I'm not robot



JavaScript Course for Beginners Overview The course covers two main parts: learning JavaScript as a programming language and working with a browser. It also includes additional thematic articles. Learning from scratch, we go through advanced concepts such as OOP, focusing on the language itself while keeping environment-specific notes minimal. We cover generators, advanced iteration, managing the browser page, including adding elements, manipulating size and position, creating interfaces dynamically, and interacting with visitors. This course teaches JavaScript fundamentals, from variables to functions, objects, and DOM. It's also a versatile language for game development, data visualization, and web application development. Mastering the fundamentals of JavaScript The course aims to help you master JavaScript basics and build a solid foundation in this often-used industry language. JavaScript is powerful and interactive, creating web pages without prior coding knowledge required. With many job openings available, knowing JavaScript can give you an edge in this field. Course Objectives - Master the basics of the JavaScript programming language - Learn to manipulate DOM using JavaScript - Create functions and understand object-oriented programming - Understand JavaScript interaction with HTML and CSS - Build professional JavaScript code - Control browser cookies - Debug code and increase performance - Learn about hoisting, strict mode, window objects, form validation, submission, exceptions, and more. Requirements Basic knowledge of HTML and CSS is necessary. No prior expertise required, making it suitable for both beginners and working professionals. JavaScript is a highly sought-after programming language, making it an ideal choice for programmers worldwide. Once you learn JavaScript, you can develop high-quality front-end and back-end software using popular frameworks like jQuery and Node.JS. With widespread support from modern web browsers, you don't need special setup to get started. You can create visually stunning and fast websites with a console-like look and feel, providing an excellent Graphical User Experience. The versatility of JavaScript has expanded its applications to mobile app development, desktop app development, game development, and more. This opens up numerous job opportunities with high pay and growth prospects. Additionally, the abundance of existing frameworks and libraries makes software development faster and more efficient. Many prominent companies like Google, Meta, Microsoft, PayPal, and LinkedIn use JavaScript, and its extensive community provides support to learners. JavaScript is a versatile programming language used in various applications such as web development, game development, machine learning, and more. It can be used to build fast and scalable network applications with Node JS, create sophisticated server applications including Web Servers, and develop games using multiple libraries and NPM packages. Additionally, JavaScript is used in mobile application development with frameworks like React Native and for data visualization with D3.js. It can also be used in cloud computing platforms such as Cloudflare and AWS lambda to write and deploy functions on the cloud. A basic JavaScript example can print "Getting started with JavaScript tutorial" on a webpage, while an online editor allows users to edit and execute code directly from their browser. Jobs in the world, over 15 million JavaScript developers are actively working, yet there's still a shortage of skilled professionals. As a result, it's an excellent opportunity for you to start your career as a JavaScript developer. Here are some top companies that offer JavaScript developer roles: Amazon, Google, Microsoft, Apple, Adobe, Facebook, and PayPal, among many others. You can pursue various career paths after learning JavaScript, including front-end, back-end, full-stack, web, game, mobile app development, or even DevOps engineering. Some popular frameworks to use for website, app, or backend development are React, Angular, Vue.js, Ember.js, and more. Libraries like jQuery, Axios, Chart.js, D3.js, Socket.io, Underscore.js, Lodash, Three.js, can help you streamline your code. To boost your skills and career, consider getting certified in JavaScript. Practice your skills with interactive coding challenges and explore articles on the latest JavaScript trends. Frequently asked questions about JavaScript include: what is JavaScript, its features, advantages, and more. To check if users have missed something. Improved interactivity − Websites can respond to mouse hovers or keyboard input with interactive interfaces. More engaging interfaces − JavaScript enables drag-and-drop elements, sliders, and other rich interface features. However, there are some drawbacks to using JavaScript. It's not a full-fledged programming language due to lacking key features. Client-side JavaScript has file reading and writing restrictions for security reasons. Networking applications are also limited as JavaScript doesn't support networking functions. Additionally, it lacks multithreading or multiprocess capabilities. JavaScript is case-sensitive, meaning keywords, variables, function names, and identifiers must be typed consistently. For learning JavaScript, a simple tutorial can be helpful, but it's essential to note that the tutorial actually refers to Java programming concepts, not JavaScript. Some popular backend JavaScript frameworks include Express, Next, Gatsby, Meteor, and others. A JavaScript function is defined with the 'function' keyword followed by a name in parentheses. The code inside curly brackets will execute when the function is invoked, passing parameters and receiving arguments. Functions often return values, which are passed back to the caller. For instance, a 'myFunction' calculates the product of two numbers and returns the result: let x = myFunction(4, 3); function myFunction(a, b) { return a * b; } Reusing Code: A Sneak Peek into Functions You can write code that gets reused multiple times, using various arguments to produce diverse results. The () Operator allows you to invoke functions by calling them with parentheses. For instance, to convert Fahrenheit to Celsius, you'd use the following function: `function toCelsius(fahrenheit) { return (5/9) * (fahrenheit-32); }` Then, when you call it like this: `toCelsius(77)`, it returns the result. But if you accidentally pass no arguments or just the function name without calling it, it'll either give an incorrect answer or return the function itself, not the actual outcome. Think of `toCelsius` as a function object when used alone, and `toCelsius()` as the actual calculation when called with parentheses. Functions in Formulas & Assignments You can even use functions directly within calculations just like regular variables - no need to store their results beforehand! For example: `let text = "The temperature is " + toCelsius(77) + " Celsius";` This way, you get the same result without the extra step of storing it in a variable. Note that this technique will be explored further later on in the tutorial. Local Variables & Scope Inside functions, variables declared using `let`, like `carName`, are only accessible within that specific function. They don't exist outside their defined scope, so you can use the same names for variables across different functions without worrying about conflicts. These local variables come into existence when a function starts and disappear once it's completed.