

知 識 ゼ ロ か ら は じ め る

ゆっくり・ていねい

JavaScript

ES6対応

かのうすけはる
狩野祐東[著]

[諸注意]

- ・本書に記載されている情報は 2018 年 12 月時点のものであり、URL 等の各種情報や内容は、ご利用時には変更されている可能性があります。
- ・本書の内容は参照としてのみ使用されるべきものであり、予告なしに変更されることがあります。また、ソシム株式会社がその内容を保証するものではありません。本書の内容に誤りや不正確な記述がある場合も、ソシム株式会社は一切の責任を負いません。
- ・本書の内容の運用によっていかなる障害が生じても、ソシム株式会社、著者のいずれも責任を負いかねますので、あらかじめご了承ください。
- ・本書の一部または全部について、ソシム株式会社との書面による事前の許諾なしに、電気、機械、複写、録音、その他のいかなる形式や手段によっても、複製、および検索システムへの保存や転送は禁止されています。

[商標等について]

- ・Mozilla、Firefox の名称およびそのロゴは、米国 Mozilla Foundation の米国およびその他の国における商標または登録商標です。
- ・Microsoft、Internet Explorer、Windows、および Microsoft のロゴは、米国およびその他の国における Microsoft Corporation および / またはその関連会社の登録商標または商標です。
- ・Apple、macOS、Safari および iPhone の名称およびそのロゴは、米国 Apple Inc. の米国およびその他の国における商標または登録商標です。
 - ※ iPhone 商標は、アイホン株式会社のライセンスに基づき使用されています。
- ・Yahoo! は、米国 Yahoo! Inc. の登録商標または商標です。
- ・そのほか、本書に記載されている社名、商品名、製品名、ブランド名、システム名などは、一般に各社の商標または登録商標です。
- ・本文中では TM、©、® を表示していません。

はじめに

ゆっくり・ていねい。 そして「楽しく」



JavaScriptはブラウザを操作し、表示されているWebページのHTMLをリアルタイムで書き換えられるプログラミング言語です。環境設定をほとんどせずに始められる手軽さだけでなく、「ブラウザを操作する」という特性上、なにをやっているのかがイメージしやすく、またプログラミングをした結果が理解しやすいこともあって、[初学者にも取り組みやすい言語](#)として人気があります。

本書はJavaScriptの新しい仕様「ES6」に準拠した書き方をもとに、プログラミングがまったく初めての人でも理解しながら楽しく進められる構成になっています。

本書の全体の流れを簡単に紹介しておきます。

Chapter 1～Chapter 4ではJavaScriptの[基本的な文法](#)を紹介しています。ここまででひととおりコードを読んで書ける基礎を習得しましょう。

Chapter 5では、HTML要素を書き換えてページの表示を変える、JavaScriptの特徴的な機能のひとつである「[DOM操作](#)」に触れます。

Chapter 6はChapter 5の延長で、DOM操作に特化したライブラリ「[jQuery](#)」を使って簡単なUI部品を作成します。

Chapter 7ではDOM操作と並ぶJavaScriptの主要な機能、[非同期通信（Ajax）](#)を取り上げます。少し高度なトピックではありますが、現代的なWebサイトの開発には必要不可欠な知識です。

最後のChapter 8では[トランスパイル](#)に入門します。[NodeJS](#)、[npm](#)を使った外部プログラムのインストール方法から設定ファイルの編集まで、トランスパイルに必要な作業をひととおり取り上げます。

解説は可能な限り[ていねい](#)にしたつもりです。その一方で、実践的な応用力をいちはやく身につけられるように、ある程度幅広いトピックを取り上げています。本書が読者のみなさまのお力になれることを願っています。

本書はたくさんの方のお力添えで完成しました。編集者の木津 滋、粕川 満両氏には大変お世話になりました。弊社スタッフ、そして家族に。この場を借りて厚く御礼申し上げます。

株式会社Studio947
狩野 祐東

本書の使い方

本書は別途配布されるサンプルプログラムを使って、JavaScript の学習を進めてゆきます。

サンプルプログラムは出版元の Web ページに掲載されているリンク、または次の URL から直接入手することができます。Windows などから警告が出されることもあります。そのままダウンロードを継続してください。

<http://www.socym.co.jp/download/1124/sample.zip>

●実行するとこんな感じ

ここで、おおまかな内容をつかむために、実行しているところをさらっと見てみます。

実行するとこんな感じ

「入室」ボタンをクリックしてしばらく待ってから「退室」ボタンをクリックすると、入室時間から退室時間までの経過時間を計算して、「利用時間」として表示します。7-1「日時を取得しよう～Date オブジェクト～」(p.194) のサンプルプログラムを改造して、経過時間計算機能を追加します。



「入室」ボタン、「退室」ボタンをクリックすると経過時間が表示される

●何をやっているの？

実行しているところをさらっと見たら、次にやっていることの流れをつかみます。

なにをやっているの？

「退室」ボタンをクリックしたときの時間から「入室」ボタンをクリックしたときの時間を引いて、経過時間を求めます。プログラミングの観点からいえば、2つの Date オブジェクトの差を求めて、その結果をページに表示することになります。

