DWA_02.8 Knowledge Check_DWA2

---

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

ES5 also known as ES2009 is the fifth edition of ECMAScript (JavaScript). ES6 also known as ES2015 is the sixth edition of ECMAScript (JavaScript).

- ES5 supports primitive data types (string, null, number, boolean and undefined) while ES6 brought us a new data type symbol for supporting unique values.
- When we want to define variables, ES5 only uses “var” and ES6 introduced “const” & “let”.
- ES5 is more time consuming while ES6 processes destructuring, speed operations & object manipulation more smoothly.
- In ES5 to define a function you use keywords “function” & “return” but with ES6 arrow functions where introduced by which we don’t need to use the keyword “function” to define a function.
- ES5 uses the “for” loop to iterate over elements. ES6 introduced “for ... .of” loops to iterate over the values of the iterable object.

---

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

JScript was made by Microsoft for the browser “Internet Explorer” and is implemented as an active scripting engine. ActionScript is an object-oriented programming language which was made by Macromedia (later known as Adobe), ActionScript is usually used for websites. ECMAScript is JScript & JavaScript. It’s a standards organization for communication & information systems.
JavaScript was the first programming language made to help run Netscape Navigator. When Netscape began to be more popular, Mirosoft decided to make a competitor named Internet Explorer. For internet explorer to work, Mirosoft decided to reverse engineer JavaScript and made JScript. Internet Explorer became very popular with a peak of 95% of the market share, but over time JavaScripts browsers grew larger. After some time developers started demanding some over-arching standardisation amongst browsers, that is when ECMA stepped in and made standardized specifications. Eventually ECMA made a proposal for specification and became ECMAScript. To this day we use JavaScript instead of the name ECMAScript.

---

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

Reference is a resolved name binding that consists of the “base” value, the “referenced name” and the Boolean value “strict reference flag”. There is different base values:

- Undefined (this value indicates that the reference was not set on a binding)
- Object
- Boolean
- String
- Number

“The following abstract operations are used in this specification to access the components of references:

- GetBase(V). Returns the base value component of the reference V.
- GetReferencedName(V). Returns the referenced name component of the reference V.
- IsStrictReference(V). Returns the strict reference component of the reference V.
- HasPrimitiveBase(V). Returns true if the base value is a Boolean, String, or Number.
- IsPropertyReference(V). Returns true if either the base value is an object or HasPrimitiveBase(V) is true; otherwise returns false.
- IsUnresolvableReference(V). Returns true if the base value is undefined and false otherwise.”

Copyright © 2011 Ecma International

Ecma International
Rue du Rhone 114
CH-1204 Geneva
Tel: +41 22 849 6000
Fax: +41 22 849 6001

Web: http://www.ecma-international.org

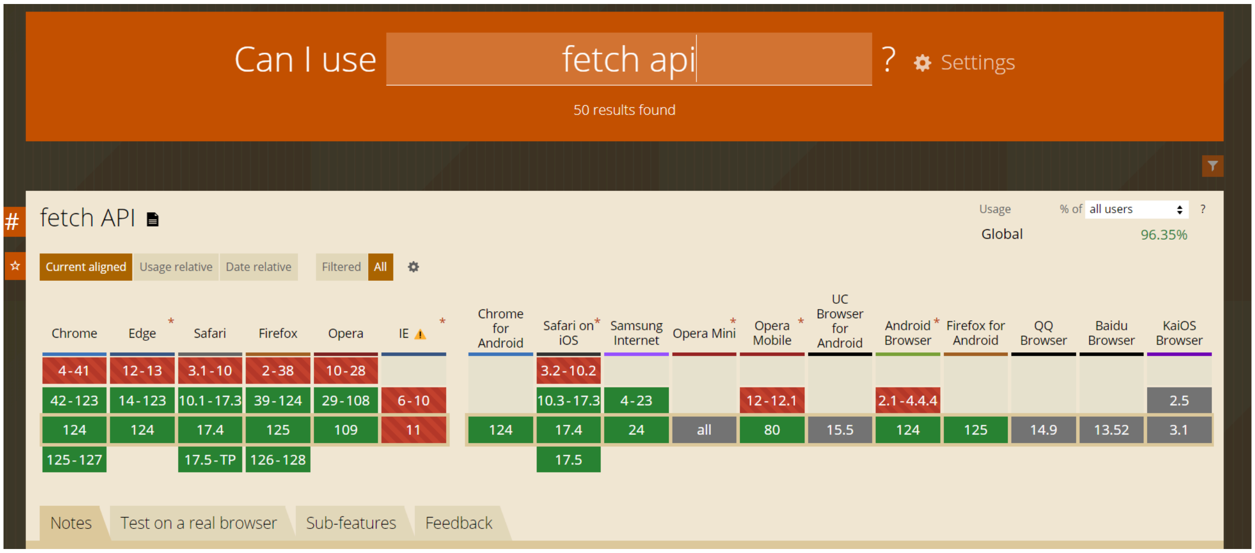
---

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

V8, SpiderMonkey, Chakra and Tamarin are all JavaScript compilers. Compilers are all programs to translate code from one language to another.
They do run JavaScript differently, because they change your code into another language so the computer can understand better. An example is, your compiler can change the code to 0s & 1s, which the computer reads very fast.

