新しい JavaScriptの教科書

境 祐司 著

SCC

新しい JavaScript の教科書

境 祐司 著

著者略歴

境 祐司（Yuji Sakai）

インストラクショナルデザイナーとして講座企画、IDマネジメント、記事執筆、講演などを中心に活動。2012年5月、電子出版専門のパブリッシャーとして電子書籍のプランニング、情報設計、デリバリデザイン等を手掛ける。2016年より、ニューラルネットワークを使ったスタイル置換処理等の研究に着手。
2017年より、Adobe Community Evangelistとして主にAdobe Sensei（AdobeのAI技術の総称）やXD関連のイベント登壇、企業向け講演等を中心に活動中。

Adobe Community Evangelist ■ https://adobe.ly/2XIuCJf
Twitter ■ https://twitter.com/commonstyle

著書

『実習で身につく！新しいWebデザインの教科書　～基礎から学べるHTML&CSSデザイン～』（SCC）『Adobe Muse ランディングページ制作ガイド ～コード知識ゼロで作るWeb広告』（技術評論社）など。

サポートページのご案内

下記の本書専用サポートページでは、正誤情報や、本書に記載された項目等に関する補足情報などを、必要に応じて掲載します。

本書専用サポートページ
https://www.scc-kk.co.jp/scc-books/b-414.html

なお、サポートページの内容は必要に応じて随時更新されますので、ご注意下さい。

はじめに

インターネット黎明期の代表的なブラウザーだった「Netscape Navigator」のスクリプト言語として登場した「JavaScript」は、数あるプログラミング言語の中でも大変ユニークな存在です。

Microsoftが”JavaScriptと類似した”スクリプト言語「JScript」を同社のブラウザー「Internet Explorer」に搭載し、Netscapeとの熾烈なブラウザー戦争を繰り広げていた1990年代後半、世界中のWeb開発者はその互換性の低さに頭を悩ませていました。

1997年、標準化団体のEcma Internationalが、JavaScriptとJScriptの共有化できる部分を抜き出して標準化しましたが、この仕様が「ECMAScript」です。なお、実際のプログラミングでよく利用されるDOM（Document Object Model）などは、技術コミュニティの「WHATWG」が策定を行っています。

JavaScriptはこのような多層的な言語であり、進化が速く、Webのトレンドにも密接しています。

しかし、JavaScriptの学習を始めたばかりの初心者にとって、これらの関係は理解しづらいものです。そこで本書は、JavaScriptの全体像を常に意識しながら、言語仕様であるECMAScript、ブラウザーに文字や画像を表示する仕組み、ブラウザーとプログラムをつなぐAPIなどの技術を体系的に学べるように構成しました。

各章の冒頭では、達成目標や学習のポイント、学習の流れを理解し、「今何を学んでいるのか」を常に把握できます。章末では、練習問題や達成目標のチェックシートで理解度を自己評価し、理解が曖昧な箇所や、復習すべき学習項目を洗い出すことができます。

また、JavaScriptには20年以上の歴史があり、何度も改訂が繰り返されてきましたが、特に大きな改訂となったのがECMAScript2015（ES6）です。ソフトウェア開発の生産性に関わる重要な仕様が盛り込まれており、これからJavaScriptを学ぶ人たちにとっての「入り口」になります。本書はこのES6に準拠しているので、より実践的なスキルを身につけることができます。

JavaScriptは他のプログラミング言語より敷居が低い（初心者向きの）言語として知られていますが、仕様や技術の全体像を理解していないと、なかなか使いこなせないものです。

本書は初心者が覚えるべき要素を厳選して解説しているので、基礎を学ぶ最初の一冊として最適です。JavaScript習得のきっかけとして、役立てていただければ幸いです。

2019年11月

境 祐司

本書の読み方

本書は全 12 章を通して、JavaScript に関するさまざまな知識を解説しています。15 時限の授業を想定して構成していますが、ご自身の学習しやすいースで読み進めてください。

Chapter 1 WebとJavaScript

1時限 講義

Step-01	JavaScript 誕生の歴史
Step-02	Webブラウザーの種類
Step-03	JavaScript とECMAScriptの関係

達成目標

1. JavaScript がどのような経緯で誕生したか他人に説明できる
2. 代表的な6つのWebブラウザーを列挙できる
3. レンダリングエンジンとJavaScriptの役割を理解している
4. ECMAScriptとは何か理解している
5. JavaScript の全体像について他人にわかりやすく説明できる

Chapter 2 準備と基本文法

2時限 講義 実習

Step-01	JavaScriptを学ぶための準備をしておこう
Step-02	HTML ファイルを作成する
Step-03	オートコンプリートを活用してコードを記述しよう
Step-04	プログラミングの「デバッグ」を体験してみよう
Step-05	外部のJavaScriptファイルを作成してHTMLに読み込む
Step-06	用途に応じてコメントを記述する

達成目標

1. モダンなコードエディタ「Atom」の基本的な使い方がわかる
2. 基本的なHTMLコードをコードエディタの入力支援機能を活用して記述できる
3. 簡単なJavaScriptのコードを記述できる
4. HTMLおよびJavaScriptのコメントを用途に応じて記述できる
5. Google Chromeの開発者向けツールを活用して「バグ」をすばやく発見できる

Chapter 3 変数とデータ型

3時限 実習

Step-01	変数とは？
Step-02	演算子の種類と優先順位
Step-03	JavaScript のデータ型

達成目標

1. 変数の役割と使い方を他人に説明することができる
2. 変数を宣言するときに必要な3種類のキーワードについて理解している
3. 変数名を決めるときのルールや記法について理解している
4. 演算子の種類や優先順位について理解している
5. データ型とは何か他人にわかりやすく説明することができる