2つの Date オブジェクトの差を求めると、その結果はミリ秒で算出されます。それを人間にわかりやすいように「分」とか「秒」にするためには、さらにちょっとした計算をしなければなりません。

JavaScript では、四則演算以外の計算処理をするには「Math オブジェクト」というオブジェクトを使用しま

●使用する機能やツール

サンプルプログラムをつくるために必要な、機能やツールを箇条書きでまとめています。

使用する機能やツール

- Date オブジェクト
- Date オブジェクトの各種メソッド
- Math.ceil() メソッド
- 配列
- JQuery の各種メソッド

● やることをまとめると…

機能やツールを使ってどんなことを学ぶのかを箇条書きでまとめています。

- ☑ Date オブジェクトを使った時間の計算方法
- ☑ ミリ秒の数値を分数に換算
- ☑ Math オブジェクトを使った計算
- ☑ 小数点以下の切り上げ



やることを
まとめるよ…

- Chapter 1
- Chapter 2
- Chapter 3
- Chapter 4
- Chapter 5
- Chapter 6
- Chapter 7
- Chapter 8

● それではやってみよう！

ここからが本番。サンプルプログラムについてじっくりと解説します。



やるぞ！

それではやってみよう！

基本的なプログラムの処理の流れや考え方

【入室】 ボタンや【退室】 ボタンをクリックしたらそのときの日時を取得しますが、経過時間を計算できるように、両方の日時を Date オブジェクトのままで保存しておくことにします。2つの Date オブジェクトは配列に保存します。

【入室】 ボタンと【退室】 ボタン、どちらもクリックされたときの時間を保存することに変わりはありませんが、それぞれの処理は少しずつ異なります。詳しく見てみることにしましょう。

【入室】 ボタンがクリックされたときの処理

【入室】 ボタンがクリックされたらまず、入室時間と退室時間の Date オブジェクトを保存しておく配列 times を作成（宣言）します。そして、その配列 times に、入室時間の Date オブジェクトを保存します。それと同時に、ページに表示する処理もおこないます。7-1「日時を取得しよう～Date オブジェクト～」

- Chapter 1
- Chapter 2
- Chapter 3
- Chapter 4
- Chapter 5
- Chapter 6
- Chapter 7
- Chapter 8

● プログラムを書いてみよう

これまでの学びを活かして、実習ファイルを編集し、完成データを作りあげてみましょう。



書くぞ！

プログラムを書いてみよう

それでは、経過時間を計るサンプルプログラムを作ってみましょう。今回のサンプルプログラムはプログラム量が多いように見えますが、7-1「日時を取得しよう」(p.194) のサンプルプログラムに数行追加する程度です。

HTML ファイル

[\[実習ファイル\] practices/c07/7-02-calc/index.html](#)
[\[完成データ\] samples/c07/7-02-calc/index.html](#)

```
<body>
...
<main>
  <div class="container">

    <!-- メインコンテンツ -->
    <h2>入室退室管理システム</h2>
    <p><button id="enter">入室</button><span id="enter-time"></span></p>
    <p><button id="exit">退室</button><span id="exit-time"></span></p>
    <p>利用時間 : <span id="enter-exit"></span></p>
    <!-- メインコンテンツ -->

  </div>
</main>
...
<script src="scripts/jquery-3.3.1.min.js"></script>
<script src="scripts/script.js"></script>
</body>
```

- Chapter 1
- Chapter 2
- Chapter 3
- Chapter 4
- Chapter 5
- Chapter 6
- Chapter 7
- Chapter 8

なお、本書中でもサンプルプログラムを掲載しておりますが、紙面の都合上、学習に差し支えない範囲で一部を省略していることがあります。予めご了承ください。

CONTENTS

知識ゼロからはじめる

ゆっくり・ていねい JavaScript

ES6対応

かのうすけはる
狩野祐東[著]

〔目次〕

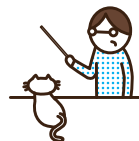
■ はじめに	3
■ 本書の使い方	4

Chapter 1

イントロダクション

JavaScript はブラウザで動作するプログラミング言語です。実際にプログラミングを始める前に、JavaScript がどんなプログラミング言語で、どういうことができるのか、Chapter 1 で大まかに把握することにしましょう。また、JavaScript プログラムの編集や実行に必要なアプリケーションも準備しましょう。

1-1 JavaScript ってどんな言語？	12
1-2 JavaScriptの2大機能	15
1-3 新しいJavaScript ～ ES6とは～	20
1-4 実習を始めましょう	22



Chapter 2 JavaScriptの基本的な文法

Chapter 2 では、ごく簡単な機能を使った実際に動作するプログラムを作ります。どのプログラムも数行で完成するわかりやすいものなので、はじめて取り組む方も心配はいりません。実習をとおして、JavaScript プログラムの基本的な文法や、正しく動作させるための「書き方のルール」をマスターしましょう。

2-1	プログラムを書いてみよう～ window.alert ～	26
2-2	プログラムを書く場所 ～ <script> ～	32
2-3	コンソールを使ってみよう ～ console.log ～	37
2-4	大事な機能を知ろう ～コメント文と'use strict'～	42



