DWA_02.8 Knowledge Check_DWA2

---

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

ES5, ES6, and ES2015 refer to different versions of the ECMAScript language specification, which is the standard underlying JavaScript. Here's a detailed explanation of each term and the differences between them:

ES5 (ECMAScript 5)

Released: December 2009
Significance: ES5 is an important update to the language that introduced several new features and improvements aimed at enhancing the capabilities of JavaScript and making it easier to develop robust applications.

Key Features:
- "Strict Mode": A way to opt into a restricted variant of JavaScript, which catches common coding errors and "unsafe" actions such as using undeclared variables.
- New Object Methods: Methods like Object.create, Object.defineProperty, Object.defineProperties, Object.keys, Object.freeze, Object.seal, and Object.preventExtensions.
- Array Methods: New array iteration methods like forEach, map, filter, reduce, reduceRight, some, and every.
- JSON Support: Native JSON parsing and stringifying using JSON.parse and JSON.stringify.
- Property Attributes: More control over property definitions with configurable, enumerable, and writable attributes

ES6 (ECMAScript 6) / ES2015

- Released: June 2015
- Significance: ES6/ES2015 is one of the biggest updates to the ECMAScript language, introducing major features that modernized JavaScript development. It is often seen as a

turning point for JavaScript, adding syntax and constructs that greatly enhance developer productivity.

Key Features:

- Arrow Functions: Shorter syntax for writing function expressions.
- Classes: A new syntax for creating objects and dealing with inheritance, making JavaScript's prototype-based inheritance easier to use.
- Template Literals: Multi-line strings and string interpolation using backticks (`).
- Destructuring: Syntax for extracting values from arrays or properties from objects into distinct variables.
- Modules: Native support for modular JavaScript code, using import and export statements.
- Promises: A new way to handle asynchronous operations, improving the management of async code.
- Let and Const: New ways to declare variables, with block scope (for let) and constants (for const).
- Default Parameters: Allowing function parameters to have default values.
- Rest and Spread Operators: Simplifying work with arrays and function arguments.
- Iterators and Generators: New constructs to work with data sequences and implement custom iteration behaviors.
- Symbols: A new primitive type for creating unique identifiers.

ES2015

- Explanation: ES2015 is simply another name for ES6. The naming convention was changed to reflect the year of the release. Hence, ES2015 and ES6 are interchangeable terms and refer to the same version of the ECMAScript specification.
- Context: The renaming was part of an effort to adopt a more predictable and regular update cycle for ECMAScript, with new versions released annually. Therefore, subsequent versions were named ES2016, ES2017, and so on.

Differences Between ES5 and ES6/ES2015

- Syntax Enhancements: ES6 introduced significant syntax improvements (like arrow functions, classes, and template literals) that make the code more concise and readable.
- Modules: ES5 lacked native module support, whereas ES6 introduced a standardized module system.
- Variable Declarations: ES6 introduced let and const for better variable scoping, compared to var in ES5.
- Asynchronous Programming: ES6 added Promises, which simplified working with asynchronous code compared to the callback approach in ES5.
- New Data Structures and APIs: ES6 introduced new data structures like Maps and Sets, as well as new built-in methods and properties.

---

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

ECMAScript

- Definition: ECMAScript is the standardized specification for scripting languages, officially defined by Ecma International in the ECMA-262 standard.
- Significance: ECMAScript serves as the foundation for several scripting languages, including JavaScript, JScript, and ActionScript. It provides a consistent core language syntax and features that these languages implement and build upon.
- Versions: The specification has evolved through various versions (ES1, ES2, ES3, ES5, ES6/ES2015, ES2016, etc.), each introducing new features and improvements.

JavaScript

- Definition: JavaScript is a high-level, interpreted scripting language primarily used for web development. It is implemented in web browsers to create dynamic and interactive web pages.
- Relation to ECMAScript: JavaScript is the most well-known and widely used implementation of the ECMAScript specification. It adheres to the ECMAScript standards, ensuring compatibility and consistency across different environments and browsers.
- History: JavaScript was created by Brendan Eich at Netscape and was first released in 1995. It was originally called Mocha, then LiveScript, and finally JavaScript.

JScript

- Definition: JScript is Microsoft's implementation of the ECMAScript standard.
- Usage: JScript was primarily used in Microsoft's Internet Explorer and other Windows-based environments. It enabled web developers to write scripts for web pages and automate tasks within Windows applications.

- Relation to ECMAScript: JScript conforms to the ECMAScript standard, ensuring compatibility with other ECMAScript-compliant languages. However, it also included some proprietary extensions specific to Microsoft platforms.
- History: JScript was first introduced in 1996 with Internet Explorer 3.0.

