence on Facebook, which can give you some insight into their company and culture. On the other hand, consider the sobering thought that every potentially embarrassing thing you've ever posted on Facebook is at risk of being discovered by that employer. You should also be aware that the U.S. Federal Trade Commission recently came to an agreement with Facebook over their accusation that Facebook "deceived consumers by telling them they could keep their information on Facebook private, and then repeatedly allowed it to be shared and made public." You could assume that this agreement between Facebook and the FTC will lead to better privacy controls at Facebook, or you could take a more cynical view that Facebook is unlikely to care for your privacy unless it is forced to by external regulation.

There is another privacy aspect to Facebook, although perhaps it has been exaggerated in the popular media. It has been reported that employers and academic institutions have begun asking for access to the accounts of Facebook users, or to become Facebook "friends" with potential employees or students. Whether this is right or wrong or even legal is largely irrelevant; the implications for many job seekers faced with this kind of request is unpleasant. At the very least, this gives many people a reason to hesitate before freely posting social updates.

Ultimately, Facebook is great for purely social networking, but not so great for the job seeker.

## Using LinkedIn

If you had to choose exactly one networking site for the purpose of finding a programming job, you are well-advised to choose LinkedIn. More than any other networking site, LinkedIn clearly targets professionals; that is, those in paid employment and employers. LinkedIn's mission statement makes it clear beyond any doubt that this is the site for connecting with other professionals.

---

**The mission of LinkedIn is to connect the world's professionals to enable them to be more productive and successful.**

---

Using LinkedIn is mostly self-evident. You connect with people you know, join groups that interest you, keep your employment history up to date, and post occasional updates. You can also interact with other LinkedIn users by sending "InMail," which is handy if you don't have any other contact details for that person.

LinkedIn also provides a convenient way to showcase recommendations from your colleagues and business associates. You can display (or hide) these recommendations in your profile.

For a monthly fee LinkedIn also offers a number of extra features for job seekers, including:

- Five “InMails,” which allow you to contact anyone on LinkedIn (normally you are limited to contacting just your immediate network).
- Information about who has viewed your profile (normally you can see just limited information).
- Your profile displays a “job seeker” badge, though this is optional.
- When you apply for a job via LinkedIn, your profile appears as a “featured applicant” at the top of the list as viewed by the job poster.
- Recruiters (in fact, anyone on LinkedIn) can send you messages via “OpenLink,” which appears as an icon displayed in your profile. This is an optional feature.

## Writing a blog

Writing a blog is incredibly effective as a way to connect you with potential employers. If you write well, and have interesting or amusing things to say, you can reach a very large audience who will return to your blog or sign up for your RSS feed. Over time, you can develop a group of loyal readers, and the value of that loyalty for the job seeker is huge.

That’s the good news.

The reality is that many blogs are started with the best of intentions, and then fall silent after an initial burst of enthusiasm. Why is this so? It’s because writing takes time, accumulating readers takes time, and building up a significant following can take months or even years.

If you regularly post on Twitter or any social network then blogging might be the option for you, in the sense that every interesting thing you might post to a social network can instead be posted to your blog. If not, then you have to ask yourself whether you have the capacity to sustain a blog. The content has to come from somewhere, and if your heart isn’t in it, then it will probably show in your writing—and consequently you won’t gain many readers.

Getting started blogging is easy, and many free services are available to get started with. I don’t recommend these. Hosting your blog on your own domain and with a paid hosting provider lends credibility to your blog but perhaps more important, gives you ultimate control over your blog and your content. In contrast, anything you post on a “free” site is subject to terms that detract from a professional image. For example, some free sites adorn your blog with unrelated advertising. Similarly, if you post your content on one of the well-known social networks, you probably don’t have much control over that content, and depending on the terms and conditions you agreed to when signing up (you

do remember those, right?) then the possibility exists that you might not even own the content you've created.

If you decide to set up a blog, here's what I recommend:

- Choose a topic for your blog that is relevant to your career interest. Programmers are lucky—we have an enormous number of very interesting topics to talk and write about. Resist the temptation to blog about random unrelated things—a focused blog is more likely to attract loyal readers.
- Pay for hosting and buy a suitable domain name.
- Set yourself a reasonable schedule for writing, one that is regular and to which you can realistically commit over the long term.
- Try to build up a buffer of blog posts for those inevitable times when for whatever reason (for example, holiday or sickness), you can't write new material.
- Use blogging software. If you don't know which blogging software to use then I recommend WordPress as a good starting point.
- Enable comments on your blog posts. Yes, this attracts spammers, but the benefit of engaging with your readers far outweighs the hassle of filtering spam comments. People love being heard, so don't ignore them when they leave comments.