Chapter 3 基本的なプログラミング

Chapter 3 では JavaScript の基本的な機能を取り上げます。値を保存しておく変数や配列、数値の簡単な計算など、プログラミングに欠かせない重要な知識を紹介します。

3-1	データを一時的に保存する ～変数～	48
3-2	テキストを操作する① ～文字列～	54
3-3	テキストを操作する② ～テンプレート文字列～ ES6	59
3-4	簡単な計算をする ～数値、算術演算子～	64



3-5	データをひとつにまとめる① ～配列～	74
3-6	データをひとつにまとめる② ～オブジェクト～	79

Chapter 4 条件文、繰り返し、ファンクション

Chapter 4 で取り上げる条件文、繰り返し、ファンクションは、状況によって処理を切り替えたり、同じ処理を繰り返すようなときに使います。また、繰り返しやファンクションには ES6 で新たに導入された機能も多く、要注目です。

4-1	状況によって処理を振り分ける ～ if ～	88
4-2	より高度な処理の振り分け ～ else if と各種演算子～	95
4-3	新しい変数宣言の方法 ～ let と const ～ ES6	105
4-4	同じ処理を連続して繰り返す ～ while ～	110
4-5	一連の処理をまとめて呼び出す ～ファンクション～	117
4-6	新しいファンクションの定義法 ～アロー式～ ES6	127
4-7	配列に使える新しい繰り返し ～ for...of ～ ES6	133
4-8	データの順序を入れ替える ～配列の操作～	139



Chapter 5

ブラウザへの表示

JavaScript で HTML を書き換えて、ブラウザに表示されているページの一部の表示を更新するテクニックを「DOM 操作」といいます。DOM 操作は JavaScript プログラミングの中でも重要なトピックのひとつ。Chapter 5 では、イベントの設定と、タグの属性、とくに class 属性の書き換えを中心に紹介します。

5-1 トグルボタンの作成 ～ DOM操作とイベント～

152

5-2 フォームの入力内容を調べる ～フォームのDOM操作～

162



Chapter 6

jQueryでユーザーインターフェースを簡単に作成

JavaScript でよく使われるライブラリのひとつに「jQuery」があります。Chapter 6 では、画像ギャラリーとタブという、Web ページでよく使われるユーザーインターフェースを作成してみましょう。

6-1 画像ギャラリーの作成 ～ jQuery の基礎～

172

6-2 タブUIの作成 ～別の要素を操作する～

185



Chapter 7

組み込みオブジェクトと非同期通信

Chapter 7 では、より実践的なプログラム作りに役立つテクニックを紹介します。これまで取り上げていない 2 つのオブジェクト—— Date オブジェクトと Math オブジェクト——を使って、日時の取得や日時の計算をしてみましょう。

また、ほかの Web サイトからデータを取得する「非同期通信」という技術を利用して、郵便番号検索フォームも作成してみます。

7-1 日時を取得しよう ～ Dateオブジェクト～

194

7-2 経過時間を表示しよう ～ Mathオブジェクト～

204

7-3 Web APIを使った住所検索 ～ Yahoo! デベロッパーネットワーク～

212



Chapter 8

Node.js、npm、トランスパイル

ES6 で書かれたプログラムは、IE11 などの古いブラウザでは動作しません。多くのブラウザで動作するようにするためには「トランスパイル」という処理をする必要があります。Chapter 8 では、作成したプログラムをトランスパイルする方法を紹介します。

8-1 トランスパイルとは ～ ES6をES5に変換～

234

8-2 Node.js、npmとは

236

8-3 トランスパイルに挑戦

239



索引

251



Chapter 1

イントロダクション

JavaScript はブラウザで動作するプログラミング言語です。実際にプログラミングを始める前に、JavaScript がどんなプログラミング言語で、どういうことができるのか、Chapter 1 で大まかに把握することにしましょう。また、JavaScript プログラムの編集や実行に必要なアプリケーションも準備しましょう。

JavaScriptって どんな言語？

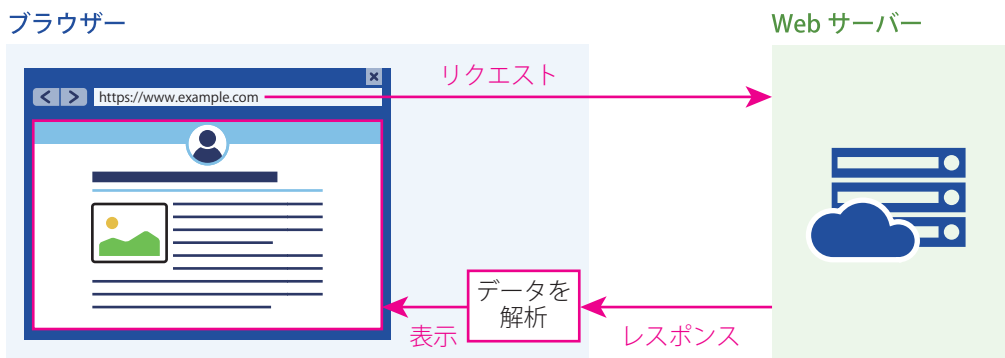
JavaScript はブラウザーに搭載された、ブラウザーと Web ページを操作するために作られたプログラミング言語です。ここでは、JavaScript の大まかな機能を把握しましょう。

