DWA_02.8 Knowledge Check_DWA2

---

1. What do ES5, ES6 and ES2015 mean - and what are the differences between them?

ES5 is a version of the ECMAScript programming language specification. It introduced some new features and improvements over its predecessor, ES3. The key features of ES5 include strict mode for writing more reliable code, accessors for defining custom behavior for properties, array methods for manipulating arrays easily, JSON object for working with JSON data, object enhancements for better object manipulation, and function enhancements like the `bind` method for controlling the value of `this` within a function.

ES6 is a version of the ECMAScript programming language specification. It introduced significant enhancements and new features compared to its predecessor, ES5. Some key features of ES6 include arrow functions for concise function syntax, block-scoped variables with `let` and `const`, classes for object-oriented programming, modules for organizing code, template literals for creating multi-line strings, enhanced object literals, native promises for handling asynchronous operations, and destructuring assignment for extracting values from arrays or objects.

ES2015 is another name for ECMAScript 2015, which is the same as ES6. The year-based naming convention was adopted to align the ECMAScript specification with the year it was finalized. ES2015 (or ES6) introduced significant enhancements and new features to the ECMAScript programming language. These features include arrow functions, block-scoped variables (`let` and `const`), classes, modules, template literals, enhanced object literals, promises, and destructuring assignment, among others. The use of ES2015 or ES6 is interchangeable, referring to the same version of the ECMAScript specification.

The difference between all the above is that ES5 improved upon ES3 with various enhancements. ES6 (or ES2015) was a major release that introduced modern syntax and a wide range of new features and capabilities. ES2015 and ES6 refer to the same version, while ES5 came before it.

---

2. What are JScript, ActionScript and ECMAScript - and how do they relate to JavaScript?

JScript, ActionScript, and ECMAScript are all scripting languages used for web development. Here's a brief overview of each:

JScript: JScript is a scripting language developed by Microsoft. It is a dialect of ECMAScript and is largely compatible with JavaScript. JScript was primarily used in Microsoft's Internet Explorer web browser and Windows scripting environments.

ActionScript: ActionScript is a scripting language primarily associated with Adobe Flash. It is based on ECMAScript and was used to create interactive and multimedia-rich content for websites and applications using Adobe Flash Player. With the decline of Flash, ActionScript has become less prevalent.

ECMAScript: ECMAScript is a standardized scripting language specification. It defines the syntax, semantics, and behavior of scripting languages such as JavaScript, JScript, and ActionScript. JavaScript is the most widely used implementation of ECMAScript and is commonly used for client-side web development. ECMAScript serves as a standard for scripting languages, providing guidelines and specifications that implementations should follow.

JScript is a Microsoft-specific version of ECMAScript, ActionScript is a scripting language used in Adobe Flash, and ECMAScript is the standardized specification that defines the syntax and behavior of scripting languages like JavaScript.

Relation between them is that JScript and ActionScript are specific implementations of the ECMAScript specification developed by Microsoft and Adobe, respectively. JavaScript is the most commonly used implementation and is widely supported in web browsers. All three languages are based on the ECMAScript standard and share many similarities in syntax and behavior, although there may be some platform-specific differences.

---

3. What is an example of a JavaScript specification - and where can you find it?

An example of a JavaScript specification is the ECMAScript 2021 (ES12) specification. It defines the syntax, semantics, and behavior of the JavaScript programming language.

The ECMAScript specifications are maintained by Ecma International, a standards organization. You can find the latest ECMAScript specification, including ECMAScript 2021 and any future updates, on the Ecma International website at the following URL:

https://www.ecma-international.org/publications-and-standards/standards/ecma-262/

On this page, you can access the ECMAScript standard in PDF format or explore the different versions and editions of the specification. The specification provides a comprehensive guide to the JavaScript language, describing its syntax, data types, operators, control structures, built-in objects, and more. It serves as a reference for understanding the JavaScript language in depth.

---

4. What are v8, SpiderMonkey, Chakra and Tamarin? Do they run JavaScript differently?

V8: V8 is a high-performance JavaScript engine developed by Google. It powers the Google Chrome web browser and is also used in other projects like Node.js. V8 is written in C++ and is known for its speed and efficient memory management. It employs various optimization techniques, such as just-in-time (JIT) compilation, to enhance JavaScript execution performance.

SpiderMonkey: SpiderMonkey is the JavaScript engine developed by Mozilla, the organization behind the Firefox web browser. It was the first-ever JavaScript engine and is written in C++. SpiderMonkey supports various Mozilla projects and has been influential in advancing JavaScript standards and features.

Chakra: Chakra is the JavaScript engine developed by Microsoft. It was initially used in the Internet Explorer web browser and later in Microsoft Edge. Chakra was designed to

optimize JavaScript execution, focusing on fast startup times and efficient memory management. With the introduction of Microsoft Edge based on Chromium, Chakra has been replaced with the V8 engine.

Tamarin: Tamarin was a JavaScript engine created by Adobe Systems. It was specifically developed for the Adobe Flash Player to provide improved performance and support for ActionScript 3.0, which is a programming language used in Flash applications. Tamarin incorporated a just-in-time (JIT) compiler and other optimizations to enhance the execution speed of JavaScript and ActionScript code. However, Adobe discontinued active development of Tamarin in 2012.

These JavaScript engines play a crucial role in executing JavaScript code efficiently and provide the backbone for web browsers and other platforms that rely on JavaScript-based applications.

---

5. Show a practical example using caniuse.com and the MDN compatibility table.