JavaScript is most widely used now a days and most preferable language for development as front-end as well back-end(Nodejs),if we want to learn JavaScript we need to understand the basics of JavaScript and ECMAScript(ES), ECMAScript is an simple standard for JavaScript and adding new features to JavaScript,

ECMAScript is a subset of JavaScript. JavaScript is basically ECMAScript at its core but builds upon it. Languages such as ActionScript, JavaScript, JScript all use ECMAScript as its core. As a comparison, AS/JS/JScript are 3 different cars, but they all use the same engine… each of their exteriors is different though, and there have been several modifications done to each to make it unique.

***ECMAScript as Standard:***

* **ECMAScript:** Sun (now Oracle) had a trademark on the name“JavaScript” (which led to Microsoft calling its JavaScript dialect “JScript”). Thus, when it came to standardizing the language, a different name had to be used. Instead, “ECMAScript” was chosen, because the corresponding standard is hosted by Ecma International (see below). In practice, the terms “ECMAScript” and “JavaScript” are interchangeable. If JavaScript means “ECMAScript as implemented by Mozilla and others” then the the former is a dialect of the latter. The term “ECMAScript” is also frequently used to denote language versions (such as ECMAScript 5).
* **ECMA-262:** The [Ecma International](http://en.wikipedia.org/wiki/Ecma) (a standards organization) has created the ECMA-262 standard which is the official specification of the ECMAScript language.
* **Ecma’s**[**Technical Committee 39**](http://www.ecma-international.org/memento/TC39.htm)**(TC39):** is the group of people (Brendan Eich and others) who develop the ECMA-262 standard.
* **ECMAScript 3 (December 1999).** This is the version of ECMAScript that most browsers support today. It introduced many features that have become an inherent part of the language:

regular expressions, better string handling, new control statements, try/catch exception handling, tighter definition of errors, formatting for numeric output and other enhancements.

* **ECMAScript 4 (abandoned July 2008).** ECMAScript 4 was developed as the next version of JavaScript, with a prototype written in ML. However, TC39 could not agree on its feature set. To prevent an impasse, the committee met at the end of July 2008 and came to an accord, summarized in four points

1. Develop an incremental update of ECMAScript [which became ECMAScript 5.
2. Develop a major new release, which was to be more modest than ECMAScript 4, but much larger in scope than the version after ECMAScript 3. This version has been code-named Harmony, due to the nature of the meeting in which it was conceived.
3. Features from ECMAScript 4 that would be dropped: packages, namespaces, early binding.
4. Other ideas were to be developed in consensus with all of TC39.

* Thus: The ECMAScript 4 developers agreed to make Harmony less radical than ECMAScript 4, the rest of TC39 agreed to keep moving things forward.
* **ECMAScript 5 (December 2009).** This version brings several enhancements to the standard library and even updated language semantics via a strict mode.
* **currently browsers are supporting upto ES5 only.**

***ECMAScript 6(2015)***

ES6 is an next-gen of JavaScript, the ecma technical committee 39 governs ecma specification, they discovered new features to the javascript.

ES6 can`t compile directly in browsers, so need an compiler from ES6 to ES5, for that compiling using [babel](https://babeljs.io/), it produce the browser compatible javascript.

ES6 includes the following new features:

* [arrows](https://github.com/lukehoban/es6features#arrows)
* [classes](https://github.com/lukehoban/es6features#classes)
* [enhanced object literals](https://github.com/lukehoban/es6features#enhanced-object-literals)
* [template strings](https://github.com/lukehoban/es6features#template-strings)
* [destructuring](https://github.com/lukehoban/es6features#destructuring)
* [default + rest + spread](https://github.com/lukehoban/es6features#default--rest--spread)
* [let + const](https://github.com/lukehoban/es6features#let--const)
* [iterators + for..of](https://github.com/lukehoban/es6features#iterators--forof)
* [generators](https://github.com/lukehoban/es6features#generators)
* [unicode](https://github.com/lukehoban/es6features#unicode)
* [modules](https://github.com/lukehoban/es6features#modules)
* [module loaders](https://github.com/lukehoban/es6features#module-loaders)
* [map + set + weakmap + weakset](https://github.com/lukehoban/es6features#map--set--weakmap--weakset)
* [proxies](https://github.com/lukehoban/es6features#proxies)
* [symbols](https://github.com/lukehoban/es6features#symbols)
* [subclassable built-ins](https://github.com/lukehoban/es6features#subclassable-built-ins)
* [promises](https://github.com/lukehoban/es6features#promises)
* [math + number + string + array + object APIs](https://github.com/lukehoban/es6features#math--number--string--array--object-apis)
* [binary and octal literals](https://github.com/lukehoban/es6features#binary-and-octal-literals)
* [reflect api](https://github.com/lukehoban/es6features#reflect-api)
* [tail calls](https://github.com/lukehoban/es6features#tail-calls)

So above I have mentioned only list of features you can read it further,

***ECMAScript 7(2016)***

They decided to release a new version of ECMAScript every year starting in 2015. A yearly update means no more big releases like ES6.