ActionScript

- Definition: ActionScript is a scripting language used primarily for developing applications and animations on the Adobe Flash platform.
- Usage: ActionScript was widely used in the early 2000s for creating interactive content, games, and rich internet applications (RIAs) within Flash.
- Relation to ECMAScript: ActionScript is based on the ECMAScript standard, but it includes additional features and libraries specific to the Flash runtime environment.
- Versions: There have been several versions of ActionScript, with ActionScript 3.0 being a significant update that introduced a more robust and structured programming model.
- History: ActionScript was developed by Macromedia (later acquired by Adobe) and was first introduced with Flash 4 in 1999. It evolved significantly with Flash MX 2004 and later versions.

Summary of Relationships

- ECMAScript: The standardized specification that provides the foundation for several scripting languages.

- JavaScript: The most popular and widely-used implementation of ECMAScript, primarily used in web browsers.

- JScript: Microsoft's implementation of ECMAScript, used in Internet Explorer and Windows environments.

- ActionScript: An implementation of ECMAScript tailored for the Adobe Flash platform, used for creating interactive multimedia content.

---

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

An example of a JavaScript specification is ECMAScript 2015 (ES6). This specification is formally known as ECMA-262, 6th Edition. It introduced many new features and improvements to the language, such as let and const for variable declarations, arrow functions, classes, modules, template literals, promises, and much more.

- You can find the official ECMAScript specifications on the Ecma International website. Here are the steps to access it:

- **1.Visit the Ecma International Website:**

  - Go to the Ecma International homepage: [Ecma International](https://www.ecma-international.org/).

- 2. **Navigate to the ECMAScript Section:**

  - On the Ecma International homepage, look for the "Standards" section, and from there, navigate to the "ECMA-262" specification, which is the standard for ECMAScript.

- 3. Access Specific Editions:
  - You can find the specific editions of the ECMAScript standard by clicking on the desired version. For example, for ES6 (ECMAScript 2015), you can directly access it via this link: [ECMAScript 2015 (6th Edition) Specification](https://www.ecma-international.org/ecma-262/6.0/).

Here is a brief summary of how to navigate to a specific ECMAScript specification like ES6:

- Home Page: [Ecma International](https://www.ecma-international.org/)
- Direct Link to ES6:[ECMAScript 2015 (6th Edition) Specification](https://www.ecma-international.org/ecma-262/6.0/)

---

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

V8, SpiderMonkey, Chakra, and Tamarin are all JavaScript engines, but they are developed by different organizations and used in different contexts. Here's a detailed look at each one, including how they run JavaScript and any differences in their approaches:

V8
- Developer Google

- **Usage**: Primarily used in Google Chrome and Node.js.

- **Key Features**:
  - **Performance** :V8 is known for its high performance, utilizing just-in-time (JIT) compilation to convert JavaScript into machine code before execution.
  - **Optimizatio**n: Employs various optimization techniques such as inline caching, hidden classes, and an optimizing compiler called TurboFan.
  - **Memory Management**: Implements efficient garbage collection to manage memory.
  - **Compatibility**: Continuously updated to comply with the latest ECMAScript standards.
- **Running JavaScript**:V8 translates JavaScript into more efficient machine code, which can run directly on the CPU, providing fast execution speeds.

SpiderMonkey

- **Develope**r: Mozilla
- **Usage**:Used in Mozilla Firefox and as a basis for other projects like the GNOME desktop environment's GJS.
- **Key Features**:
  - **JIT Compilation**: Like V8, SpiderMonkey uses JIT compilation to improve performance.
  - nterpreted and Compiled Code: Can interpret JavaScript and also compile it to machine code for faster execution.
  - **Garbage Collection**: Employs an advanced garbage collection mechanism to manage memory.
  - **Compatibility**: Regularly updated to support new ECMAScript features.
- **Running JavaScript**: Combines interpreting and JIT compilation to run JavaScript efficiently, with ongoing performance improvements and compliance with ECMAScript standards.

Chakra

-**Develope**r:Microsoft
- **Usage**:Originally used in the Microsoft Edge browser (Legacy version) and in Node.js via ChakraCore.
- **Key Features:**

- **JIT Compilation:**Uses JIT compilation to speed up JavaScript execution.
- **Background Optimization:**Optimizes code in the background to enhance performance without blocking the main execution thread.
- **Garbage Collection:** Includes an efficient garbage collector for memory management.
- **Compatibility:** Supports ECMAScript standards, with particular focus on enterprise-level applications and performance.

- **Running JavaScript:** Chakra compiles JavaScript to machine code using JIT techniques, optimizing for both startup time and execution speed.

Tamarin

- **Developer:** Adobe
- **Usage:**Originally used in the Adobe Flash Player for executing ActionScript, which is based on ECMAScript.
- **Key Features:**
  - **ActionScript Execution**: Specifically designed to execute ActionScript 3, but also capable of running ECMAScript.
  - **JIT Compilation**: Uses JIT compilation to improve the performance of ActionScript and ECMAScript code.
  - **Garbage Collection**: Implements garbage collection to manage memory efficiently.
  - **Integration:**Closely integrated with the Flash runtime for enhanced multimedia capabilities.
- **Running JavaScript:** Although primarily focused on ActionScript, Tamarin can run JavaScript, using JIT compilation and other techniques to optimize execution.

