ES6 Tutorial



ES6 tutorial provides you the basic and advanced concepts.

ES6 or ECMAScript 6 is a scripting language specification which is standardized by ECMAScript International. This specification governs some languages such as [JavaScript](https://www.javatpoint.com/javascript-tutorial), ActionScript, and Jscript. ECMAScript is generally used for client-side scripting, and it is also used for writing server applications and services by using [Node.js](https://www.javatpoint.com/nodejs-tutorial).

ES6 allows you to write the code in such a way that makes your code more modern and readable. By using ES6 features, we write less and do more, so the term 'Write less, do more' suits ES6.

This tutorial introduces you to the implementation of ES6 in JavaScript.

What is ES6?

ES6 is an acronym of **ECMAScript 6** and also known as ECMAScript 2015.

ES6 or **ECMAScript6** is a scripting language specification which is standardized by ECMAScript International. It is used by the applications to enable client-side scripting. This specification is affected by programming languages like Self, [Perl](https://www.javatpoint.com/perl-tutorial), [Python](https://www.javatpoint.com/python-tutorial), [Java](https://www.javatpoint.com/java-tutorial), etc. This specification governs some languages such as JavaScript, ActionScript, and Jscript. ECMAScript is generally used for client-side scripting, and it is also used for writing server applications and services by using Node.js.

ES6 allows you to make the code more modern and readable. By using ES6 features, we write less and do more, so the term 'Write less, do more' suits ES6. ES6 introduces you many great features such as scope variable, arrow functions, template strings, class destructions, modules, etc.

ES6 was created to standardize JavaScript to help several independent implementations. Since the standard was first published, JavaScript has remained the well-known implementation of ECMAScript, comparison to other most famous implementations such as **Jscript** and **ActionScript**.

History

The ECMAScript specification is the standardized specification of scripting language, which is developed by **Brendan Eich (**He is an American technologist and the creator of JavaScript programming language**)**of**Netscape (**It is a name of brand which is associated with Netscape web browser's development**).**

Initially, the ECMAScript was named **Mocha,**later **LiveScript,**and finally, **JavaScript.**In December 1995, **Sun Microsystems (**an American company that sold the computers and its components, software, and [IT](https://www.javatpoint.com/it-full-forms) services. It created Java, NFS, ZFS, SPARC, etc.**)**and **Netscape**announced the JavaScript during a press release.

During November 1996, Netscape announced a meeting of the ECMA International standard organization to enhance the standardization of JavaScript.

ECMA General Assembly adopted the first edition of ECMA-262 in **June 1997**. Since then, there are several editions of the language standard have published. The Name **'ECMAScript'** was a settlement between the organizations which included the standardizing of the language, especially Netscape and Microsoft, whose disputes dominated the primary standard sessions. **Brendan Eich**commented that 'ECMAScript was always an unwanted trade name which sounds like a skin disease (eczema).'

Both JavaScript and Jscript aims to be compatible with the ECMAScript, and they also provide some of the additional features that are not described in ECMA specification.

ES6 versions

There are ten editions of ECMA-262 published. The work on version 10 was finalized in June 2019. The ten editions of ECMA-262 are listed as follows:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Edition** | **Date Published** | **Name** | **Changes from the prior edition** | **Editor** |
| 1. | June 1997 |  | First Edition | Guy L. Steele Jr. |
| 2. | June 1998 |  | Editorial changes for keeping the specification fully aligned with the ISO/IEC 16262 international standard. | Mike Cowlishaw |
| 3. | December 1999 |  | Addition of Regular expressions, new control statements, better string handling, tighter error definitions, numeric output formatting, try/catch exception handling, and other enhancements. | Mike Cowlishaw |
| 4. | Abandoned |  | The fourth edition was abandoned because of the political differences concerning language complexity. Several features proposed for this edition have been dropped completely. |  |
| 5. | December 2009 |  | Addition of 'strict mode,' a subset which is intended to give more by error checking and avoid error-prone constructs. It clarifies several ambiguities in the 3rd edition specifications, and it also accommodates the behavior of the real-world implementations, which consistently differed from that specification. Addition of some new features, like getters and setters, library support for JSON. | Pratap Lakshman, Allen Wirfs-Brock |
| 5.1 | June 2011 |  | This 5.1 edition of the ECMAScript standard is fully aligned with the third edition of the ISO/IEC 16262:2011. | Pratap Lakshman, Allen Wirfs-Brock |
| 6. | June 2015 | ECMAScript 2015 (ES 2015) | Addition of new syntax for writing applications, including the class declarations, ES6 modules, but defines them in the same terms as ECMAScript5 strict mode. Other features of this update include iterators, arrow function expression, Python-style generators, binary data, typed arrays, new collections, etc. | Allen Wirfs-Brock |
| 7. | June 2016 | ECMAScript 2016 (ES 2016) | The major standard language includes features such as destructing patterns of variables, block-scoping of functions and variables, proper tail calls, exponentiation operator for numbers, etc. | Brian Terison |
| 8. | June 2017 | ECMAScript 2017 (ES 2017) | It includes async/await constructions, which works by using generators and promises. It also contains the features for concurrency and atomics, syntactic integration with promises (async/await). | Brian Terison |
| 9. | June 2018 | ECMAScript 2018 (ES 2018) | Its new features include rest/spread operators for variables, asynchronous iteration. | Brian Terison |
| 10. | June 2019 | ECMAScript 2019 (ES 2019) | Addition of new features include Array.prototype.flatMap, Array.prototype.flat, and changes to Array.sort and Object.fromEntries. | Brian Terison, Bradley Farias, Jordan Harband |