Chapter 4 条件分岐

4時限 実習

Step-01	アルゴリズムって？
Step-02	体格指数を算出するプログラムをif文で書いてみよう
Step-03	体格指数を算出するプログラムをif/else文で改良しよう
Step-04	曜日によって表示を変更するプログラムをswitch文で書いてみよう

達成目標

1. アルゴリズムとは何か他人に説明することができる
2. if 文で使う比較演算子の使い方を理解している
3. if/else 文の使い方を理解している
4. if 文で使う論理演算子の使い方を理解している
5. switch文とif/else文の違いについて他人に説明することができる

Chapter 5 繰り返し

5時限 実習

Step-01	ループ文
Step-02	while文とdo while文を理解する
Step-03	for 文を理解する
Step-04	break文とcontinue文を理解する

達成目標

1. ループ文とは何か他人に説明することができる
2. while文とdo while文がどのような処理でどこが異なるのか理解している
3. for 文がどのような処理なのか理解している
4. break文とcontinue文がどのような処理でどこが異なるのか理解している
5. 制御構文の分類、それぞれの特徴を他人に説明することができる

Chapter 6 配列

6時限 実習

Step-01	配列とは何か？
Step-02	JavaScript の配列はオブジェクト
Step-03	連想配列
Step-04	基本的な配列の操作

達成目標

1. JavaScript の配列とは何か他人に説明することができる
2. JavaやCなどの配列と何が異なるのか理解している
3. 連想配列について他人にわかりやすく説明することができる
4. 配列に格納した要素を取り出したり入れ替えたりすることができる
5. 配列に新しい要素を追加できる

Chapter 7 関数

7時限 実習

Step-01 関数の基礎知識
Step-02 関数の使い方
Step-03 関数を使って簡単なプログラムを作る

達成目標
1. JavaScript の関数とはどのようなものか他人に説明することができる
2. 関数を使うメリットについて他人にわかりやすく説明することができる
3. 関数を定義することができる
4. 関数のことを「メソッド」と呼ぶ理由について理解している
5. 関数リテラルについて理解している

Chapter 8 組み込みオブジェクト

8時限 実習

Step-01 JavaScript の組み込みオブジェクト
Step-02 Dateオブジェクトを使う
Step-03 String オブジェクトを使う

達成目標
1. JavaScript の組み込みオブジェクトについて他人に説明することができる
2. 組み込みオブジェクトにどのようなものがあるか理解している
3. 今日の日付を取得して「○○年○○月○○日」のように表示できる
4. さらに曜日を取得して「○○年○○月○○日（曜日）」と表示できる
5. 「東京都→東京」のように指定した範囲の文字列を取り出すことができる

Chapter 9 オブジェクト指向プログラミング

9時限 講義

Step-01 オブジェクト指向とは？
Step-02 DOMオブジェクトとは？

10時限 実習

Step-03 DOMの操作

達成目標
1. オブジェクト指向やクラス／インスタンスについて理解している
2. JavaScript がプロトタイプベースのオブジェクト指向であることを理解している
3. DOMとは何か他人に説明することができる
4. DOMツリーが何を意味しているのか理解している
5. DOMオブジェクトを使ってHTMLの要素を追加したり削除したりすることができる

Chapter 10 Web API

11時限 講義 実習

Step-01 APIとは？
Step-02 Webブラウザーのオブジェクト

12時限 実習

Step-03 その他のオブジェクト（XMLHttpRequest）

達成目標
1. APIとは何か他人に説明することができる
2. ネイティブオブジェクトとホストオブジェクトの違いを理解している
3. ブラウザーのAPIとサードパーティのAPIの違いを理解している
4. ブラウザーのオブジェクトでどんなことができるのか理解している
5. Ajaxとは何か理解している
6. JSONとは何か他人に説明することができる

Chapter11 jQuery

13時限 実習

Step-01 JavaScript のライブラリ「jQuery」
Step-02 スライドショーを作成する

達成目標
1. JavaScript のライブラリやフレームワークについて理解している
2. jQueryとは何か他人に説明することができる
3. jQuery CDNを利用するメリットについて理解している
4. uncompressedとminifiedの違いを他人に説明することができる
5. jQueryを使ってWebページの見出しの色を変更することができる
6. スライドショーのライブラリ（bxSlider）を使用することができる

Chapter12 アニメーション

14時限 実習

Step-01 Vue.js の使い方
Step-02 基本的なアニメーション表現
Step-03 モーションデザイン

達成目標
1. jQueryとVue.jsの違いについて理解している
2. プログレッシブ・フレームワークという概念について理解している
3. Vue.js をインストールして簡単なプログラムを動かすことができる
4. Vue.js で文字列などをフェードイン･アウトする表現ができる
5. Vue.js でイージング効果（加速減速）を適用できる
6. リスト項目をアニメーションさせながら置き換えることができる

総合演習 （第3～10章の練習問題の復習）

15時限 実習

Chapter 3 変数とデータ型　練習問題の復習
Chapter 4 条件分岐　練習問題の復習
Chapter 5 繰り返し　練習問題の復習
Chapter 6 配列　練習問題の復習
Chapter 7 関数　練習問題の復習
Chapter 8 組み込みオブジェクト　練習問題の復習
Chapter 9 オブジェクト指向プログラミング　練習問題の復習
Chapter 10 Web API　練習問題の復習

目次

Chapter

1

WebとJavaScript

JavaScriptは、クライアントサイドのスクリプト言語です。Webブラウザーとエディタがあれば誰でもすぐにコードを書くことができます（この手軽さがJavaScriptの魅力のひとつ）。初心者にとって理解しにくいのは、JavaScriptの全体像です。全体像を把握するにはECMAScriptやAPI、JavaScriptエンジンなどの基礎知識が必要になります。