JavaScript はブラウザーに搭載されたプログラミング言語

JavaScript はブラウザーに組み込まれたプログラミング言語で、ブラウザーそのもの、または表示されている Web ページを操作するのに使われます。JavaScript を使ったらどういうことができるかを、ブラウザーが Web ページを表示するまでの流れを通して見ていきましょう。

Web ページを表示する際、ブラウザーはまず URL で指定されたデータ——ここでいうデータには、HTML ファイル、CSS ファイル、画像ファイル、そして JavaScript ファイルなどが含まれます——を Web サーバーに「リクエスト」します。

Web サーバーは、リクエストされたデータをブラウザーに返します。このデータを返す処理のことを「レスポンス」といいますが、ブラウザーはレスポンスを受け取ると、そのデータをウィンドウに表示します。この、「リクエストする」「レスポンスを受け取って表示する」という 2 つの処理が、ブラウザーのおもな役割です。



ブラウザーの基本的な機能と、Web ページが表示されるまでの流れ

しかし、Web サーバーから返ってきた HTML や CSS は、そのままブラウザのウィンドウに表示できるわけではありません。ブラウザは、HTML や CSS に書かれている内容を解析してウィンドウに表示します。この解析処理のことを「パース」と呼びます。

ところで Web ページの表示にはもうひとつ大事なポイントがあります。それは「リクエスト→パース→表示」の処理が完了して、ブラウザにページが表示されてしまったら、再読み込みしないかぎりもうそのページが変化することはない、ということです。Web サーバーからダウンロードしてきた HTML や CSS は、ダウンロードしたときの状態から変わることがないのです。

ここで JavaScript が登場します。JavaScript を使うと、すでに Web サーバーからダウンロードされ、パースされ、表示もされている HTML や CSS を、プログラミングによって「書き換える」ことができます。

JavaScript でできること

JavaScript の大きな機能のひとつは、すでに表示されている HTML や CSS を書き換えることです。

具体的には、たとえば、 タグの src 属性を書き換えたり、 タグをもう 1 個追加したり削除したり、あるいは <div> ~ </div> に含まれるすべてのコンテンツを丸ごと書き換えたりというようなことが、JavaScript を使えばできます。そして、ブラウザは、すでに表示している HTML や CSS であっても、その一部が JavaScript によって書き換えられると、瞬時に、新しくなった HTML や CSS で Web ページを描き直します。つまり、JavaScript を使ってプログラミングをしておくと、再読み込みすることなしに、Web ページの一部を書き換えることができるようになるのです。



HTML や CSS を書き換える例。ドロップダウンメニューなども JavaScript で作られている

ブラウザーそのものを操作する

JavaScript にはそれ以外にもたくさん機能があります。

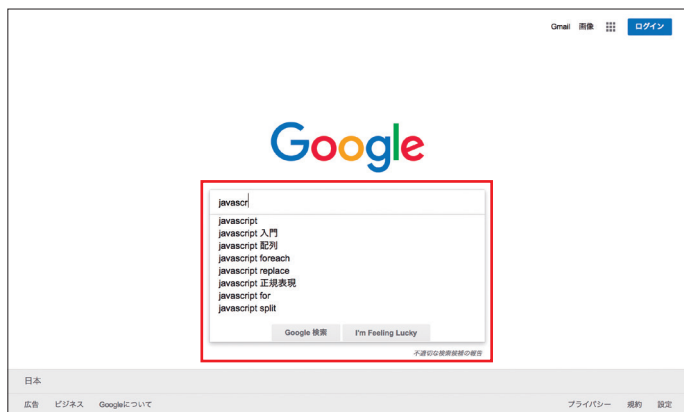
JavaScript は、ブラウザーウィンドウの幅や高さを調整したり、新たなウィンドウを開いたり、あるいはダイアログを表示させたりと、ブラウザーそのものを操作することもできます。



ブラウザーそのものを表示する例。ダイアログボックスを表示する

HTML や CSS とは別に、あとから追加のデータを取得する

非同期通信^[1]と呼ばれる技術を使って、ブラウザーと Web サーバーとのあいだでデータを送受信することで、新たなコンテンツをページに表示させるようなこともできます。



非同期通信の例。入力が終わらないうちに検索語を Web サーバーに問い合わせ「よく検索されているワード」を取得して表示している

[1] XHR、Ajax と呼ばれることもあります。

JavaScriptの2大機能

JavaScriptにはたくさんの機能がありますが、その中でもとくに重要なのが「DOM 操作」と「非同期通信」です。DOM 操作や非同期通信とはどういうものなのか見ていきましょう。