During June 2004, ECMA International published the ECMA-357 standard for defining an extension to ECMAScript, which is known as **ECMAScript for XML**.

Let us try to elaborate on the versions of ECMAScript.

4th Edition (abandoned)

This fourth edition of ECMA-262 (**ECMAScript 4**or**ES4**) was the first update to ECMAScript since the third edition was published in 1999. This specification was targeted to completion by October 2008.

By August 2008, the fourth edition of ECMAScript had been scaled back into the project code-named **ECMAScript Harmony.**Features within the discussion for the harmony at the time included the classes, destructing assignment, a module system, optional type annotations, and static typing, etc.

In addition, to add new features, some of the ES3 bugs were planned to be fixed in Edition 4. These fixes and others, and also the support for the JSON encoding/decoding, were folded within the ECMAScript fifth edition specification.

In late 2007, a debate between **Eich (**later the CTO of **Mozilla Foundation)**and **Chris Wilson (Microsoft's**platform architect for the Internet explorer**)**became public on many blogs.

This fourth edition was abandoned because of the political differences concerning language complexity. Several features proposed for this edition have been dropped completely.

5th Edition

Yahoo, Microsoft, Google, and other objectors of the fourth edition were formed their subcommittee for designing a less ambitious update of ECMAScript 3 named ECMAScript 3.1. This edition will focus on the library and security updates, with a large emphasis on compatibility. After the public debate, the ECMAScript 3.1 and ECMAScript 4 teams get agreed on a compromise.

After some time, **Brendan Eich** announced that the ECMA TC39 would focus on the work of ECMAScript 3.1 (later ECMAScript, fifth edition) along with the complete collaboration of all parties, and vendors will target minimum two implementations by 2009. In April 2009, the **ECMA TC39**has published the draft related to 5th edition and also announced that testing of the interoperable implementations could be completed by mid of July. But, on December 3, 2009, the fifth edition of ECMA-262 was published.

6th Edition - ECMAScript 2015

Initially, this edition was termed as **ECMAScript 6** or **ES6** and then later renamed to **ECMAScript 2015,**which was finalized in June 2015. This update adds the new essential syntax for writing the complex applications, including the declarations of classes such as **(class Example { ... })**, ES6 modules like **import \* as moduleName from "..."; export const Example,**but defines them in the same order as ECMAScript 5 strict mode. It also includes some other new features such as Python-style generators, **let keyword**for local declarations, arrow function expression, the **const**keyword for the constant variable declarations, binary data, typed arrays, new collections, reflection, number and math enhancements and many more.

7th Edition - ECMAScript 2016

The 7th edition was officially known as **ECMAScript 2016,**which was finalized in June 2016. The standard language includes features such as block scoping of functions and variables, destructing the patterns of variables, proper tail calls, **async/await**keywords for asynchronous programming, exponentiation operator **\*\*** for numbers.

8th Edition - ECMAScript 2017

The 8th edition was officially known as **ECMAScript 2017,**which was finalized in June 2017. It includes the **async/await** constructions which work using promises **(**In CS **future, promise, deferred,**and **delay** refers to the constructs which are used to synchronize the execution of the program in some concurrent programming languages**)** and generators.

ECMAScript 2017 or the eight edition also includes the features of atomic and concurrency, syntactic integration with promises.

9th Edition - ECMAScript 2018

The 9th edition was officially known as **ECMAScript 2018,**which was finalized in June 2018. It includes the new features like **rest/spread**operators for the variables **(**three dots:**…identifier),**asynchronous iteration, etc.

10th Edition - ECMAScript 2019

The 10th edition was officially known as **ECMAScript 2019,**which was published in June 2019. It includes the addition of some new features like **Array.prototype.flatMap, Array.prototype.flat,**and changes to **Array.sort**and **Object.fromEntries.**

ES.Next

It is nothing but a dynamic name that refers to the next version at the writing time. The harmony plans were too committed for the single version, which splits its features within the two groups: the first group had the highest priority and was to become the succeeding version after ES5. ECMAScript.next was the code name of that version, for avoiding the prematurely committing to a version number which already proved suspect with ES4. The second group had time until after ECMAScript.next.