本章で学ぶこと

▶ **達成目標：**

1. JavaScriptがどのような経緯で誕生したか他人に説明できる
2. 代表的な6つのWebブラウザーを列挙できる
3. レンダリングエンジンとJavaScriptの役割を理解している
4. ECMAScriptとは何か理解している
5. JavaScriptの全体像について他人にわかりやすく説明できる

▶ **学習ポイント：**

- JavaScriptが生まれた歴史的背景を学ぶことができます
- Webブラウザーのレンダリングエンジンとは何か解説します
- WebブラウザーのJavaScriptエンジンとは何か解説します
- JavaScriptとECMAScriptの関係について詳しく学べます
- JavaScriptの全体像をしっかり理解することができます

▶ **学習の流れ：**

- Step-01 JavaScript誕生の歴史
- Step-02 Webブラウザーの種類
- Step-03 JavaScriptとECMAScriptの関係

Step 01

JavaScript誕生の歴史

JavaScriptは、Webサイトやアプリなどのあらゆる開発が手軽にできるプログラミング言語です。誕生の経緯や、関連する企業などを見てみましょう。

JavaScriptとは？

プログラミングは、英会話の習得と同じように2つの学び方があります。ひとつは「最初に文法や単語をしっかりと学習してから会話の練習に進む」方法、もうひとつは「基本をある程度理解したら後は実践しながら（失敗と学びを繰り返しながら）身につけていく」方法です。

どちらが早道になるかは学習する人の態度によって変わってきますが、JavaScriptに関しては後者のほうが「早く慣れることができる」有効な方法だと考えています。実際にコードを書いてプログラムを実行し、デバッグしながら同時に学習していくやり方です。

JavaScript（ジャバスクリプト）は、Webブラウザーとメモ帳のような簡易エディタがあれば、すぐにコードを書いて実行できるプログラミング言語です。誰でも今すぐに始められる敷居の低さが、JavaScriptの大きな特長だといえるでしょう。HTMLやCSSのように、一般のユーザーでもある程度のレベルであれば習得可能な言語です。

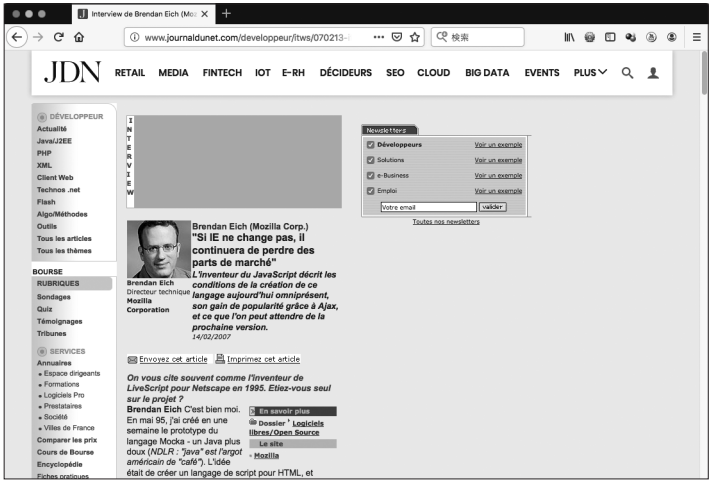
プログラミング言語の中で「異端児」と呼ばれるJavaScriptがどのような経緯で登場したのか見ていきましょう。

JavaScriptとJavaはどう違うの？

JavaScriptは、1995年にネットスケープコミュニケーションズのプログラマーであるブレンダン・アイク（Brendan Eich）によって開発されたプログラミング言語です。開発コードネームは「Mocha（モカ）」で、Webブラウザーの「Netscape Navigator 2.0 ベータ版」リリースに間に合わせるためプロトタイプは10日間（1995年5月6〜15日）で開発されました。1995年9月に「LiveScript（ライブスクリプト）」という名称に変わりましたが、リリース直前になって「JavaScript」に変更されました。

▶ **参考：**

- **ブレンダン・アイクのインタビュー記事（2007年2月14日）**
  たった一人で開発したこと、プロトタイプを10日間で作ったことなどを語っています。
  URL：http://www.journaldunet.com/developpeur/itws/070213-itw-mozilla-eich.shtml



1995年4月11日、ネットスケープコミュニケーションズは、プログラミング言語「**Java（ジャバ）**」を開発していたサン・マイクロシステムズと業務提携を締結し、（Webブラウザー開発で競っていた）マイクロソフトに対抗するための施策のひとつとして「JavaScript」を同社のWebブラウザーに実装しました。

JavaScriptという命名はあくまで戦略的なもので、Javaとの特別な関係はありませんが、ネットスケープとサンの共同発表（1995年12月4日）では「Javaの簡易版」だと誤解を与えるような表現をしており、大きな混乱を招きました。当時の文献を見ると誤ったJavaScriptの紹介記事が散見されます。

ブレンダン・アイクはJavaScript開発が評価され、1998年にはネットメディアのCNET Builder.comが主催するWeb Innovator Awardsにて、Web Innovator of the Yearを受賞しています。現在でも多くのテックメディアが「JavaScriptの父」と表現しており、その功績をたたえています。

▶ JavaScript が誕生した 1995 年の出来事

開発期間およびプレスリリース	JavaScript開発の動向
5～9月	ブレンダン・アイクが「Mocha」を開発
9～12月	「LiveScript」と呼ばれていた期間
12月1日	サン・マイクロシステムズが「JavaScript」の商標登録を申請
12月4日	ネットスケープとサンによる「JavaScript」の共同発表