ECMAScript 2016(ES7) introduced only two new features:

* Array.prototype.includes()
* Exponentiation operator

***ECMAScript 8(2017)***

At the TC39 meeting in January 2017, the last feature of ECMAScript 2017, “[Shared memory and atomics](http://2ality.com/2017/01/shared-array-buffer.html)” .

Major new features:

* [Async Functions (Brian Terlson)](http://2ality.com/2016/02/async-functions.html)
* [Shared memory and atomics (Lars T. Hansen)](http://2ality.com/2017/01/shared-array-buffer.html)

Minor new features:

* [Object.values/Object.entries (Jordan Harband)](http://2ality.com/2015/11/stage3-object-entries.html)
* [String padding (Jordan Harband, Rick Waldron)](http://2ality.com/2015/11/string-padding.html)
* [Object.getOwnPropertyDescriptors() (Jordan Harband, Andrea Giammarchi)](http://2ality.com/2016/02/object-getownpropertydescriptors.html)
* [Trailing commas in function parameter lists and calls (Jeff Morrison)](http://2ality.com/2015/11/trailing-comma-parameters.html)

***ECMAScript 9(2018)***

Major new features:

* [Asynchronous Iteration](http://2ality.com/2016/10/asynchronous-iteration.html) (Domenic Denicola, Kevin Smith)
* [Rest/Spread Properties](http://2ality.com/2016/10/rest-spread-properties.html) (Sebastian Markbåge)

New regular expression features:

* [RegExp named capture groups](http://2ality.com/2017/05/regexp-named-capture-groups.html) (Gorkem Yakin, Daniel Ehrenberg)
* [RegExp Unicode Property Escapes](http://2ality.com/2017/07/regexp-unicode-property-escapes.html) (Mathias Bynens)
* [RegExp Lookbehind Assertions](http://2ality.com/2017/05/regexp-lookbehind-assertions.html) (Gorkem Yakin, Nozomu Katō, Daniel Ehrenberg)
* [s (dotAll) flag for regular expressions](http://2ality.com/2017/07/regexp-dotall-flag.html) (Mathias Bynens)

Other new features:

* [Promise.prototype.finally()](http://2ality.com/2017/07/promise-prototype-finally.html) (Jordan Harband)
* [Template Literal Revision](http://2ality.com/2016/09/template-literal-revision.html) (Tim Disney)

**Summary**:

* ECMAScript is the “standard for” of the JavaScript language.

EcmaScript is the “official” name for JavaScript. It was eventually abandoned and ES3.1 became ES5, which is the JavaScript version used in the “HTML5” world.

we just known JavaScript history,why JavaScript evolved,In over the years javascript has changed through ECMAScript,

The main understanding of ES5 and ES6 can explore your JavaScript knowledge as well ES7,ES8,ES9 .

What is Babel?

Babel is a JavaScript compiler that can translate markup or programming languages into JavaScript.

With Babel, you can use the newest features of JavaScript (ES6 - ECMAScript 2015).

Babel is available for different conversions. React uses Babel to convert JSX into JavaScript.

Please note that <script type="text/babel"> is needed for using Babel.

## What is JSX?

JSX stands for **J**ava**S**cript **X**ML.

JSX is an XML/HTML like extension to JavaScript.

### **Example**

const element = <h1>Hello World!</h1>

As you can see above, JSX is not JavaScript nor HTML.

JSX is a XML syntax extension to JavaScript that also comes with the full power of ES6 (ECMAScript 2015).

Just like HTML, JSX tags can have a tag names, attributes, and children. If an attribute is wrapped in curly braces, the value is a JavaScript expression.

Note that JSX does not use quotes around the HTML text string.

## React DOM Render

The method ReactDom.render() is used to render (display) HTML elements:

### **Example**

<div id="id01">Hello World!</div>  
  
<script type="text/babel">  
ReactDOM.render(  
    <h1>Hello React!</h1>,  
    document.getElementById('id01'));  
</script>

## JSX Expressions

Expressions can be used in JSX by wrapping them in curly **{}** braces.

### **Example**

<div id="id01">Hello World!</div>  
  
<script type="text/babel">  
const **name** = 'John Doe';  
ReactDOM.render(  
    <h1>Hello **{name}**!</h1>,  
    document.getElementById('id01'));  
</script>

## React Elements

React applications are usually built around a single **HTML element**.

React developers often call this the **root node** (root element):

<div id="root"></div>

**React elements** look like this:

const element = <h1>Hello React!</h1>

Elements are **rendered** (displayed) with the ReactDOM.render() method:

ReactDOM.render(element, document.getElementById('root'));

[Try it Yourself »](https://www.w3schools.com/whatis/tryit.asp?filename=trywhatis_react_render)

React elements are **immutable**. They cannot be changed.

The only way to change a React element is to render a new element every time:

### **Example**

function tick() {  
    const element = (<h1>{new Date().toLocaleTimeString()}</h1>);  
    ReactDOM.render(element, document.getElementById('root'));  
}  
setInterval(tick, 1000);