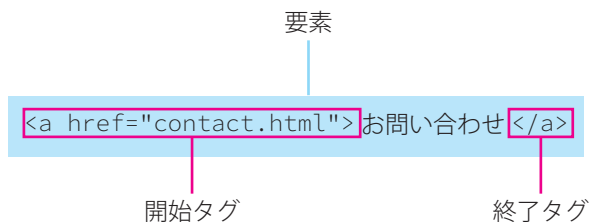
DOM 操作

「JavaScript でできること」でも簡単に紹介しましたが、JavaScript では、表示されている Web ページの HTML を書き換えることができます。この書き換え操作のことを「DOM 操作」といいます。JavaScript でプログラミングをするとき、その処理の多くが DOM 操作だといってもよいくらい、重要な機能です。

DOM 操作の最大の特徴は「HTML を書き換えると、その変更内容が瞬時にページの表示に反映されること」です。そのため、ブラウザの再読み込みボタンをクリックしなくても、いま表示されているページをどんどん書き換えることができます。具体的にはどういう処理ができるのか、少し見てみましょう。

HTML 要素を追加、削除する

HTML に含まれるタグと、開始タグ・終了タグに囲まれた部分のことをまとめて「要素」と呼びます。



要素の例

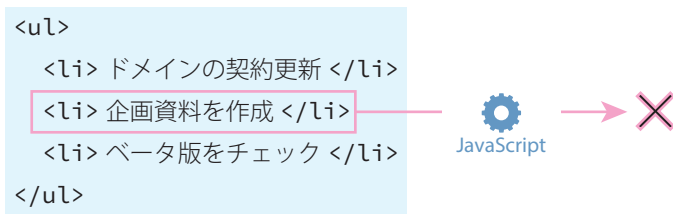
JavaScript を使うと、もともとの HTML にはない要素を追加したり、すでにある要素を削除したりすることができます。

要素を追加、削除するという処理を通して、たとえば「ToDo リスト」のようなものを作れます。テキストフィールドにテキストを入力して「追加」ボタンをクリックしたら ToDo 項目が増えるとか、それぞれの ToDo 項目のチェックボックスにチェックを付けると削除されるとか、そうした処理ができます。

要素を追加する



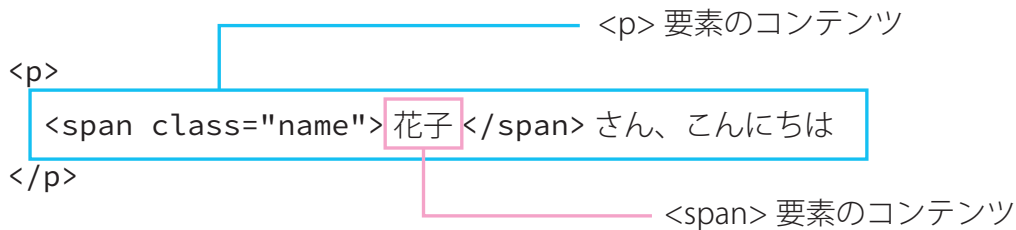
要素を削除する



要素を追加、削除する例

要素のコンテンツを書き換える

コンテンツとは、HTML の開始タグと終了タグに囲まれた、テキストやほかのタグ（要素）のことです。



「コンテンツ」とはタグに囲まれた部分のこと

JavaScript を使うと、コンテンツに含まれるテキストを書き換えることができます。たとえば、ログインが必要なページなどで、そのログイン名に合わせて「〇〇さん、こんにちは」と表示したり、状況に応じて文面を変えたりすることができます。

```
<span class="name">Guest</span> さん、こんにちは
```

花子



ログインしていれば「ログイン名」に
書き換える

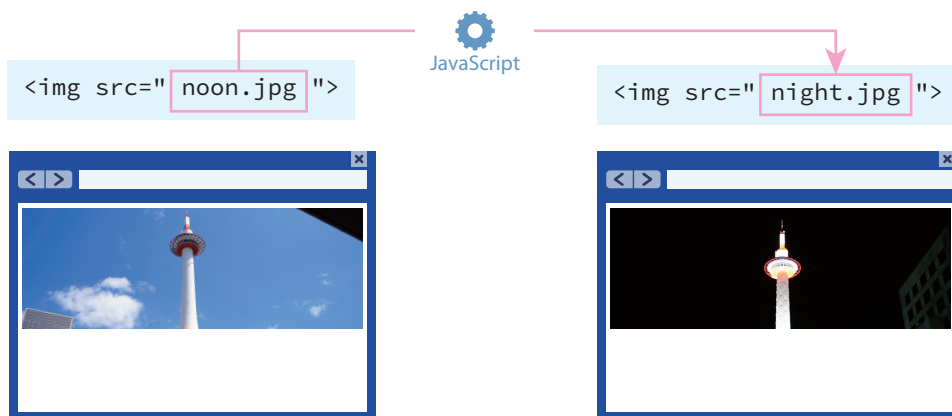


```
<span class="name">花子</span> さん、こんにちは
```

