

**DevSkiller**

The definitive  
book of  
**developer  
interview  
questions**  
for savvy  
recruiters

**259**  
INTERVIEW  
QUESTIONS



# Table of contents

<b>Introduction</b>	<b>3</b>
<b>Section I 89 General interview questions</b>	<b>4</b>
1 Behavioral interview questions	5
2 Phone interview questions	11
3 Second interview questions	13
4 Situational interview questions	14
<b>Section II 170 Technical interview questions that work</b>	<b>15</b>
5 Questions that work with any technology	17
6 Java interview questions	17
7 SQL interview questions	18
8 JavaScript interview questions	18
9 Python interview questions	18
10 .NET Core and .NET interview questions	18
11 C and C++ interview questions	19
12 Android interview questions	20
13 Web developer interview questions	20
14 Scala interview questions	23
15 iOS interview questions	23
16 Ruby on Rails interview questions	24
17 Security engineer interview questions	24
<b>Section III Why whiteboard interview questions don't work</b>	<b>27</b>
18 Whiteboard interview questions to avoid	28
19 Technical interviews focused on work sample coding tasks	28

# Introduction

Software developer interview questions take time to prepare. As a recruiter or hiring manager, you need to get to the heart of how the developer thinks and solves problems. On top of that, you need to get to the bottom of what they know about the tech stack the position requires.

No conversational interviews can replace seeing your candidate actually do the work they will be doing on the job. But when coupled with a work sample coding test or code pair interview, these questions will give you a much deeper understanding of your candidate's fitness for the role. This approach allows you to screen out more candidates earlier in the funnel so that you only spend your time on candidates who could fill the role.

This ebook is for the efficient recruiter and hiring manager who doesn't always have time to organize a full slate of interview questions for each interview. In addition to general interview questions, you can find technical questions for 12 different tech stacks. Also, find out what questions not to ask (hint: they involve whiteboards).

We know you are busy so take a look at these questions and have an amazing interview.



## Section I

# 89 General interview questions

General interview questions can be used by recruiters for any technical position. They do not require any technical expertise. Instead, they will help you get to know your candidate's approach to their work. Use these questions to get to know how your candidate thinks and how they solve problems.



### Questions in this section:

- Behavioral questions
- Phone interview questions
- Second interview questions
- Situational interview questions

## 1. Behavioral interview questions

Behavioral questions are used to evaluate a candidate's past experiences and behaviors in order to determine their potential. They involve asking the candidate about how they resolved a situation in the past.

Behavioral-based questions reveal in-depth information about the way a candidate thinks, feels, and what type of experience the candidate has from previous jobs.

### 45 Behavioral interview questions

#### → → **Action-oriented and self-motivation** →

1. Describe a situation when you did much more than was expected of you to get the project done. Were your efforts recognized? By whom and how? How did that make you feel?
2. Tell me about a time when you took ownership of a project. Why did you do this? What was the result of you taking on the challenge? What could have happened if you did not take ownership?
3. Think about an instance in which you came up with a project idea which was implemented primarily because of your efforts. What was it about? What was its outcome? What was your role?
4. Describe a time when you made a suggestion to improve something on the project that you were working on.
5. Give me an example of the project or initiative that you started on your own. It can be a non-business one. What prompted you to get started?



## Ability to adapt

6. Describe a situation in which you met a major obstacle in order to complete a project. How did you deal with it? What steps did you take?
7. Tell me about a time you had to work on several projects at once. How did you handle this?
8. Describe a situation in which you experienced a significant project change that you weren't expecting. What was it? How did that impact you, and how did you adapt to this change? How did you remain productive through the project?
9. Describe a situation in which you had to adjust to changes over which you had no control. How did you do this?



## Communication skills

10. I'd be interested in hearing about a miscommunication you had with your supervisor. How did you solve it? What was the reason for that? How did you deal with that situation?
11. Tell me about an instance when you had to communicate a really bad piece of news to your supervisor or team members. How did you handle it? What was the outcome?
12. Give an example of a time when you didn't agree with another developer. Did you stand up for something that you believed was right?
13. Tell me about a time when you had to present a complex programming problem to a person that doesn't understand technical jargon. How did you ensure that the other person understood you?
14. Describe a situation in which you felt you had not communicated well enough. What did you do? How did you handle it?
15. Tell me about a situation that you had to speak up and be assertive in order to get a point across that was important for you.