商標の正式登録（#2416017）は2000年12月26日です。2010年1月27日、サンがオラクルに買収されたことで「JavaScript」の商標権はオラクルに移転しています。

▶ 補足（関連企業について）：

JavaScriptはネットスケープコミュニケーションズのWebブラウザー「Netscape Navigator」のスクリプト言語です。Javaとの技術的な関係はありませんが、開発元のサン・マイクロシステムズとは業務提携していますので、ビジネス上の関連性はあったと考えてよいでしょう。

また、JavaScriptの発展には、マイクロソフトとの熾烈な開発競争も大きく影響していますので、関連企業について情報整理しておきましょう。

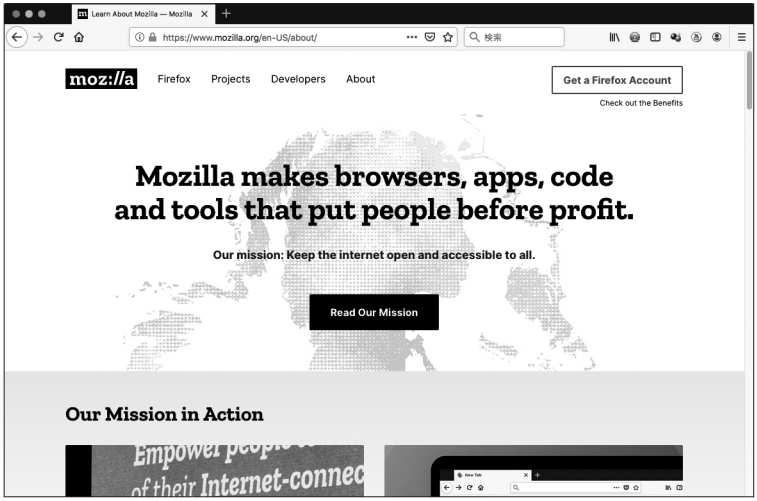
▶ ネットスケープコミュニケーションズ（Netscape Communications Corporation）

イリノイ大学のNCSA（National Center for Supercomputing Applications：米国立スーパーコンピュータ応用研究所）に在籍していたマーク・アンドリーセン（Marc Lowell Andreessen）が、実業家のジム・クラーク（James H. Clark）に誘われ、設立された企業です。マーク・アンドリーセンは、現在のWebブラウザーのルーツといえる「NCSA Mosaic（モザイク）」の開発に携わっていました。

同社は、インターネット黎明期の代表的なブラウザーとなる「Netscape Navigator（ネットスケープ・ナビゲーター）」を開発。1995年にリリースされたNetscape Navigator 2.0に、クライアントサイドのスクリプト言語「JavaScript」が実装されました。

1998年、同社は大手ネット企業のAOLに買収されますが、Netscapeはオープンソースソフトウェアとなり、非営利団体の「Mozilla Organization」が引き継ぎ、2003年7月15日に「Mozilla Foundation」を設立、2004年にWebブラウザー「Mozilla Firefox（モジラ・ファイアーフォックス）」のバージョン1をリリース、2005年8月3日には完全子会社となる「Mozilla Corporation」を設立しました。

- **Learn About Mozilla**
  https://www.mozilla.org/en-US/about/



▶ サン・マイクロシステムズ（Sun Microsystems）

1982年に設立されたコンピュータ企業で、業務用コンピュータの開発製造やソフトウェア開発、ITサービスなど事業を幅広く展開、マイクロソフトと並ぶ大企業に成長。1995年に開発されたオブジェクト指向プログラミング言語「Java」が人気を博し、1996年1月23日に「JDK（Java Development Kit）1.0」がリリースされます。

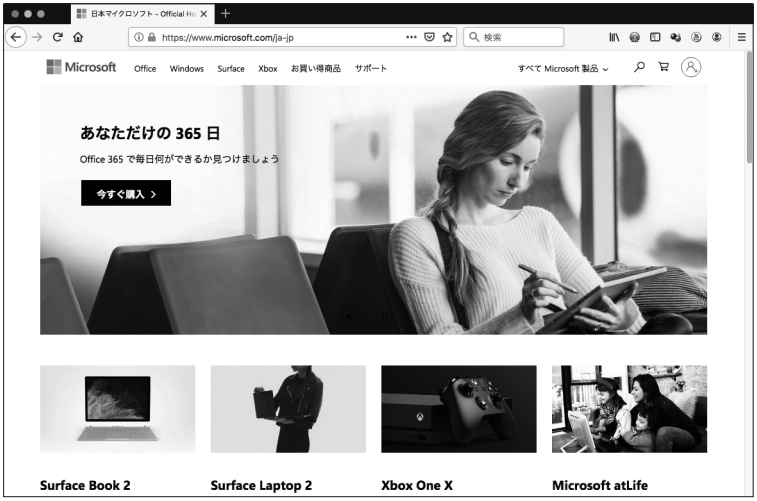
1995年4月11日、ネットスケープコミュニケーションズとの業務提携を締結し、12月4日には「JavaScript」の共同発表を行っています。なお、JavaScriptの商標権はサンが取得していますが、2010年1月27日、米国のソフトウェア開発企業「オラクル（Oracle Corporation）」がサンを買収し、商標権はオラクルに移転しています。

▶ マイクロソフト（Microsoft Corporation）

WindowsやOfficeなどで世界的に知られている大企業。1995年8月に「Internet Explorer（インターネット・エクスプローラー）」をWindowsの機能拡張パッケージに収録。1996年8月にリリースされたInternet Explorer 3.0には、JavaScriptと類似したスクリプト言語「JScript（ジェイ・スクリプト）」が実装されています。JavaScriptのライセンス供与が認められなかったのが独自開発の理由です。