## Careers at Stack Overflow

At some point, most programmers find themselves searching the Web for answers to a difficult or obscure problem. Many of these programmers find the answer on Stack Overflow, a relatively new site that has become a mecca for programmers and a truly valuable resource. Most programmers reading this will know of Stack Overflow and will have visited it many times.

What isn't quite as well-known is that the good people at Stack Overflow have branched out from their initial programming Q&A site and now also run a site for employers and job seekers at <http://careers.stackoverflow.com/>.

The operation of this site is similar to most job sites: You can submit your details to allow employers and recruiters to find your profile, and employers and recruiters can search for profiles using keyword and location. The unique aspect of <http://careers.stackoverflow.com> lies in its connection with Stack Overflow. If you have contributed answers to Stack Overflow then your "top answers" display in your profile along with how you rate in various subjects. Your Stack Overflow reputation will also be on display.

Whether your involvement with Stack Overflow is seen as a significantly positive aspect of your application depends on the attitude of the potential employer. Tech-savvy employers might be impressed, and some might even browse your contributions (including your questions!) to gain insight into your

style of communication and to see how well you can express yourself. On the other hand, the possibility exists that some employers won't see your involvement with Stack Overflow in quite the same light. For example, they might understandably have concerns about the time you might spend on this site versus actually working for them. If you have a particularly high reputation, this might be an issue you must prepare to address during a phone or in-person interview. If your involvement with Stack Overflow was during a period of unemployment, the time you've spent on Stack Overflow probably won't be an issue, but otherwise you should consider how you can credibly respond to the question when it is asked.

## Skills matter: “Find Your Ninja”

An interesting alternative to the usual approach to matching employers with programmers is the Find Your Ninja project run by the training company Skills Matter, London.

The basic idea is that employers who want to hire a programmer pay a fee to attend, and pitch their company and project to an audience of programmers. Programmers attend the event for free.

This event sounds similar to a traditional job fair, but an important difference exists. Because the audience consists mostly of experienced programmers who have heard about the event while attending training at Skills Matter, the presentations (that is, the “pitches”) are more technical than usual. Rather than emphasizing general benefits, employers focus more on the technology and programming environment in addition to the usual information about the company. Employer presentations are usually given by technical leads within the company rather than by recruiting representatives.

Fundamentally, the Find Your Ninja project and other similar events are based on the assumption that the audience is filled with highly motivated and competent programmers. If you see yourself in that category and you live in or live close to London, then perhaps this kind of event is for you.

If you don't live near London then contact technical/programmer training companies near where you live, or near where you want to work, to see if they run similar events.



# Contents

<b>Introduction</b>	<b>xxiii</b>
<b>Chapter 1   Hiring Programmers: The Inside Story</b>	<b>1</b>
Reasons They Recruit	2
Planned expansion	3
The interviewer's motivation and approach	3
Your approach	3
Specific projects	5
The interviewer's motivation and approach	5
Your approach	5
Replacing someone	6
The interviewer's motivation and approach	6
Your approach	6
Talking to Managers	7
Tech talk—don't hold back	7
Using metaphors	8
Preparing Your CV	8
Include relevant keywords but keep them in context	9
Write as well as you can	9
Justify your claims of experience	9
Ignore anyone who tells you "strictly one or two pages"	10
Emphasize skills that match the job advertisement	10
Don't leave unexplained gaps between jobs	10
"Reading, music, and cinema"	11
Use a logical layout	11
Graduate CVs	11
CVs containing more experience	12
CVs of those returning to work	12
Avoiding common CV blunders	13