## Conflict management

16. Tell me about a time when you had a disagreement with another programmer. How did you handle the situation? Were you able to reach a mutually beneficial resolution to that conflict? If not, why were you and your co-worker unable to reach a mutually beneficial resolution? If you knew then what you know now, what would you have done differently to either prevent the conflict or to resolve it?
17. Tell me about a time when you had to work with a difficult person to accomplish a goal. What was the biggest challenge? How did you handle it?
18. Has there been a time on a project when you disagreed with someone? What did you do about it?
19. Tell me about when you had to deal with conflict within your team. How was the conflict solved? How did you handle that? How would you deal with it now?



## Creativity

20. Give me an example of a time you had to take a creative and unusual approach to solve a coding problem. How did this idea come to your mind? Why do you think it was unusual?



## Decision making

21. Give me an example of a time when you were faced with a complex project related matter and you could not decide on the best way to deal with it. What did you do? How did you go about making the decision? Can you lead me through your decision process? If you could make the decision once again, would you change anything?
22. Think about an instance in which you made a decision at work that was unpopular. How did you handle it?



23. Give me an example of a project that completely failed. Why do you think it was a failure? Could there be anything done differently in order to turn it into a success?
24. Describe a situation in which you worked diligently on a project and it did not produce the desired results. Why didn't you get the desired results? What did you learn from the experience?
25. Think about a situation when you made a poor decision or did something that just didn't turn out right. What happened?



## Goal orientation

26. Give an example of an important project goal you reached and how you achieved it.
27. Think about an instance in which you worked on and achieved multiple project goals.
28. Describe a circumstance when you were not able to achieve a project goal that was set by your supervisor. How did you handle this situation? What was the outcome?
29. Think about an instance in which you had to depend on others to help you achieve a project goal. How did you feel?



## Influence and persuasion

30. Tell me about a recent situation at work in which you were able to get management to accept one of your ideas.
31. Describe a situation in which you experienced difficulty in getting others to accept your ideas? What was your approach? How did this work? Were you able to successfully persuade someone to see things your way?
32. Have you ever had to "sell" an idea to your project team? How did you do it? Did they "buy" it?





## Planning, priority setting, time management

- 33.** Tell me about a situation when you were responsible for project planning. Did everything go according to your plan? If not, then why and what kind of counteractions did you have to take?



## Problem-solving skills

- 34.** Tell me about a situation when you made a mistake at work. What happened exactly and how did you deal with it? What steps did you take to improve the situation?
- 35.** What is the biggest problem you have faced on projects so far and how did you solve it? What made the problem difficult to resolve? What was the result? Would you do anything differently now?
- 36.** Give me an example of a time when you noticed a small problem before it turned into a major one. Did you take the initiative to correct it? What kind of preventive measure did you undertake?
- 37.** Walk me through a difficult/complex problem/project you encountered. How did you decide what to do first? What information did you need? What obstacles did you face? Which ones were you able to overcome? Did you have to ask for help?



## Teamwork

38. Tell me about a time when you worked with someone who was not completing his or her share of the work. How did you handle the situation? Did you discuss your concern with your coworker? With your manager? If yes, how did your coworker respond to your concern? What was your manager's response?
39. Describe a situation where you had to work in a team that didn't get on very well. What happened? What did you do and what role did you take? How did the situation evolve?
40. Describe a team experience you found disappointing. What would you have done to prevent this?
41. Give me an example of working cooperatively as a team member to accomplish an important goal. What was the objective? To what extent did you interact with other project members?
42. Tell me about the most difficult situation you have had when leading a team. What happened and how did you handle it? Were you successful? What was the most important thing you did?



## Working under pressure

43. Describe a situation when you worked effectively under pressure. How did you feel when working under pressure? What was going on, and how did you get through it?
44. Tell me about a situation when you had problems working under pressure. How did you handle that situation? Did you decide to ask for support? How and when did you ask for help?
45. Give me a recent example of a stressful situation on the job. What happened? How did you handle it?

## 2. Phone interview questions

Phone interviews are a lean way of getting an idea about who your candidate is and what they believe they can achieve before they take a technical assessment. Compared to in person interviews, they take less time and are less disruptive to the candidate. Combining the information you get from the phone interview with a technical assessment will allow you to screen out any unsuitable candidates before any face-to-face interaction, reducing the number of technical interviews you need to conduct. The best questions to ask in a phone interview aim to verify a number of things, from job experience to working preferences, expectations, and attitude, to name but a few.