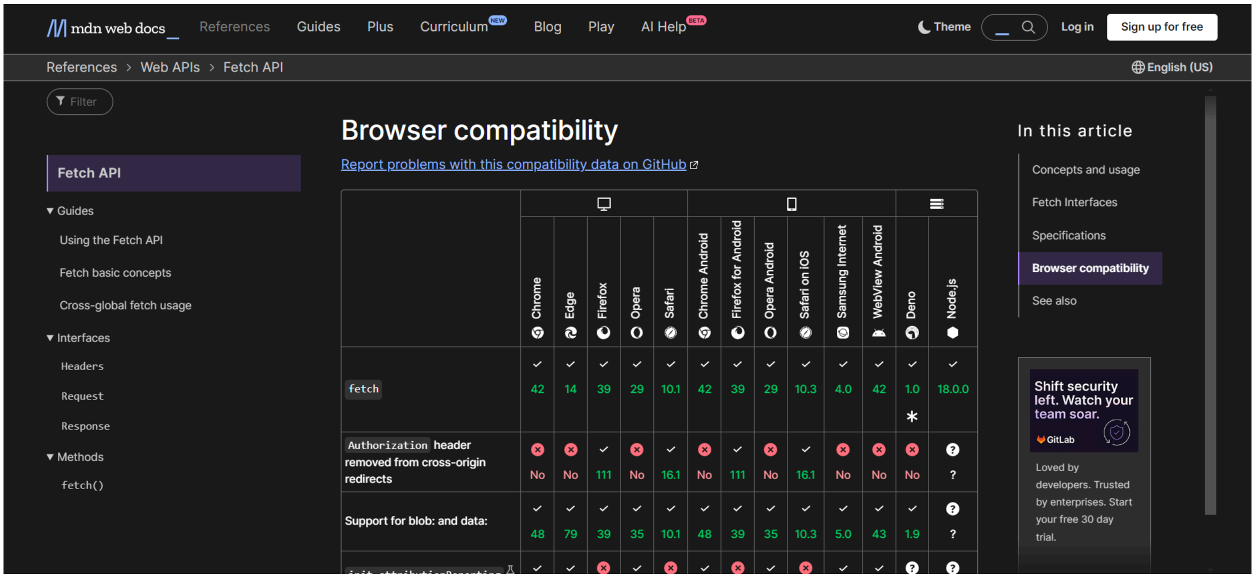
---

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



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 Browser
4-41	12-13	3.1-10	2-38	10-28			3.2-10.2									
42-123	14-123	10.1-17.3	39-124	29-108	6-10		10.3-17.3	4-23		12-12.1		2.1-4.4.4				2.5
124	124	17.4	125	109	11	124	17.4	24	all	80	15.5	124	125	14.9	13.52	3.1
125-127		17.5-TP	126-128				17.5									

https://caniuse.com/?search=fetch%20api



References › Web APIs › Fetch API

English (US)

Fetch API

- Guides
  - Using the Fetch API
  - Fetch basic concepts
  - Cross-global fetch usage
- Interfaces
  - `Headers`
  - `Request`
  - `Response`
- Methods
  - `fetch()`

Browser compatibility

Report problems with this compatibility data on GitHub

	Chrome	Edge	Firefox	Opera	Safari	Chrome Android	Firefox for Android	Opera Android	Safari on iOS	Samsung Internet	WebView Android	Deno	Node.js
`fetch`	✓ 42	✓ 14	✓ 39	✓ 29	✓ 10.1	✓ 42	✓ 39	✓ 29	✓ 10.3	✓ 4.0	✓ 42	✓ 1.0 *	✓ 18.0.0
`Authorization` header removed from cross-origin redirects	✗ No	✗ No	✓ 111	✗ No	✓ 16.1	✗ No	✓ 111	✗ No	✓ 16.1	✗ No	✗ No	✗ No	? ?
Support for blob: and data:	✓ 48	✓ 79	✓ 39	✓ 35	✓ 10.1	✓ 48	✓ 39	✓ 35	✓ 10.3	✓ 5.0	✓ 43	✓ 1.9	? ?
`init.attributionReporting`	✓	✓	✗	✓	✗	✓	✗	✓	✗	✓	✓	?	?

In this article

- Concepts and usage
- Fetch Interfaces
- Specifications
- Browser compatibility
- See also

https://developer.mozilla.org/en-US/docs/Web/API/Fetch_API