Responsive Web Design:

Q: What is the difference between adaptive design and responsive design?

A: Adaptive design uses predefined layouts based on specific device sizes, while responsive design fluidly adjusts to different screen sizes and orientations.

Q: How do you define breakpoints in responsive design?

A: Breakpoints are defined using media queries in CSS, specifying specific screen widths at which layout changes should occur.

Q: What are some best practices for designing responsive layouts?

A: Best practices include using flexible grids and layouts, prioritizing content for mobile devices, optimizing images and media, and testing across various devices and screen sizes.

Q: How do you ensure that images and videos are responsive?

A: Images and videos can be made responsive by setting their max-width to 100% and using CSS techniques like `object-fit: cover;` for videos to maintain aspect ratio.

Q: What is the role of flexible grids in responsive design?

A: Flexible grids, often created using CSS frameworks like Bootstrap or CSS Grid, allow content to adapt and resize fluidly based on the size of the viewport, enhancing responsiveness.

CSS Reset/Normalize:

Q: What is the purpose of using a CSS reset or normalize?

A: CSS reset or normalize helps to remove browser default styles and inconsistencies, providing a consistent baseline for styling across different browsers.

Q: What are the differences between CSS reset and CSS normalize?

A: CSS reset completely resets all styles to a consistent baseline, while normalize aims to make browser default styles more consistent across different browsers.

Q: How do CSS reset and normalize affect browser default styles?

A: CSS reset removes all default styles, while normalize aims to normalize browser default styles to a consistent baseline across different browsers.

Q: Are there any potential drawbacks to using a CSS reset or normalize?

A: Using a CSS reset can lead to over-resetting and increased specificity issues, while normalize may not completely remove all browser default styles, potentially causing unexpected results.

Q: Can you give an example of a popular CSS reset or normalize library?

A: Popular examples include Normalize.css, which normalizes browser default styles, and Reset CSS, which completely resets all styles to a baseline.

Debugging JavaScript:

Q: What are some common tools used for debugging JavaScript?

A: Common tools include browser developer tools (e.g., Chrome DevTools), VS Code debugger, console methods (e.g., console.log, console.error), and third-party tools like Sentry or Bugsnag.

Q: How can you use console.log() effectively for debugging?

A: console.log() can be used to output variable values, object properties, and debug messages to the browser console, helping identify issues and trace code execution.

Q: What is the difference between an error and an exception in JavaScript?

A: An error is a generic term for any issue that occurs during script execution, while an exception specifically refers to issues that disrupt the normal flow of code execution and can be caught and handled.

Q: How can you handle and debug asynchronous code in JavaScript?

A: Asynchronous code can be debugged using browser developer tools, async/await, or Promise chaining, and handling errors using try...catch blocks or .catch() methods.

Q: What are some best practices for writing debuggable JavaScript code?

A: Best practices include writing modular and well-structured code, using descriptive variable names, commenting code, utilizing error handling techniques, and testing code frequently.