## 20 Phone interview questions

1. Could you tell me a little bit about yourself?
2. Why did you apply for this position?
3. What are you looking for in your next job?
4. Why does this position appeal to you?
5. How can you contribute and help us grow?
6. Tell me about your current or most recent job. What did you do?
7. Could you describe your typical work week?
8. In your previous roles, what major challenges and problems did you face?  
How did you handle them?
9. What challenges do you look for in a position?
10. Do you prefer working on your own or as a part of the team?
11. Which work environments do you function best in?
12. What keeps you motivated?
13. What are your biggest strengths?
14. What are your biggest weaknesses?
15. Why do you want to make a career change?
16. What are your long-term career goals?
17. What are your salary expectations?
18. Are you willing to relocate for this role?
19. If you were offered this job, when would you be able to start?
20. Are there any questions that I can answer for you?

### 3. Second interview questions

Once candidates pass the screening and the first interview stage, they often get invited to the second interview. The second interview is used to either assess company fit in a second soft skills interview or verify technical skills through a technical interview.

Second interview questions are more detailed than the questions used in the first interview. They are often asked to inquire about specific skills, attitudes, personality traits, patterns of behavior, or particular events.

#### 12 second interview questions

1. What are your career goals?
2. Was there a time in your career when a project was changed? How did you adapt?
3. What are the main attributes needed to be successful in this role?
4. How have you resolved a conflict with a colleague or superior?
5. Could you tell me a little more about your current or most recent job?
6. What challenges are you currently looking for?
7. Have you ever been assigned to more than one project? How did that affect your work?
8. What role do you typically take on a team?
9. What would you change about the company?
10. Can you tell me about a time when you worked successfully as part of a team?
11. What salary are you seeking?
12. What is your notice period?

## 4. Situational interview questions

Situational interview questions are questions about how the candidate would handle a challenge they would face in the role. They can help push candidates past generic answers by making them think about how they would handle a situation. Situational interview questions are a great way of establishing how a candidate's values and priorities might influence a future action at your company.

### How to structure a situational interview question

#### 1. "What would you do..."

*These 5 words can turn a simple yes or no question into a detailed, intricate response. Try rewording your favorite interview questions to start with this phrase. You'll be able to see how candidates respond to the same question, just worded differently.*

### 11 more situational interview questions

2. How would you fix a mistake that you make in a project?
3. How would you motivate someone or others around you?
4. Imagine you found yourself in a situation where you couldn't achieve your goals. What would you do?
5. Describe how you would prioritize, organize, and track your work.
6. Describe a situation where you would be proud of your work.
7. Tell me about how you would navigate a situation where you had multiple projects with conflicting deadlines or goals.
8. What would you do if you had a disagreement or conflict with a co-worker and what would your role be in resolving it?
9. What would you do if you felt nervous, stressed or unconfident?
10. How would you handle a situation that regularly happens in your workplace?
11. What would you do if you had to work for your least favorite boss or manager, and why?
12. How would you adjust to big changes in your workplace?



## **Section II**

# 170 Technical interview questions that work

Technical interview questions will help you learn about your candidate's technical knowledge. Hiring managers can use these to get a feel for how well the candidate understands the specific technologies they need for the position. Unlike the general interview questions in the first half of the book, the answers to these questions require a technical background to understand, so should be asked by a technical interviewer.

Technical interview questions should be used along with a work sample coding test. The candidate's answers can show their breadth of knowledge but they can't show if the candidate can apply their knowledge practically.





## Questions in this section:

- Questions that work in any technology
- Java interview questions
- SQL interview questions
- JavaScript interview questions
- Python interview questions
- .NET Core and .NET interview questions
- C and C++ interview questions
- Android interview questions
- Web developer interview questions
- Scala interview questions
- iOS interview questions
- Ruby on Rails interview questions
- Security engineer interview questions

## 5. Questions that work with any technology

1. With which technologies listed on your resume have you had commercial experience in the past 2 years? What were your responsibilities? What was your biggest achievement?
2. Think about a programming project decision you made that was a failure. Why do you think it was a mistake? Why did it happen? Could there be anything done differently in order to turn it into a success? What steps did you take to improve the situation? What did you learn from this experience?
3. I've noticed you listed framework/technology X on your resume. How is it used? What's your opinion about it? Is it a good choice?
4. On what stage did you join recent projects? Were you involved in the choice of technology or project setup? If yes, which technology did you choose or recommend for the project and why?
5. What was the most interesting project you've participated in? What was your role? Can you describe it and explain why you considered it to be so attractive?
6. Do you like to participate in the analysis, design, and deployment phases of an IT project or do you prefer to concentrate on the pure development of a well-described task? Why?
7. What is your biggest programming success story? Why did it happen? How can you repeat it?
8. For more senior-level applicants: Would you like to mentor a junior developer? Why? How would you go about doing it? Do you have any experience mentoring other people?