	Poor spelling or grammar	13
	Vague or nonspecific accomplishments	13
	Unclear or cluttered layout	13
	Unprofessional e-mail address	14
	Using Job Sites	14
	Comparison of major job boards	15
	Recruitment Agencies	15
	Working effectively with a recruiter	17
	Searching for Jobs Yourself	17
	Networking for introverts	18
	Finding potential employers	20
	Approaching potential employers	20
	Being persistent	21
	Timing	22
	Emerging Alternatives	22
	Using Twitter	22
	Using Facebook	23
	Using LinkedIn	24
	Writing a Blog	25
	Careers at Stack Overflow	26
	Skills matter: “Find Your Ninja”	27
<b>Chapter 2</b>	<b>Handling the Phone Interview with Confidence</b>	<b>29</b>
	Knowing What to Expect	30
	Preparing your cheat sheets	32
	Relating your experience	32
	Answering hard questions	33
	Asking good questions	34
	Having a phone interview checklist	35
	Using a phone interview cheat sheet template	35
<b>Chapter 3</b>	<b>In-Person Interviews</b>	<b>39</b>
	Preparing for the Interview	39
	Knowing what to expect	40
	Doing your homework	41
	Dressing appropriately	42
	Handling different types of interview questions	42
	Fielding social and behavioral questions	43
	Handling design problems	43
	Tackling technical pop-quiz questions	43
	Fielding the general intelligence test	44
	Dealing with the stress test question	44
	The Most Important Thing	45
	Establishing rapport	46
	It takes work	47
	Be a good listener	47
	Ask good questions	47

	Mirror your interviewer	47
	Look for ways to interact	47
	The Second Most Important Thing	48
	Speaking up	48
	Being aware of how much time you have	48
	Stories are good, evidence is better	49
	Communicating Effectively	49
	Using your passion to combat nerves	49
	Using your hands	49
	Speaking slower than usual	50
	Starting and finishing clearly	50
	Repeating your main point	50
	Spontaneity improves with practice	51
<b>Chapter 4</b>	<b>Negotiating a Job Offer</b>	<b>53</b>
	Understanding the Market	54
	Doing the Numbers	54
	Considering the whole package	55
	Must have, should have, could have	56
	Must have	56
	Should have	56
	Could have	57
	Won't have	57
	The Role of the Recruiting Agent	57
	Start as You Mean to Go On	57
	Avoid overcommitting	58
	Realism and idealism	58
	Evaluating a Contract	59
	Intellectual Property (IP)	59
	Non-compete clauses	60
	Non-solicitation clauses	60
	What to Do If Things Go Wrong	60
	"It's a standard contract"	60
	The silent treatment	61
	Escalation and ultimatums	61
	Summary of Negotiating Tips	61
<b>Chapter 5</b>	<b>Programming Fundamentals</b>	<b>63</b>
	Understanding Binary, Octal, Hexadecimal	64
	Converting hexadecimal to binary	66
	Using unicode	67
	Understanding Data Structures	68
	Using arrays	68
	Using hash tables	69
	Using queues and stacks	70
	Using trees	70
	Using graphs	72
	Understanding graph traversal	72

Sorting	73
Working with Recursion	75
Modeling with Object-Oriented Programming	76
Understanding classes and objects	76
Untangling inheritance and composition	78
Exploring polymorphism	78
Data-hiding with encapsulation	80
Thinking Like a Functional Programmer	81
Understanding SQL	81
What is ACID?	81
Set-based thinking	82
Full-Stack Web Development	82
Deciphering Regular Expressions	83
Finding content with anchors and word boundaries	85
Matching character classes	86
Constraining matches with quantifiers	88
Working with groups and captures	89
Avoiding gotchas	90
More reading	92
Recognizing Hard Problems	92
Questions	93
Answers	95
<b>Chapter 6</b>	
<b>Code Quality</b>	<b>109</b>
Writing Clear Code	110
Writing Expressive Code	112
Measuring Efficiency and Performance	112
Big-O notation	113
Constant, $O(1)$	113
Logarithmic, $O(\log n)$	114
Linear, $O(n)$	114
Quadratic, $O(n^2)$	116
Using big-O	116
Measure performance	117
Consider context	118
Have a goal	118
Measure more than once, take an average	118
Divide and conquer	118
Try the easy things first	119
Use a profiler	119
Understanding What “Modular” Means	119
Understanding the SOLID principles	121
Single Responsibility Principle	121
Open/Closed Principle	123
Liskov Substitution Principle	123
Interface Segregation Principle	124
Dependency Inversion Principle	125