要素のコンテンツを書き換える例

タグの属性を書き換える

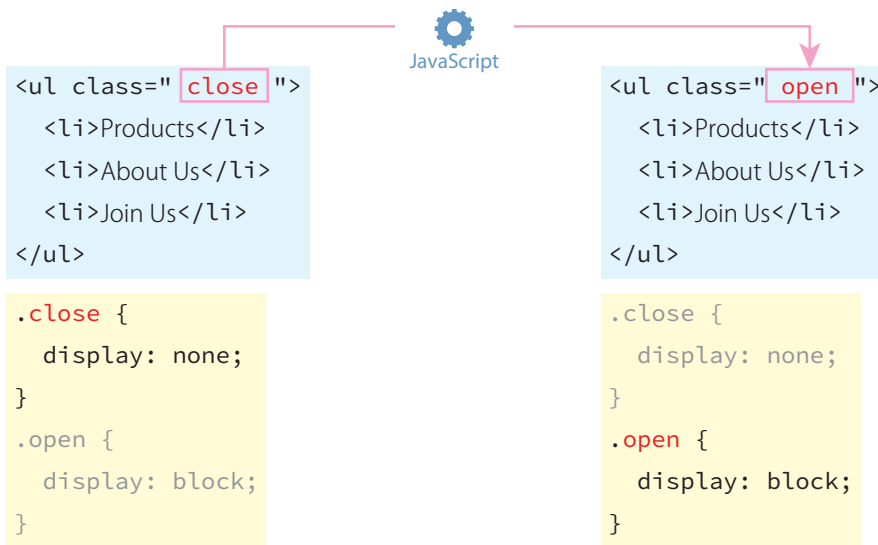
タグの属性を書き換えることもできます。属性には、<a> タグなら href 属性、 タグなら src 属性などたくさんありますが、JavaScript を使えばどんな属性でも書き換えることができます。属性を書き換えるという処理を通じて、たとえば タグの src 属性を操作すれば表示する画像を切り替えることができます。



 タグの src 属性を書き換えれば、表示される画像が切り替わる

CSS を書き換える、または適用されるスタイルを変更できる

要素に適用されている CSS のスタイルを書き換えることもできます。また、「タグの属性を書き換える」を応用して、class 属性を書き換えれば、適用されている CSS を入れ替えることができます。たとえば、クリックしたらメニューを表示、もう一度クリックするとメニューが非表示になるようなユーザーインターフェースを作る場合、class 属性を書き換えて、適用される「display: block;」と「display: none;」を入れ替えるようなことをします。

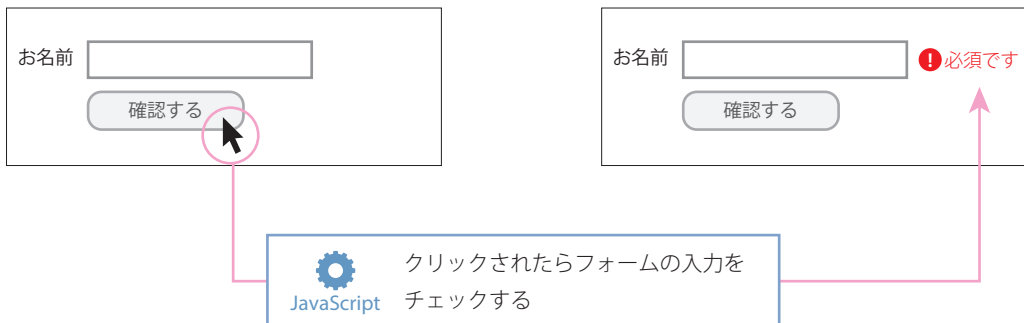


HTML の class 属性を書き換えて、適用される CSS を切り替える

イベントを設定する

「ボタンをクリックしたら～する」、「ページが読み込まれたら～する」など、JavaScript は基本的になにかが起きたときに、プログラムに書かれた処理を実行します。具体的には「あるリンクの `<a>` タグをクリックされたら、ある `<img>` タグの `src` 属性を書き換える（結果的に表示される画像が切り替わる）」というようなプログラムを書きます。この、「〇〇が起きたら」という、処理が実行されるきっかけのことを「イベント」といいます。

JavaScript にはたくさんのイベントが用意されています。イベントには、クリックしたときに処理を実行するイベントや、HTML がダウンロードされ、完全に読み込まれたときに発生するイベントなどがあります。



イベントの例。ボタンがクリックされたらフォームの入力内容を確認する

非同期通信

JavaScript には DOM 操作のほかにもうひとつ、重要な機能があります。それが「非同期通信」です。非同期通信は近年、非常に多くのサイトで使われています。

非同期通信とは

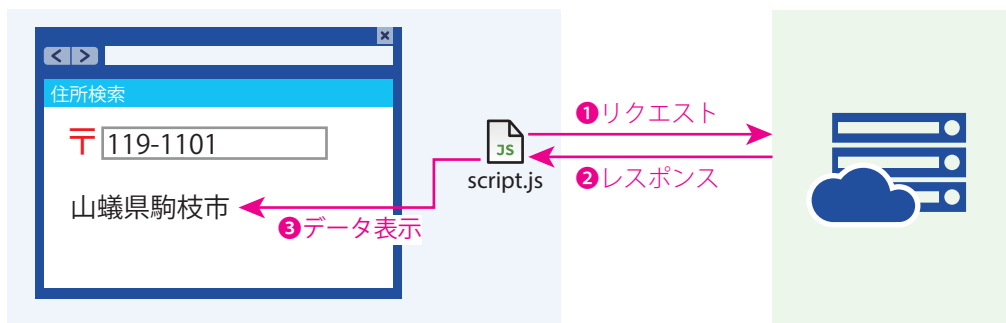
非同期通信とは、通常ブラウザ自体がおこなう Web サーバーへのリクエストとデータの受け取り（レスポンス）を、JavaScript が代わりにおこなう通信のことです。「Ajax」や「XHR」と呼ばれることもあり、こちらの方が一般的によく聞くかもしれません。ブラウザ自身がリクエストしないため再読み込みが発生せず、表示されているページ全体が入れ替わることがありません。