## 6. Java interview questions

9. Which Java open source libraries do you consider to be valuable and why?

 **Why should I ask this and other questions to a Java candidate?**



## 7. SQL interview questions

10. On what stage did you join recent projects? Were you involved in the choice of technology or project setup? If yes, which technology/ relational database management system did you choose or recommend for the project and why?

[Why should I ask this and other questions to an SQL candidate?](#)



## 8. JavaScript interview questions

11. Which JavaScript libraries do you consider to be valuable and why?
12. If you would like to learn new technology connected to JavaScript, what would it be?
13. Are you contributing to any Open Source project or maybe are you maintaining your own Open Source Project? Are you attending any JavaScript conferences?

[Why should I ask these and other questions to a JavaScript candidate?](#)



## 9. Python interview questions

14. Which Python open-source libraries do you consider to be valuable and why?

[Why should I ask these and other questions to a Python candidate?](#)



## 10. .NET Core and .NET interview questions

15. Could you explain the difference between similar frameworks A and B (for example the difference between ASP.NET MVC and Web Forms)?
16. I've noticed you listed framework X on your resume. What's your opinion about it? Is it a good choice?

[Why should I ask these and other questions to a .NET candidate?](#)



## 11. C and C++ interview questions

17. (C/C++) (C/C++) What were the constraints for your previous projects?
18. (C/C++) Which systems did you program for?
19. (C/C++) Were you coding adhering to any specific standards?
20. (C++) What are the main differences between C++ and C?
21. (C/C++) What is pointer arithmetic?
22. (C++, entry-level) What is the difference between a class and an object?
23. (C++) What is a lambda expression?
24. (C/C++) What are locks, what problems do they solve and what are the potential problems with them?
25. (C/C++, expert) What is volatile, and how does it relate to the question about locks and synchronization?
26. (C++) How would you create a dynamic array?
27. (C++) What is RAII? Do other languages have it?
28. (C++) Can you throw from a destructor?
29. (C++) Can you inherit a constructor?
30. (C++) Can you have a virtual constructor?
31. (C++) What is an interface?
32. (C++, expert) Can you have an implementation of a pure virtual function?
33. (C++, expert) Can you have a virtual template function and why?
34. (C++, expert) How would you implement `std::is_same`?
35. (C/C++) What is your favorite change (or a number of changes) in C++?
36. (C++) What are the differences between C++98 and C++11?
37. (C/C++) How would you detect and fix a memory corruption bug?
38. (C/C++) Do you have experience using custom allocators?
39. (C/C++) You have framework/library X in your résumé, describe your experience with it. Was using it a good choice? Is there an alternative that you had preferred, or would prefer now?

40. (C/C++) Which is your preferred build system and why? How does it compare to the competition?

 **Why should I ask these and other questions to a C or C++ candidate?**



## 12. Android interview questions

41. Tell us about the most interesting problem you encountered after publishing an application for users. How did you solve this problem?
42. Enumerate ready-made components of Material Design whose implementations can be found in the Support library family.
43. What is your favorite shortcut in Android Studio?
44. What do you have to do to make an Android application freeze?
45. How do you deal with fragmentation?
46. It is also a good idea to ask about the source of the developer's projects, i.e. PSD or provided by a designer. How do they deal with cutting graphics and cooperating with designers?

 **Why should I ask these and other questions to an Android candidate?**



## 13. Web developer interview questions

### General

47. Have you recently learned something new or interesting?
48. What made you interested in programming?
49. In which programming environment do you feel most comfortable?
50. What is the piece of code you are most proud of? Are you working on any personal projects at the moment?
51. What industry sites and blogs do you read regularly?
52. Do you prefer working alone or in a team?
53. What size websites have you worked on before?

