Ingeniero de Frontend: Entrevista

Empezando con HTML:

1. Etiquetas Semánticas: Entender y usar `<article>`, `<section>`, `<header>`, `<footer>`, `<nav>`.
2. Formularios: Implementar validación, manejar `<input>`, `<textarea>`, `<select>`, `<button>`.
3. Tablas: Crear tablas accesibles con `<table>`, `<thead>`, `<tbody>`, `<tfoot>`.
4. Metadatos: Usar etiquetas `<meta>` para charset, viewport y SEO.
5. Enlaces y Anclas: Entender etiquetas `<a>`, atributos `href`, `target` y `download`.
6. Elementos Multimedia: Usar `<img>`, `<video>`, `<audio>` correctamente con atributos como `src`, `alt`, `controls`.
7. Listas: Crear listas ordenadas `<ol>` y desordenadas `<ul>`, incluyendo listas anidadas.
8. Encabezados: Usar jerarquía de encabezados correcta `<h1>` a `<h6>`.
9. Incrustar Contenido: Usar `<iframe>`, `<embed>` y `<object>` para incrustar contenido externo.
10. APIs de HTML5: Familiaridad con Geolocalización, Web Storage y Fetch API.

Ahora, CSS:

11. Modelo de Caja: Entender margen, relleno, borde y cómo afectan el diseño.
12. Flexbox: Dominar alineación, envolvente y ordenamiento con propiedades de Flexbox.
13. Diseño de Cuadrícula: Crear diseños complejos usando CSS Grid.
14. Diseño Responsivo: Usar consultas de medios, etiqueta meta de viewport y imágenes responsivas.
15. Preprocesadores CSS: Conocimiento de sintaxis y características de Sass, Less o Stylus.
16. CSS-in-JS: Entender marcos como styled-components o emotion.
17. Animaciones y Transiciones: Implementar transiciones suaves y animaciones de keyframes.
18. Estilizar Formularios: Personalizar elementos de formulario y mejorar su apariencia.
19. CSS Reset y Normalize: Saber cuándo y por qué usarlos.
20. CSS Grid vs Flexbox: Entender las diferencias y elegir la herramienta adecuada para el trabajo.

JavaScript:

21. Características de ES6+: Usar funciones de flecha, desestructuración, operadores spread/rest y plantillas literales.

22. Manipulación del DOM: Seleccionar elementos, modificar el DOM y manejar eventos.
23. JavaScript Asíncrono: Entender Promesas, async/await y Fetch API.
24. Bucle de Eventos: Explicar cómo funciona el bucle de eventos en JavaScript.
25. Cierres: Entender y usar cierres de manera efectiva.
26. Herencia Prototípica: Explicar cómo funciona la herencia prototípica en JavaScript.
27. Módulos: Usar módulos ES6 con `import` y `export`.
28. Manejo de Errores: Usar bloques try/catch y entender rechazos de promesas sin manejar.
29. Rendimiento de JavaScript: Optimizar el código para un mejor rendimiento.
30. Consola del Navegador: Usar herramientas de desarrollo del navegador para depurar.

Frameworks:

31. React.js: Entender componentes, JSX, estado, props y hooks.
32. Vue.js: Entender instancia de Vue, directivas, componentes y reactividad.
33. Angular: Entender componentes, servicios, inyección de dependencias y enrutamiento.
34. Gestión de Estado: Usar Redux, Vuex o Context API para la gestión de estado.
35. Enrutamiento: Implementar enrutamiento del lado del cliente con React Router, Vue Router, etc.
36. Arquitectura Basada en Componentes: Entender e implementar componentes reutilizables.
37. Métodos de Ciclo de Vida: Conocer métodos de ciclo de vida de React o hooks de Vue.
38. Bibliotecas de Interfaz de Usuario: Usar bibliotecas como Bootstrap, Tailwind o Material-UI.
39. Frameworks de Pruebas: Escribir pruebas con Jest, Jasmine o Cypress.
40. Herramientas de Construcción: Usar Webpack, Babel o Parcel para construir proyectos.

Herramientas y Control de Versiones:

41. Git: Usar git para control de versiones, incluyendo ramificación, fusión y rebase.
42. npm/yarn: Administrar dependencias y scripts del proyecto.
43. package.json: Entender scripts, dependencias y devDependencies.
44. Ejecutores de Tareas: Usar Gulp o Grunt para automatizar tareas.
45. Linting: Usar ESLint o Prettier para la calidad del código.
46. Browsersync: Usar para recarga en vivo durante el desarrollo.

47. Figma/Adobe XD: Entender la entrega de diseño y colaborar con diseñadores.
48. Integración de API: Obtener datos de APIs RESTful o GraphQL.
49. Variables de Entorno: Administrar configuraciones específicas del entorno.
50. Integración Continua: Configurar pipelines CI/CD con GitHub Actions o Jenkins.