ネットスケープコミュニケーションズとのユーザー獲得をめぐる開発競争は、Web標準の技術であるHTMLを独自に拡張するなど、Webの信頼性を低下させる大きな問題に発展していきます。この混乱期は「ブラウザー戦争」と呼ばれ、Internet Explorerが圧倒的なシェアを得る2000年頃まで続きました。

- **Microsoft**
  https://www.microsoft.com/ja-jp



Internet Explorerの最終バージョンは、2013年10月に公開されたIE 11ですが、Windows 10で「Microsoft Edge」が標準ブラウザーに切り替わったことで、IEのユーザー数は減り続けています。2020年1月14日にはIE 10のサポートが終了するため、IEはバージョン11のみとなります。

IEは過去のブラウザーになりましたが、古いWindowsを使っているユーザーがまだ多いため、しばらくはEdgeとIE 11が混在した状態が続きます。よって、Webアプリ開発では両方の検証が必要になります。

Step 02

Webブラウザーの種類

Webページを閲覧するために必要なアプリが「Webブラウザー」です。ここでは、Webブラウザーの種類と、プログラムを解釈して表示する「エンジン」について解説します。

OSの標準ブラウザーとサードパーティのブラウザー

Webブラウザーは、Webページをスクロールしながら読む閲覧用のアプリケーションソフトです。OSにインストール済みの標準ブラウザーと他の企業が提供するサードパーティのブラウザーに大別することができます。

Windowsが搭載されているパソコンやタブレットPCには「**Microsoft Edge**」、macOSが搭載されているMacやiOSが搭載されているiPhoneやiPadには「**Apple Safari**」、そしてAndroidが搭載されているデバイスには「**Google Chrome**」がインストールされています。

その他、Mozilla Foundationの「**Mozilla Firefox**」やOpera Softwareの「**Opera**」などのWebブラウザーが無償で提供されており、誰でも自由にインストールすることができます。

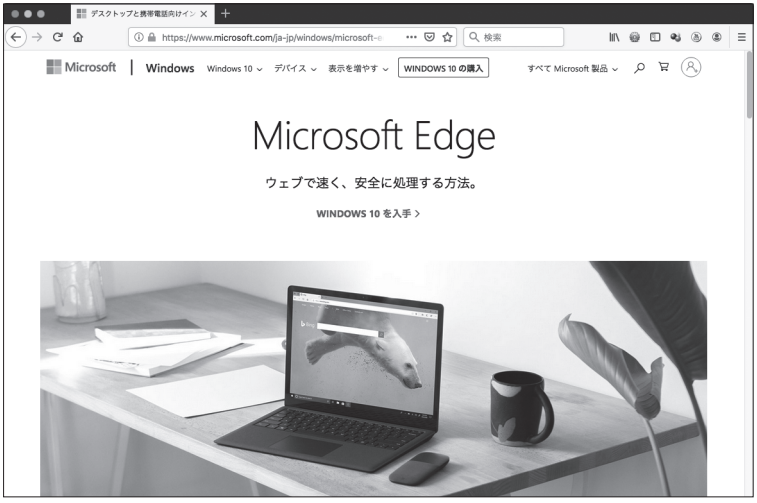
Microsoft EdgeはWindows 10以降の標準ブラウザーですが、それ以前の古いOSでは「Internet Explorer」が使用されています。Internet Explorer 8〜10はすでにサポートが打ち切られており、マイクロソフトはInternet Explorerの非推奨を表明しています。最終バージョンである11のみサポートが継続されており、利用者もいますのでしばらくは対応が必要です。

▶ 参考：

- **Internet Explorer の今後について［Microsoft Developer Network］**
  短縮URL：http://bit.ly/2LJCHe4

Webブラウザー名	開発企業・団体	種類
Microsoft Edge （マイクロソフト・エッジ）	Microsoft	Windows 10以降の標準ブラウザー
Internet Explorer （インターネットエクスプローラー）	Microsoft	Windows標準ブラウザー
Safari（サファリ）	Apple	macOSおよびiOS標準ブラウザー
Google Chrome （グーグル・クローム）	Google	Android標準ブラウザー（他のOSにも提供）
Mozilla Firefox （モジラ・ファイアーフォックス）	Mozilla Foundation	サードパーティ製品（ユーザーが自分でインストールする）
Opera（オペラ）	Opera Software	サードパーティ製品（ユーザーが自分でインストールする）

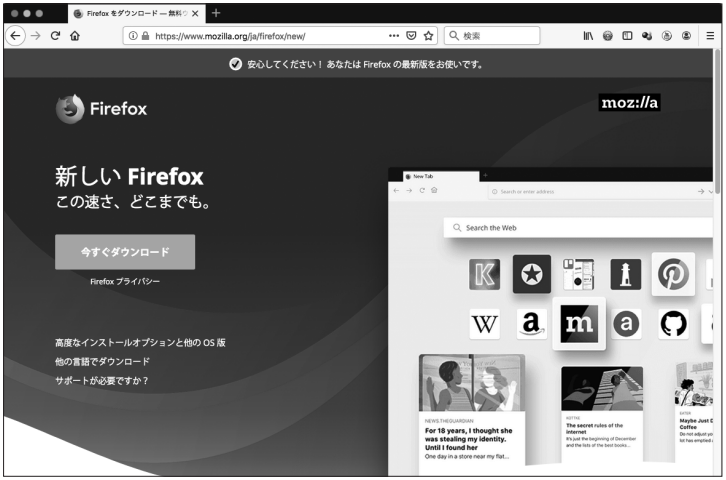
- **Microsoft Edge**
  https://www.microsoft.com/ja-jp/windows/microsoft-edge



