# 104 Ember.js interview questions to hire top engineers

## Questions

1. What is Ember.js, in simple terms?

2. Can you describe the Ember.js component lifecycle?

3. What's the role of Ember CLI?

4. What are Ember routes used for?

5. Explain Ember's data-binding concept.

6. What is a Handlebars template in Ember?

7. Describe the purpose of Ember models.

8. What is an Ember controller and what does it do?

9. How does Ember handle events?

10. What are Ember services and when would you use them?

11. What is the purpose of Ember addons?

12. Explain the concept of 'Ember way' of doing things.

13. How do you test an Ember application?

14. What are Ember mixins and how are they useful?

15. How does Ember handle application state?

16. What is the role of the Ember Inspector?

17. Explain the difference between `{{outlet}}` and components.

18. How do you handle asynchronous operations in Ember?

19. What are some advantages of using Ember.js?

20. Describe a situation where you might not choose Ember.js.

21. What is Ember Data, and what problem does it solve?

22. How do you define relationships between models in Ember Data?

23. What is the purpose of the `{{each}}` helper in Handlebars?

24. How do you pass data from a route to a component?

25. Describe Ember's approach to URL management.

26. What are some best practices for structuring an Ember application?

27. How do you manage complex component communication in Ember.js, especially when dealing with deeply nested components?

28. Explain the concept of Ember Data's adapter and serializer. How would you customize them to work with a non-RESTful API?

29. Describe the Ember.js run loop and its significance. How can you schedule tasks within the run loop?

30. What are some strategies for optimizing Ember.js application performance, particularly concerning rendering and data loading?

31. Explain the different types of Ember.js acceptance tests and their purpose. How do they differ from integration tests?

32. How would you implement authentication and authorization in an Ember.js application, including handling user sessions and permissions?

33. Describe the use of Ember.js route lifecycle hooks (model, setupController, activate, deactivate) and when each is executed.

34. How can you effectively use Ember.js mixins to share functionality across multiple components or routes?

35. Explain Ember concurrency and how it helps manage asynchronous tasks in your Ember.js applications.

36. How do you handle error states and display user-friendly error messages in an Ember.js application?

37. Describe different approaches to managing application state in Ember.js, including Ember Data, services, and custom state management solutions.

38. How do you implement and use Ember.js custom helpers to format data or perform other view-related tasks?

39. What are some best practices for writing maintainable and testable Ember.js code?

40. Explain how you can integrate third-party JavaScript libraries or frameworks into an Ember.js application.

41. How would you structure a large Ember.js application to improve code organization and maintainability?

42. Describe how you can use Ember.js's dependency injection system to improve the testability of your components and services.

43. What are some common pitfalls to avoid when working with Ember.js, and how can you prevent them?

44. Explain the concept of Ember Engines and when you might use them in a large Ember.js application.

45. How can you use Ember.js to build accessible (A11y) web applications?

46. Describe how you would handle internationalization (i18n) and localization (l10n) in an Ember.js application.

47. How do you use Ember CLI addons to extend the functionality of your Ember.js applications?

48. Explain how to debug an Ember.js application effectively.

49. Describe the differences between Ember.js tracked properties and computed properties and when you might use one over the other.

50. How do you implement responsive design principles in your Ember.js components and templates?

51. Explain how you can use Ember.js services to share data and functionality between different parts of your application.

52. How would you approach upgrading an older Ember.js application to a newer version?

53. Describe how you can implement server-side rendering (SSR) with Ember.js.

54. Explain the purpose and benefits of using Ember Octane features in a new or existing Ember.js project.

55. How would you implement a performant infinite scroll in Ember, and what are the key considerations?

56. Explain the difference between `{{each}}` and `{{#each-in}}` and when you would use each?

57. Describe a scenario where you would use Ember Engines, and what are the benefits of using them?

58. How can you optimize Ember's rendering performance, and what tools can you use to identify bottlenecks?

59. What is Ember's FastBoot, and how does it improve SEO and initial load time?

60. Explain how you would implement authentication and authorization in an Ember application.

61. How do you handle different environments (development, staging, production) in an Ember application?

62. Describe the role of Ember CLI addons and how you would create your own addon.

63. How do you test Ember components that rely on external services or APIs?

64. Explain the concept of Ember Data's adapter and serializer, and how you would customize them.

65. What are some strategies for handling complex data relationships in Ember Data?

66. How do you implement a custom route transition in Ember?

67. Describe how you can use Ember concurrency to manage asynchronous tasks.

68. Explain how you would implement a real-time collaboration feature in an Ember application.

69. How would you integrate Ember with a design system library like Material UI or Bootstrap?

70. What are the trade-offs of using Ember Octane versus Classic Ember?

71. How can you ensure accessibility (a11y) in your Ember applications?

72. Describe strategies for managing and deploying large Ember applications.

73. How do you handle internationalization (i18n) and localization (l10n) in an Ember application?

74. Explain the role of WebSockets in an Ember application, providing example use cases.

75. Discuss techniques for debugging memory leaks in Ember applications.

76. How do you approach code splitting in Ember to improve initial load time?

77. Describe the process of upgrading a large Ember application to a newer version of Ember.

78. How would you profile an Ember application to identify performance bottlenecks?

79. Explain different approaches to state management in Ember beyond Ember Data, like using tracked properties with services.

80. How would you implement a complex form with dynamic fields and validations in Ember?

81. Discuss how to use Ember's testing framework to write integration tests that simulate user interactions.

82. What are some advanced techniques for using computed properties in Ember, such as dependent keys and caching?

83. How would you optimize Ember Data's store for handling a very large dataset, considering memory usage and performance?

84. Explain the trade-offs between using Ember's built-in component lifecycle hooks and using a more reactive approach with tracked properties and @computed.

85. Describe a scenario where you would choose to use Ember Engines over other code-splitting techniques, and why?

86. How can you implement a custom routing solution in Ember that goes beyond the capabilities of the built-in router?

87. Discuss the implications of using native classes in Ember and how they interact with the Ember object model.

88. Explain how you would approach testing an Ember application that heavily relies on web sockets for real-time updates.

89. Describe the challenges of migrating a large, legacy Ember application to the latest Ember Octane features, and how would you tackle them?

90. Explain how you would go about debugging a memory leak in an Ember.js application. What tools and strategies would you use?

91. Describe a situation where you might choose to bypass Ember Data and interact directly with an API. What are the considerations?

92. How would you implement an undo/redo feature in an Ember application, focusing on data management and state preservation?

93. Explain how to optimize the rendering performance of a component with a very complex template and many dynamic bindings.

94. Discuss the pros and cons of using Ember CLI addons extensively in a large project. How do you manage addon dependencies?

95. How can you ensure accessibility (a11y) in a complex Ember application with dynamic content and interactions?

96. Describe how you would implement a custom authentication and authorization system in Ember that integrates with a backend API.

97. How do you handle different environments (development, staging, production) with different API endpoints and configurations in Ember?

98. Explain how you would implement server-side rendering (SSR) for an Ember application to improve SEO and initial load time.

99. Describe strategies for optimizing Ember CLI build times in a large project with many components and dependencies.

100. How can you implement a complex form with dynamic fields and validation rules in Ember, ensuring a good user experience?

101. Explain how you would implement a custom component that efficiently handles a large amount of data using virtual scrolling.

102. Discuss the challenges of maintaining code quality and consistency in a large Ember team. How do you enforce best practices?