Differences in Running JavaScript

While all these engines run JavaScript, they have differences in their approaches and optimizations:

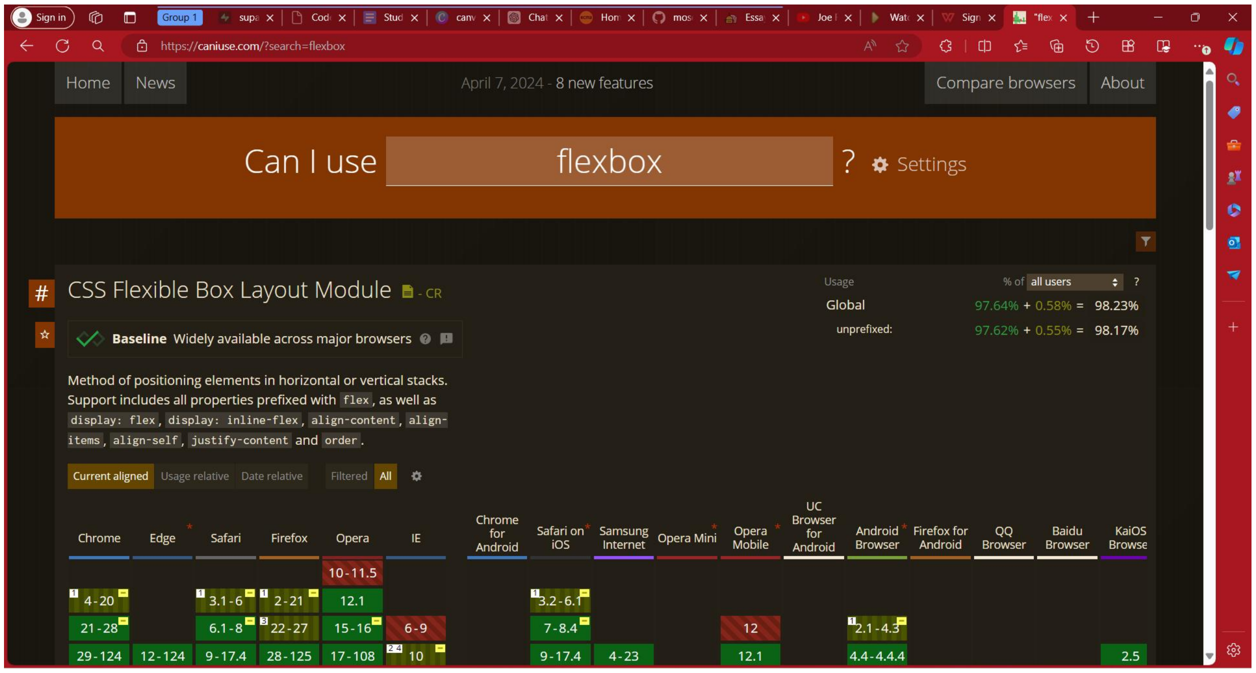
- **V8** and **SpiderMonkey** both emphasize high performance with aggressive JIT compilation and advanced optimization techniques. V8 is particularly noted for its fast execution speeds, while **SpiderMonkey** is recognized for its compliance and integration with Firefox.
- **Chakra** focuses on balancing performance with smooth execution, using background optimization and supporting enterprise applications. It has been open-sourced as ChakraCore.
- **Tamarinis** unique in its dual focus on **ActionScript** and ECMAScript, primarily used within the Flash ecosystem. It employs similar JIT compilation techniques but is tailored for multimedia applications.

Each engine's design reflects the priorities of its developers and the specific needs of their primary environments, resulting in varied implementations of the ECMAScript standard.

---

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

Bellow I am searching the compatibility of browsers that can support the css Flexbox feature for future projects


https://caniuse.com/?search=flexbox

Home News April 7, 2024 - 8 new features Compare browsers About

Can I use flexbox ? Settings

CSS Flexible Box Layout Module - CR

Baseline Widely available across major browsers

Usage % of all users ?

Global	97.64%	+	0.58%	=	98.23%
unprefixed:	97.62%	+	0.55%	=	98.17%

Method of positioning elements in horizontal or vertical stacks. Support includes all properties prefixed with `flex`, as well as `display: flex`, `display: inline-flex`, `align-content`, `align-items`, `align-self`, `justify-content` and `order`.

Current aligned Usage relative Date relative Filtered All

Chrome	Edge	Safari	Firefox	Opera	IE	Chrome for Android	Safari on iOS	Samsung Internet	Opera Mini	Opera Mobile	UC Browser for Android	Android Browser	Firefox for Android	QQ Browser	Baidu Browser	KaiOS Browse
				10-11.5												
4-20		3.1-6	2-21	12.1			3.2-6.1									
21-28		6.1-8	22-27	15-16	6-9		7-8.4			12		2.1-4.3				
29-124	12-124	9-17.4	28-125	17-108	10		9-17.4	4-23		12.1		4.4-4.4.4				2.5