- **Apple Safari**
  https://www.apple.com/jp/safari/

- **Google Chrome**
  https://www.google.co.jp/chrome/

- **Mozilla Firefox**
  https://www.mozilla.org/ja/firefox/new/

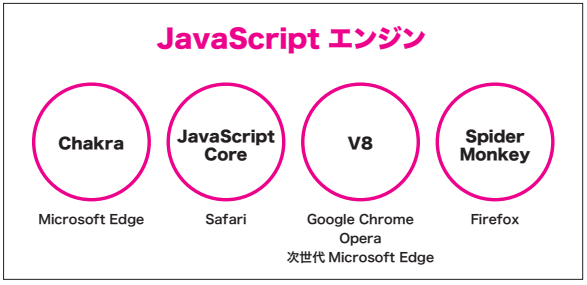


レンダリングエンジンとJavaScriptエンジン

Webブラウザーには、HTMLやCSSを解釈してブラウザーのウィンドウに文字や画像を表示するための「**レンダリングエンジン**」が搭載されています。クルマのエンジンと同様にメーカーによって差異があり、あるエンジンでは表示できても、他のエンジンでは表示できない、といったHTMLやCSSの実装の違いがあります。

JavaScriptのエンジンもWebブラウザーによって異なり、Microsoft Edgeは「**Chakra**」、Internet Explorerは「**Jscript9の Chakra**」、Safariは「**JavaScriptCore**」、Google ChromeとOperaは「**V8**」、Mozilla Firefoxは「**SpiderMonkey**」を搭載しています。

※2018年12月、次世代のMicrosoft EdgeはChromiumベースとして開発されることが発表され、2019年4月8日にプレビュー版が公開されました。JavaScriptのエンジンも、ChakraからV8に切り替わります。



▶ Web ブラウザーが採用しているエンジン

Web ブラウザー名	レンダリングエンジン名	JavaScript エンジン名
Microsoft Edge	EdgeHTML（エッジエイチティーエムエル）	Chakra（チャクラ）
Internet Explorer	Trident（トライデント）	Jscript9.dll Chakra
Safari	WebKit（ウェブキット）	JavaScriptCore（Nitro とも呼ばれる）
Google Chrome	Blink（ブリンク）	V8（ブイエイト）
Mozilla Firefox	Gecko（ゲッコー）＋一部は Servo（サーボ）	SpiderMonkey（スパイダーモンキー）
Opera	Blink	V8
次世代 Microsoft Edge	Blink	V8

▶ 参考：

- **Microsoft Edge - Build 2019 からのすべてのニュース**
  短縮 URL：http://bit.ly/2Y7O1r8
- **2017 年 11 月 30 日に正式リリースされた Android 版の Microsoft Edge には、Blink が採用されています。**
  短縮 URL：http://bit.ly/32YnlYM

Step 03

JavaScriptとECMAScriptの関係

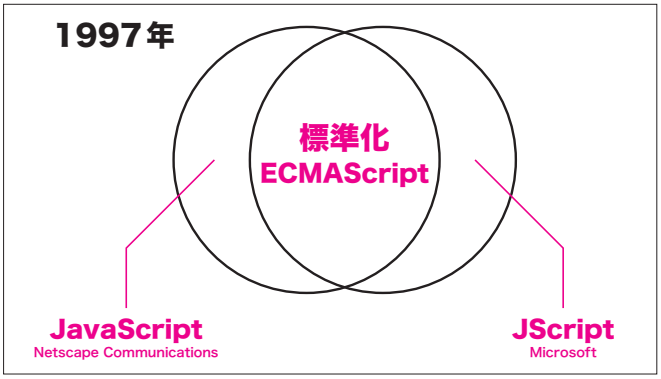
ここでは、JavaScriptを理解する上で必須である標準化規格「ECMAScript」が策定された背景から、JavaScriptの全体像までをわかりやすく解説します。

マイクロソフトのJScriptって何？

JavaScriptの文献には「**ECMAScript（エクマスクリプト）**」という名称がよく出てきます。初心者にとって、JavaScriptの全体像を理解するのは容易ではありません。なぜなら、歴史的背景を理解しておく必要があるからです。

Step-01では「JavaScript誕生の歴史」について学習しましたが、ネットスケープコミュニケーションズが開発した「JavaScript」と、マイクロソフトが開発した“JavaScriptと類似した”スクリプト言語「JScript」の存在を思い出してください。互換性に問題がある一見同じような言語が異なるWebブラウザーに搭載されているのですから、Web開発者にとってこんな厄介なことはありません。両方のブラウザーで正しく動作するプログラムを書くのは至難の業だったのですから。

この2つのスクリプトの互換性を高めるには、文法などの基本的な仕様を「標準化」しなければいけません。

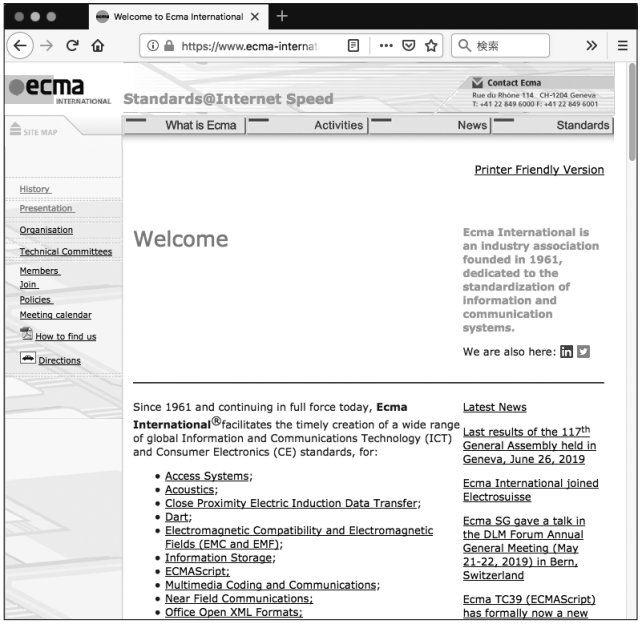


1997年、情報通信システム分野における標準化団体である「Ecma International（エクマ・インターナショナル）」が、**ネットスケープコミュニケーションズの「JavaScript」とマイクロソフトの「JScript」の共有化できる部分を抜き出し、標準化しました。**この標準化された仕様が「ECMAScript」です。