Optimización de Rendimiento:

51. División de Código: Implementar división de código con Webpack o importaciones dinámicas.
52. Carga Perezosa: Cargar imágenes, componentes y scripts de manera perezosa.
53. Minificación: Minificar archivos CSS, JavaScript y HTML.
54. Estrategias de Caché: Usar encabezados de caché HTTP y workers de servicio.
55. Optimización de Imágenes: Comprimir y optimizar imágenes para su uso en la web.
56. CSS Crítico: Incrustar CSS crítico para cargas de página más rápidas.
57. Métricas de Rendimiento Web: Entender Lighthouse, GTmetrix y PageSpeed Insights.
58. Carga de Fuentes: Optimizar la carga de fuentes con WebFont Loader o autoalojamiento.
59. Evitar Recursos que Bloquean el Renderizado: Asegurarse de que los scripts y estilos no bloqueen el renderizado.
60. Presupuestos de Rendimiento: Establecer y adherirse a presupuestos de rendimiento.

Accesibilidad:

61. Roles ARIA: Usar roles, estados y propiedades ARIA para una mejor accesibilidad.
62. HTML Semántico: Elegir elementos semánticos para mejorar la accesibilidad.
63. Texto Alternativo para Imágenes: Proporcionar texto alternativo significativo para imágenes.
64. Navegación con Teclado: Asegurarse de que el sitio sea navegable solo con el teclado.
65. Contraste de Color: Usar herramientas para verificar e mejorar el contraste de color.
66. Pruebas con Lectores de Pantalla: Probar con lectores de pantalla como NVDA o VoiceOver.
67. Gestión de Enfoque: Asegurar una gestión de enfoque adecuada en elementos interactivos.
68. Directrices de Accesibilidad: Seguir las directrices WCAG 2.1.
69. Accesibilidad de Formularios: Usar etiquetas, marcadores y validación correctamente.
70. EPub y Cumplimiento AODA: Entender estándares básicos de cumplimiento.

Mejores Prácticas:

71. Organización del Código: Mantener estructuras de código limpias y modulares.
72. Documentación: Escribir documentación clara para componentes y APIs.
73. Pruebas Multi-Navegador: Probar en múltiples navegadores y dispositivos.
74. Mejora Progresiva: Construir sitios que funcionen para todos los usuarios, independientemente del soporte del navegador.
75. Seguridad: Prevenir ataques XSS, usar Content Security Policy y asegurar APIs.
76. Mejores Prácticas de SEO: Optimizar para motores de búsqueda con metaetiquetas, encabezados y texto alternativo.
77. Versionado: Usar versionado semántico para bibliotecas y dependencias.
78. Herramientas de Colaboración: Usar GitHub, GitLab o Bitbucket para la colaboración en equipo.
79. Revisión de Código: Participar en revisiones de código y proporcionar retroalimentación constructiva.
80. Recursos de Aprendizaje: Mantenerse actualizado con MDN, blogs y cursos en línea.

Temas Avanzados:

81. WebSockets: Implementar comunicación en tiempo real con WebSockets.
82. PWA (Aplicaciones Web Progresivas): Entender workers de servicio, soporte fuera de línea y notificaciones push.
83. Canvas y SVG: Crear gráficos con elementos Canvas y SVG.
84. Diseños de CSS Grid y Flexbox: Implementar diseños complejos con CSS Grid y Flexbox.
85. Elementos Personalizados: Crear elementos HTML personalizados con Web Components.
86. Shadow DOM: Entender y usar Shadow DOM para encapsulación.
87. Variables CSS: Usar propiedades personalizadas para temas y estilos dinámicos.
88. Patrones de Diseño en JavaScript: Implementar patrones de diseño como Singleton, Observer y Factory.
89. Internacionalización (i18n): Implementar soporte de idioma y localización.
90. Perfilado de Rendimiento: Usar herramientas como Chrome DevTools para perfilar el rendimiento de JavaScript y DOM.

Habilidades Interdisciplinarias:

91. Experiencia de Usuario (UX): Entender principios de UX y colaborar con diseñadores de UX.

92. Interfaz de Usuario (UI): Crear interfaces visualmente atractivas y amigables para el usuario.
93. Gestión de Proyectos: Usar metodologías ágiles, Scrum o Kanban para la gestión de proyectos.
94. Habilidades de Comunicación: Comunicarse eficazmente con miembros del equipo y partes interesadas.
95. Resolución de Problemas: Abordar problemas de manera metódica y encontrar soluciones óptimas.
96. Adaptabilidad: Aprender y adaptarse rápidamente a nuevas tecnologías y herramientas.
97. Colaboración en Equipo: Trabajar bien en equipo, compartir conocimientos y mentorizar a otros.
98. Gestión del Tiempo: Priorizar tareas y gestionar el tiempo de manera efectiva.
99. Creatividad: Traer soluciones creativas a desafíos de diseño y codificación.
100. Pasión por el Aprendizaje: Mantenerse curioso y mejorar continuamente sus habilidades.