54. From a web software developer's perspective, what sites do you admire and why?
55. What's your favorite development language and why? What other features (if any) do you wish you could add to this language?
56. Do you find any particular languages or technologies intimidating?

## Fact-based

57. What is the difference between tags and HTML elements?
58. What is "Semantic HTML"?
59. How do you optimize a website's assets?
60. What are three ways to reduce page load time?
61. What kind of things must you be wary of when design or developing for multilingual sites?
62. What does DOCTYPE mean?
63. What's the difference between standards mode and quirks mode?
64. What are the limitations when serving XHTML pages?
65. What is the syntax difference between a bulleted list and numbered list?
66. How do you make comments without text being picked up by the browser?
67. What is the difference between linking to an image, a website, and an email address?
68. What is the difference between <div> and <frame>?
69. What is the difference between the page model of HTML and HTML5?
70. Ok, what's the real difference between HTML and HTML5?
71. What are some of the major new API's that come standard with HTML5?
72. What is the difference in caching between HTML5 and the old HTML?
73. What is the new DOCTYPE?
74. What are some new HTML5 markup elements?
75. What elements have disappeared?
76. What are the new media-related elements in HTML5?
77. What are the new image elements in HTML5?
78. What is the difference between SVG and <Canvas>?
79. What are some new input attributes in HTML5?

80. What are data-attributes good for?
81. What purpose do Work Workers serve and what are some of their benefits?
82. Describe the difference between cookies, sessionStorage, and localStorage.
83. How do you optimize your web pages for print?
84. What existing CSS frameworks have you used locally, or in production?  
How would you change/improve them?
85. How is responsive design different from adaptive design?
86. Explain how a browser determines what elements match a CSS selector.
87. What is the difference between classes and IDs in CSS?
88. What's the difference between "resetting" and "normalizing" CSS?  
Which would you choose, and why?
89. Explain Ajax in as much detail as possible.
90. What's the difference between .call and .apply?
91. What's the difference between an "attribute" and a "property"?
92. Why is extending built-in JavaScript objects not a good idea?
93. Is jQuery is a replacement of JavaScript?
94. What are the advantages of jQuery?
95. Which is the fastest selector in jQuery? Which is the slowest?
96. Where jQuery code is getting executed?

### **Situational or hypothetical questions for web developers**

97. I just pulled up the website you built and the browser is displaying a blank page. Walk me through the steps you'd take to troubleshoot the problem.
98. The website is not rendering correctly on different devices. What are the first steps you would take to correct that?
99. We're developing an eCommerce website for a small store. Give me a list of requirements and a time-frame for delivering the entire project.



- 100. A project needs to be deployed in one month but the code, written by a previous developer, is messy and not functional. Do you re-write or start from scratch? Why and what does it depend on?
- 101. We have a need to use a technology you are not familiar with. Would you begin learning it or outsource/ask for outside help?



## 14. Scala interview questions

- 102. What is pattern matching?
- 103. What are case classes?
- 104. Which object-oriented patterns are implemented in Scala?
- 105. What is a trait and is there any equivalent in Java?
- 106. Which Java open source libraries do you consider to be valuable and why?

 [Why should I ask these and other questions to a Scala candidate?](#)



## 15. iOS interview questions

- 107. Please compare Swift and Objective-C.
- 108. Which iOS technologies are in wide use now, and which may become popular in the future?
- 109. How can you avoid memory leaks?
- 110. Have you had experience tutoring and mentoring someone in Swift?

 [Why should I ask these and other questions to an iOS candidate?](#)



## 16. Ruby on Rails interview questions

- 111. What are the changes between Rails 4 and Rails 5?
- 112. In Ruby on Rails, where do you write your logic: Models, View or Controllers?  
Is there any alternative?
- 113. What are the main difference between rspec and minitest?
- 114. Given a class that has these methods,
  - a. `def factorial(number):` returns the factorial of `#number`
  - b. `def fibonacci(n):` calculates the value of the nth Fibonacci number
 what unit test would you write?
- 115. You have a code and tested a ticket for a new feature on a new page. Just after deploying it to production we detect that the new page loads, but it takes one minute. What do you look at?
- 116. Why did you choose Ruby on Rails?
- 117. What does Rails add to Ruby?