現在のブラウザーベンダーは「ECMAScript」を実装しているので、互換性の問題はほぼ解消しています。

※ECMAScriptは、Ecma InternationalのTC-39委員会が標準化を実行し、ECMA-262というドキュメントで公開されました。

- **Ecma International**
  https://www.ecma-international.org/



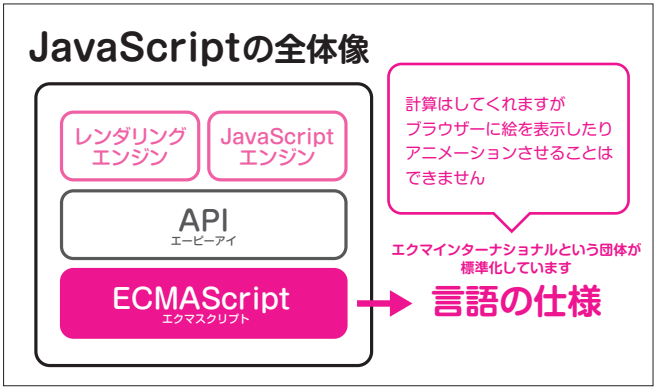
高校生でもわかるJavaScriptの全体像

JavaScriptの歴史的背景について理解できたと思いますので、続いて「JavaScriptの全体像」について学びましょう。初心者の皆さんにとっては（コーディングを習得する前に）必ずやらなくてはいけない重要な学習です。全体像が理解できていないと、学習効率にも影響してしまいます。

このパートはより理解しやすいように「**高校生でもわかるJavaScriptの全体像**」と題して、情報をコンパクトにして進めていきます。詳しい解説はこの後の章でやっていきますので、ここでは大まかに把握することを目標とします。

❶ JavaScript の箱があります。中には「EcmaScript」「API」、そして 2 つのエンジンが入っています。

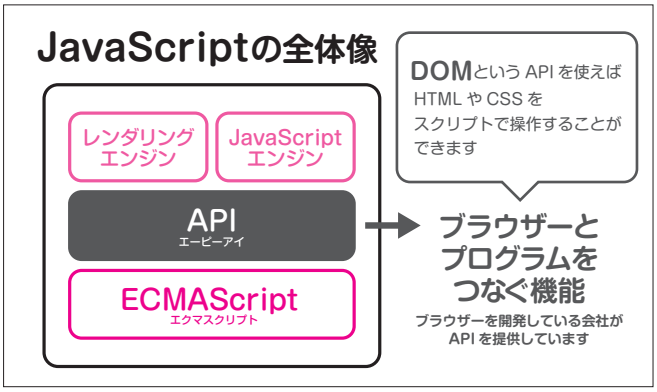
一番に下にある（基盤の部分の）EcmaScript は「**言語の仕様**」です。計算をさせることはできますが、Web ブラウザーに画像を表示したり、アニメーションを設定させることはできません。



❷真ん中にある「API」は Web ブラウザーとプログラムをつなぐ機能の集まりです。

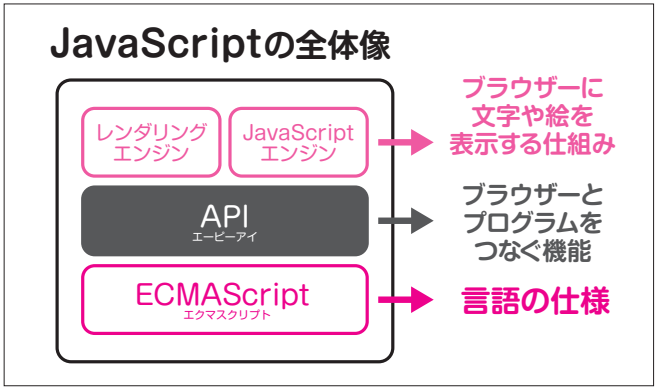
例えば、DOM という API を使用すればスクリプトで HTML や CSS を操作することができます。この API を使えば、指定した画像を表示したり、アニメーションを設定することも可能になります。

EcmaScript は標準化団体の「Ecma International」が策定していますが、API は Web ブラウザーを開発している会社が提供しています。



❸一番上の２つのエンジンは「処理」機能です。

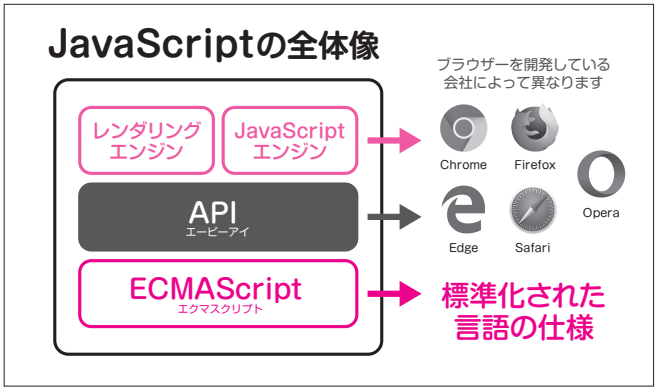
この仕組みによってWebブラウザーのウィンドウにテキストや画像をリアルタイム表示したり、スクリプトを解釈して実行したりしてくれます。レンダリングエンジンとJavaScriptエンジンについてはすでに学習済みです。