非同期通信は、一般に DOM 操作と組み合わせて、ページ全体を再読み込みすることなしに、一部のコンテンツを書き換えるのに使われます。

非同期通信は、本書の Chapter 7「組み込みオブジェクトと非同期通信」(p.193) で取り上げます。

ブラウザ

Web サーバー



非同期通信のイメージ。郵便番号を入力すると、全体を再読み込みせずに住所が表示される

新しいJavaScript ～ES6とは～

JavaScript は 2015 年に大幅なバージョンアップをしました。ここでは、2015 年以降の新しい JavaScript の特徴と、それ以前との違いを紹介します。

JavaScript は古くて新しい言語

JavaScript は現在多くの Web サイトで使われていて、非常にポピュラーなプログラミング言語です。より多くの場面で使われるようになったこともあって、JavaScript で作られるプログラムも年々大規模化しています。

ただ、もともとの JavaScript には大規模なプログラムを効率的に書ける機能や仕組みがありませんでした。そこで 2015 年、大々的な機能強化がおこなわれることになりました。それが「ES6」と呼ばれる新しい JavaScript です。

JavaScript の言語仕様を策定している団体

JavaScript は、ECMA International（エクマ・インターナショナル）という標準化団体が言語仕様を策定しています。JavaScript の正式名称は「ECMAScript」と呼ばれていて、現在普及している主要なブラウザ——Chrome、Edge、Firefox、Safari——は、この ECMAScript の仕様に忠実に則って開発が進められています。

ECMAScript は、2015 年以降、毎年仕様を改訂する計画を立てています。そして、その年に決まった仕様の JavaScript を「ES2015」「ES2016」「ES2017」と呼んでいます。「ES6」とは、これら 2015 年以降に作られた仕様をひとまとめにして呼ぶときの通称で、「ES5 以降の JavaScript」という意味の言葉だと考えておけば問題ありません。

ちなみに 2015 年より前の JavaScript は、「ECMAScript バージョン 5」と呼ばれていました。本書では、古い JavaScript を「ES5」、新しい JavaScript を「ES6」と呼んで区別しています。

ES6 とは

ES6 は、ES5 の弱点を克服して、より現代的なプログラミングができるようにしたバージョンです。その結果、プログラムが書きやすくなり、プログラムが大規模になっても管理しやすくなりました。しかし、大規模なプログラムを書かなくとも ES6 を使うメリットがあります。

あまり詳しく知る必要はありませんが、以前のバージョンの JavaScript (ES5) には、ほかのプログラミング言語にはない独特な仕様がいくつかあり、ほかの言語を経験した人にとっては違和感のある動作をするところがありました。

また、大規模なプログラムを書くための機能が用意されていなかったことで、複雑なものを作るには工夫をする必要がありました。でも、そういう作業は、本来作るべきものを作る作業とは違いますから、あまり生産的とはいえません。

ES6 で新しく導入された機能の多くは「JavaScript をより使いやすくする」ためのものです。独特な仕様に遭遇することも少なくなるので、初心者でも経験者でも、よりプログラミングしやすい環境になっています。

これから始めるなら ES6

すでに多くのブラウザが ES6 に対応しているので、これから JavaScript を始めるという方は、迷わず ES6 を使いましょう。

ただし、IE11 は ES6 にほとんど対応していないため、そのままではほとんどのプログラムが動作しません。

そこで、公開する Web サイト用のプログラムを作るときは、いまのところ「トランスパイル」という、ES6 のプログラムを ES5 のプログラムに書き換える処理をして IE11 でも動くようにするのが一般的です。トランスパイルについて詳しくは Chapter 8 「Node.js、npm、トランスパイル」(p.233) で紹介します。

IE11 を除けば、現在よく使われる主要なブラウザは ES2015 の機能をほぼ完全にサポートしています。しかし、前述のとおり JavaScript の仕様自体が毎年アップデートされることになっているので、ES2015 より後に導入された機能についてはブラウザが対応しきれていない部分もあります。各ブラウザの対応状況を知りたいときは、次のサイトが役に立ちます。

● ECMAScript 6 compatibility table

<https://kangax.github.io/compat-table/es6/>

もう少し大雑把に調べたいときは、次のサイトも役に立つでしょう。この「Can I Use」では、JavaScript だけでなく CSS や HTML の新機能の対応状況も調べることができます。

● Can I Use

<https://caniuse.com>

実習を始めましょう

本書で紹介するサンプルプログラムを実際を書いてみたり、実行して動作を確認したりするためには、ブラウザとテキストエディタが必要です。これらの道具を用意して、Chapter 2 からの実習に取り組みましょう。