 [Why should I ask these and other questions to a Ruby on Rails candidate?](#)



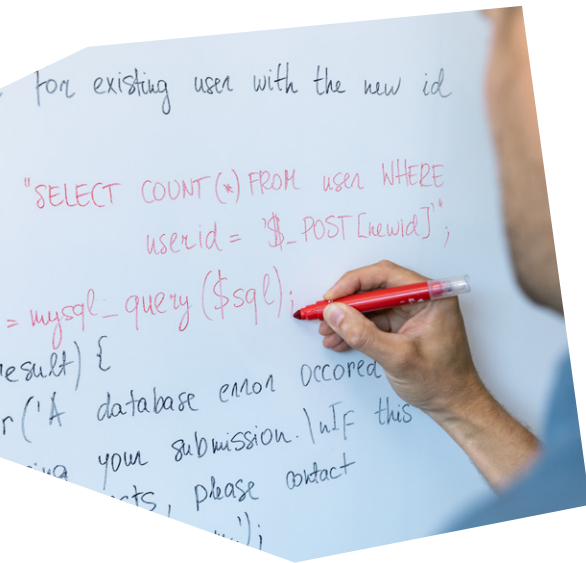
## 17. Security engineer interview questions

- 118. Have you handled a breach? How did it happen? How could it be prevented?
- 119. What's your opinion about the security engineer role in the company?
- 120. What do you think about BYOD (bring your own device)?
- 121. What is a threat, vulnerability, exploit, and mitigation? (explain)
- 122. What is a SQL Injection and how it differs from XXE? (explain)
- 123. What leads to SSTI (server-side template injection) and is it more dangerous than XSS? How do they differ?
- 124. What are: IDS, IPS, and EDR. How they differ?

125. How does asymmetric encryption work? When should you use it? What are the pros and cons in comparison to symmetric encryption? Name one symmetric and one asymmetric encryption algorithm.
126. What is the difference between stream cipher and block cipher?
127. What is hashing (cryptographical), what it is used for, when, and how does it differ from encryption? Name one hashing algorithm that should not be used and one *"not proven unsecure"*.
128. What is PBKDF, how does it work? Why use it?
129. How CSRF differs from XSS?
130. What is a fingerprint?
131. How to check if the downloaded file is correct?
132. Explain the CIA principle.
133. What is port knocking?
134. Name a secure protocol to manage remote servers?
135. What is rlogin and should it be used? Why? Why not? Explain.
136. What is hardening?
137. What is penetration testing? What is vulnerability assessment? How do they differ? What is a security audit?
138. Name one pentesting guide.
139. What is PKI (public key infrastructure)? How does it work?
140. What is Kerberos? What it's used for? Can it be used in Windows domains?
141. What is certificate pinning? How to do it properly?
142. What you do when your private certificate is stolen?
143. Name one popular vulnerability scanning tool?
144. What is a blue team, red team, and purple team? Which one is the most important one?
145. What is DLP, how does it work?
146. What is WAF? Name one WAF solution.
147. What is SOP (same origin policy)?
148. What is CSP (content security policy), when should it be used?

149. How to mitigate SQL Injection?
150. What is HSTS? Why should you use it?
151. Explain how TLS works (in a few sentences).
152. What is the difference between authorization and authentication?
153. What are ACLs? How to use them?
154. Name levels of confidentiality.
155. What is RADIUS? When should you use it?
156. What is VLAN, when should you use it? How does VLAN hopping work?
157. How to secure WiFi in an organization? (network separation)
158. Name three ways of security testing depending on the level of knowledge of the attacker. Which one is the most reliable and simulates a real-world scenario?
159. Name every layer of the ISO/OSI model.
160. What is residual risk?
161. Imagine you work for a small company. There are several interns employed each month for a short period of time. They need access to some servers and a WiFi network. How will you handle it?
162. What is a password manager? What should it be used for?
163. Which policy is better – blacklisting or whitelisting, and why?
164. Define what a man in the middle attack is.
165. How does the Diffie-Hellman key exchange (DHKEX) work?
166. What is SIEM and how does it work?
167. What are DoS and DDoS? What's the difference?
168. How do you prevent DNS spoofing and how do you secure a DNS?
169. The last two years were occupied by ransomware attacks that caused havoc in organizations and companies which caused giant financial and reputational losses. What steps would you take to prevent such accidents happen in your organization?
170. Your IDS reported a breach. What would you do to eliminate the threat?