	Avoiding Code Duplication	126
	Questions	128
	Answers	134
<b>Chapter 7</b>	<b>The Usual Suspects</b>	<b>157</b>
	Concurrent Programming	158
	Race conditions	160
	Locks	160
	Deadlocks	165
	Livelocks	166
	Relational Databases	167
	Database design	167
	Normalization	168
	First normal form: “No repeated values”	168
	Second normal form: “No partial dependencies”	169
	Third normal form: “No transitive dependencies”	169
	Boyce-Codd normal form	169
	Beyond BCNF	170
	Denormalization	170
	Populating a normalized database	170
	Pointers	171
	Functions in C receive arguments by value	173
	Arrays in C are handled like pointers	174
	Passing values and references	175
	Design Issues	177
	YAGNI is not an excuse to take shortcuts	177
	Design for performance	178
	Do not trade common sense for a methodology	178
	Bad Habits	179
	Mishandling exceptions	179
	Not being paranoid enough	180
	Being superstitious	181
	Working against the team	182
	Copying and pasting too much	182
	Eager loading	183
	Questions	184
	Answers	186
<b>Chapter 8</b>	<b>Quirks and Idioms</b>	<b>193</b>
	Binary Fractions and Floating Point Numbers	194
	Questions	195
	JavaScript	195
	Questions	195
	C#	198
	Questions	198
	Java	200
	Questions	200

	Perl	201
	Questions	202
	Ruby	205
	Questions	205
	Transact-SQL	206
	Questions	206
	Answers	208
<b>Chapter 9</b>	<b>Testing — Not Just for Testers</b>	<b>245</b>
	Unit Tests	246
	Test-Driven Development	246
	Behavior-driven development	247
	Red, green, refactor	247
	Writing Good Unit Tests	247
	Run quickly	247
	Be simple	248
	Be self-evident	248
	Be helpful when failing	248
	Be self-contained	248
	Testing Slow Things	249
	Unit Testing Frameworks	249
	Mock Objects	251
	Questions	253
	Answers	256
<b>Chapter 10</b>	<b>The Right Tools</b>	<b>265</b>
	Exploring Visual Studio	266
	Questions	266
	Exploiting Command-Line Tools	268
	Questions	269
	Understanding PowerShell	271
	Questions	271
	Troubleshooting with Utilities from Sysinternals	272
	Questions	272
	Managing Source Code	272
	Source control with Team Foundation Server	273
	Questions	273
	Source control with Subversion	273
	Questions	274
	Source control with git	274
	Questions	275
	Answers	275
<b>Chapter 11</b>	<b>Notorious Interview Questions</b>	<b>303</b>
	Estimating on the Spot	303
	Questions	304
	Solving Puzzles and Brain-Teasers	304
	Questions	304

	Solving Probability Problems	306
	Questions	306
	Coping with Concurrency	307
	Questions	307
	Doing Tricks with Bits	308
	Questions	309
	Devising Recursive Algorithms	309
	Questions	309
	Understanding Logic Gates	311
	Questions	313
	Writing Code to...Prove You Can Code	315
	Questions	315
	Answers	316
<b>Chapter 12</b>	<b>Programming Wisdom</b>	<b>351</b>
	Questions	352
	Answers	357
<b>Appendix</b>	<b>Preparing Your Cheat Sheets</b>	<b>395</b>
	General and Behavioral	396
	Programming, General	397
	Programming Concepts	397
	Work History	398
	Questions to Ask, If Given the Opportunity	399
<b>Index</b>		<b>401</b>

# IT'S YOUR DREAM JOB. YOU'RE QUALIFIED. HERE'S HOW TO SEAL THE DEAL

There's more to acing a job interview than correct answers. This down-to-earth guide, written by a programmer who has been on both sides of the desk, covers it all. Learn what your resumé should include, what to expect from the interviewer, how to answer tough questions, why spelling matters, what to wear, and even ways to gain confidence. From preparing a phone interview cheat sheet to code-writing best practices, language quirks, and testing, this complete reference empowers you to ace that interview and land the job.

- Understand how the hiring manager sees the interview process
- Learn what to research before the interview
- Be prepared for social and behavioral questions
- Get tips on communicating effectively and establishing rapport
- Master the most common problems interviewers present
- Conquer quirks and idioms of JavaScript, C#, Java, Perl, Ruby, and T-SQL
- Study the open-ended questions that test a programmer's experience

**Edward Guinness** is a software developer who has been both interviewer and interviewee over his long career. He has been programming since before the birth of Windows 1.0. In 2012, Edward founded SocialCoder ([socialcoder.org](http://socialcoder.org)), a volunteering organization for programmers, designers, and other technical people.



For more information and code downloads, visit  
[www.wiley.com/go/acetheprogramminginterview](http://www.wiley.com/go/acetheprogramminginterview)

Cover Design: Ryan Sneed

Programming / General

**WILEY**



Also available  
as an e-book

ISBN 978-1-118-51856-4