❹全体像についてまとめましょう。

「EcmaScript」は標準化された言語の仕様です。JavaScriptの核の部分だと捉えてください。JavaScriptの文法の学習は「**EcmaScriptの学習**」ということになりますので覚えておきましょう。

APIとレンダリングエンジン／JavaScriptエンジンはWebブラウザー固有の技術になります。



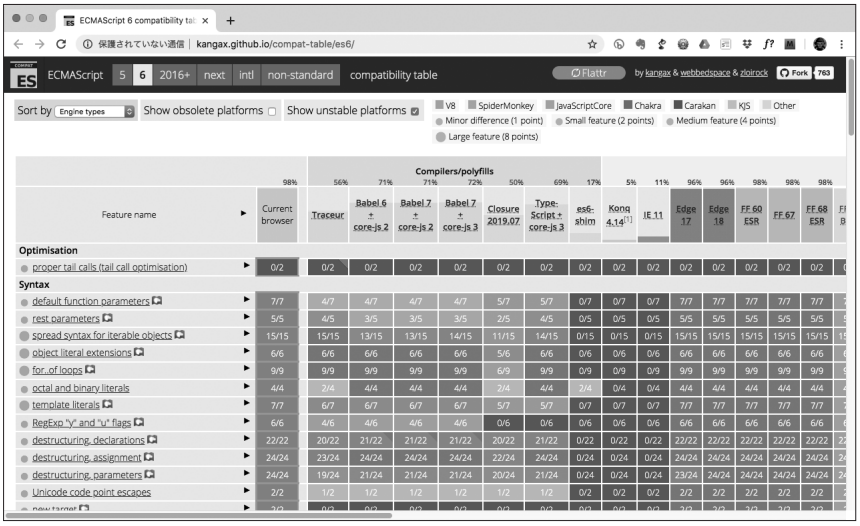
ECMAScript のバージョン

HTMLやCSSにバージョンがあるように**ECMAScriptにもバージョンがあります**。新しいバージョンは時間をかけてゆっくりと普及していきますので、まずは（現時点で）普及率の高いバージョンをしっかり習得した上で、最新のバージョンの機能を覚えていくことになります。

2019年7月現在、習得すべきバージョンは2015年6月に公開された「**ECMAScript 2015 6th Edition**」です。略して「ES6」とも呼ばれます。

▶ ECMAScript 2015［ES6］の Web ブラウザーの対応状況

http://kangax.github.io/compat-table/es6/



ECMAScript 2015以前はすべての仕様の合意がなければリリースできなかったので、策定に大変時間がかかっていました。バージョン3から5.xまでは10年以上もかかっています。ECMAScript 2016以降は、仕様策定のプロセスが変わり、次の図で示されているとおり毎年リリースされています。

▶ ECMAScript のバージョンとリリース日

リリース日	ECMAScriptのバージョン	略称
1995年	Netscape 2.0のベータ版に最初のJavaScriptを実装	
1996年3月	Netscape 2.0が、JavaScript 1.0 を実装	
1996年8月	Netscape 3.0が、JavaScript 1.1 を実装	
1996年11月	ECMAScript 言語仕様の開発が始まる	
1997年6月	ECMAScript 1th Editionが、Ecma総会で採択	ES
1997年7月	Netscape 4.0および4.05が、JavaScript 1.2 を実装	
1998年4月	ISO/IEC 16262 international standard対応（仕様は変わらず）	
1998年6月	ECMA-262 2th Editionが、Ecma総会で承認	ES2
1998年10月	Netscape 4.06および4.7が、JavaScript 1.3 を実装	
1999年12月	3th Editionが、Ecma総会で採択	ES3
2002年6月	ISO/IEC 16262：2002 国際標準化の公開	
	4th Editionは破棄（仕様の一部は6th Editionへ）	ES4
2009年12月	5th Editionが、Ecma総会で採択	ES5
2011年6月	5.1が、Ecma総会で採択	ES5.1
2015年6月	ECMAScript 2015 6th Edition（※大きな改訂）	ES2015 [ES6]
2016年6月	ECMAScript 2016 7th Edition	ES2016 [ES7]
2017年6月	ECMAScript 2017 8th Edition	ES2017 [ES8]
2018年6月	ECMAScript 2018 9th Edition	ES2018 [ES9]

- **ECMAScript® 2018**
  https://www.ecma-international.org/ecma-262/9.0/index.html

※バージョン4は意見がまとまらず破棄されましたが、Adobe Flashのスクリプト言語「ActionScript」はこのバージョンの草案を採用しています。W3Cが策定しているHTMLもバージョン3.0が廃案になっています（バージョン3.2が勧告されています）。

▶ ECMAScriptの仕様書

- **ECMAScript 2015**
  ［HTML］http://www.ecma-international.org/ecma-262/6.0/index.html
  ［PDF］http://www.ecma-international.org/ecma-262/6.0/ECMA-262.pdf
- **ECMAScript 2016**
  ［HTML］http://www.ecma-international.org/ecma-262/7.0/index.html
  ［PDF］短縮URL：http://bit.ly/2Gyrzwk
- **ECMAScript 2017**
  ［HTML］http://www.ecma-international.org/ecma-262/8.0/index.html
  ［PDF］短縮URL：http://bit.ly/2oP7qwb
- **ECMAScript 2018**
  ［HTML］http://www.ecma-international.org/ecma-262/9.0/index.html
  ［PDF］http://www.ecma-international.org/publications/files/ECMA-ST/Ecma-262.pdf