 **Why should I ask these questions to a security engineering candidate?**



### Section III

## Why whiteboard interview questions don't work

Whiteboard interviews are still common in tech recruitment but in most cases are inefficient.

Whiteboard interviews are essentially tech quizzes which tell you very little about real coding skills. Candidates are asked to invert binary trees on a whiteboard or recall algorithms off the top of their head. To quote coding instructor Quincy Larson,

"Unfortunately, interviewing practices at big tech companies aren't that scientific. The decision of whether to hire a developer usually comes down to the candidate walking up to a whiteboard and regurgitating algorithms that haven't changed since the 1970s, like a (classically) trained monkey."

Whiteboard testing puzzles don't resemble the day-to-day work of developers – there are no computers, no access to reference materials. This scenario is unrealistic and as such, tells you very little about one's real coding abilities. As Quincy Larson wrote,

*"The only world where you would actually need to be able to recall an algorithm would be a post-apocalyptic one, where the hard drives of all the computers connected to the internet were fried, and all copies of foundational academic papers and computer science textbooks had been reduced to ashes."*



## 18. Whiteboard interview questions to avoid

Here are a **few common questions** that you should avoid:

1. How do you find the missing number in a given integer array of 1 to 100?
2. How do you find the middle element of a singly linked list in one pass?
3. How do you print duplicate characters from a string?
4. How do you perform an inorder traversal in a given binary tree?

So if whiteboard interview questions don't work, what is the solution?



## 19. Technical interviews focused on work sample coding tasks

Ask your candidate to complete a task which meets the following criteria:

- The test is an authentic work sample
- It gives your candidate all of the resources they would normally use at work
- It bases the task on a business problem they will face when they start working for you

These tasks should be tailored to the work of the position you will be filling so we can't feed you questions that work for every developer who uses the tech stack. You can find tests you can use in our **coding test catalog** or make your own. Here are the requirements Devskiller has for creating effective coding tasks:

# The 4 key requirements for creating quality tasks on **DevSkiller**

1.

Tasks are created by reputable subject-matter experts, also from outside our team. Due to the fact that our test library comprises of over 57 languages, frameworks, and libraries, we use both our development team as well as subject-matter experts who specialize in the tech stack we want to cover

2.

All tasks are in alignment with our **RealLifeTesting™** methodology so that they mirror real work. Task concepts are based on real-life work and first-hand experience of the developers creating the tasks, our Tech Team, and most importantly, our customer base (based on their experience and internal needs)

3.



Multiple quality assurance touchpoints are created for every project and every expert, regardless of their level of expertise. This is a safeguarding mechanism to ensure that each and every task added to our library meets our standards

4.

Assigning the right difficulty levels to tasks is based on the results of beta testing and the suggestions of additional technology experts. Furthermore, the scores of each test are constantly monitored after they are published on the platform. In some cases, test difficulty levels and/or time limits are adjusted based on these numbers



Here is **a task** created using these requirements:

 **Senior JavaScript Developer | React, Redux | Address Book**  
Tested skills: AJAX ES6 JavaScript React Redux Web Development

---

TEST DURATION	Contains following tasks:
98 minutes	1) <i>Choice questions</i> - assessing knowledge of ES6, JavaScript, AJAX, Web development
OVERALL SENIORITY LEVEL	2) <i>Programming task</i> [level: Hard] - JavaScript   React, Redux   Address Book - Implement missing features of a small web application built on top of
<span>Senior</span>	

It doesn't ask a developer to come up with a complicated algorithm to get a predefined result. Instead, the tasks ask the candidate to do real software development, in this case adding features to an address book application using React built on Redux. Through doing this, the developer needs to show expertise in the entire tech stack by solving a real business problem.

This task can be used in a code pair interview or as an automatic tech screen. When it is a part of a tech screen, Devskiller will automatically evaluate the solution the developer comes up with based on whether it works, the cleanliness of the code and the efficiency of the solution.

This will tell you whether your candidate has the right coding skill, tech stack experience, and problem-solving ability that you need for the position you want to fill.

At Devskiller, we've made it easy for companies to create their own work sample tasks or to choose from our **extensive catalog** of premade work sample coding tasks in 57+ technologies.

Like our interview question ideas?

**You'll love our technical screening platform!**

DevSkiller is the CRM for tech skills.  
Find out how you can assess and develop  
tech skills seamlessly.

[Book a demo →](#)

**DevSkiller**

[devskiller.com](https://devskiller.com)