学習に必要なアプリケーション

本書では、完成プログラムのほかに、実習のためのファイルを用意しました。この実習ファイルを使って、自分でプログラミングを体験してください。

実習を始めるにあたり、必要なアプリケーションを紹介しておきます。JavaScript プログラミングをするには、最低限、ブラウザと、プログラムを入力するためのテキストエディタが必要です。

ブラウザ

ブラウザは、Chrome、Firefox、Edge (Windows 10)、Safari (macOS) のいずれかを使用します。これらのブラウザであればどれもかまいませんが、必ず最新版を使うようにしましょう。なお、ES6 に対応していない IE は実習に使えませんので、Windows 7、8 をお使いの方は、Chrome か Firefox をインストールします。

● ブラウザーのダウンロード

	ブラウザ名	Web サイト
	Google Chrome	https://www.google.co.jp/chrome/browser/desktop/index.html
	Mozilla Firefox	https://www.mozilla.org/ja/firefox/

By Google - Google Play Store, パブリック・ドメイン, <https://commons.wikimedia.org/w/index.php?curid=46544760>
By The Mozilla Foundation - <http://design.firefox.com/photon/visuals/product-identity-assets.html><https://github.com/FirefoxUX/product-identity>, CC 表示 3.0, <https://commons.wikimedia.org/w/index.php?curid=62823019>

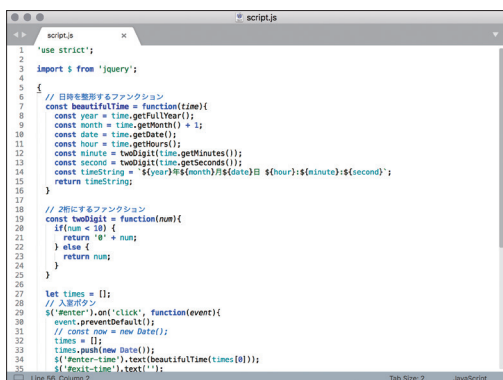
テキストエディタ

プログラムを入力するために、テキストエディタも用意しましょう。Windows にははじめからインストールされているノートパッドや、Word などのワープロソフトは使えません。また、Windows のメモ帳、Mac のテキストエディットも、まれにプログラムが動作しなかったり、プログラミングに使うには設定が必要だったりするのでおすすめしません。

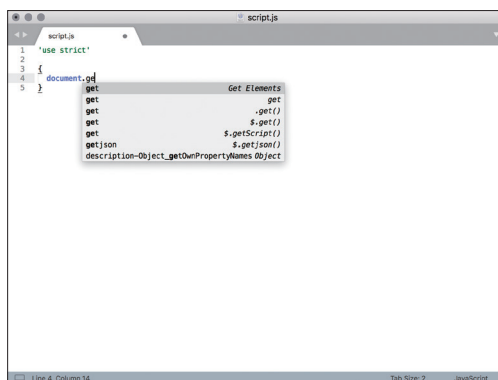
そのようなわけで、テキストエディタはダウンロードしてインストールする必要があります。普段使っているものがあればそれでかまいません。もしこれからダウンロードするのであれば、コードの色分け機能があるものを選びましょう。コードの色分け機能とは、プログラムコードに書かれた単語の種類によって自動的にテキストに色が付くものです。

また、コードの自動補完機能がついているテキストエディタもあります。自動補完機能とは、コードを途中で書くとその続きを自動的に入力してくれたり、選択肢を提示してくれたりする機能です。すべてのテキストエディタに自動補完機能がついているわけではなく、また絶対に必要な機能というわけでもありませんが、あれば便利です。

コードの色分け機能



自動補完機能



コードの色分け機能と自動補完機能

次のページの表で JavaScript プログラミングをするのにおすすめのテキストエディタをいくつか紹介しておきます。テキストエディタを使ったことがない方は、この中から選んでおけばまず問題はありません。どれもコードの色分け機能、自動補完機能がついています。

なお、表で紹介しているテキストエディタは、Windows、Mac 両方に対応しています。

● おすすめのテキストエディタ

アプリケーション名	有料・無料	URL	特徴
Brackets	無料	http://brackets.io	Adobe 社が中心となって開発を進めているオープンソースのテキストエディタ。シンプルで初心者やプログラマーでない Web デザイナーなどでも使いやすい
ATOM	無料	https://atom.io	GitHub 社が中心となって開発を進めているオープンソースのテキストエディタ。プログラマーの間で人気がある
Visual Studio Code	無料	https://www.microsoft.com/ja-jp/dev/products/code-vs.aspx	マイクロソフト社の高機能テキストエディタ。プログラマーの間で人気がある。Windows 版だけでなく Mac 版もある
Sublime Text 3	有料（試用期間あり）	https://www.sublimetext.com	高機能テキストエディタ。プログラマーの間で人気がある。プラグインで機能を拡張できる。日本語の扱いが不得意な点がマイナス。有料だが試用期間あり

```

1 'use strict';
2
3 {
4   // 日時を整形する関数
5   const beautifulTime = function(time) {
6     const year = time.getFullYear();
7     const month = time.getMonth() + 1;
8     const date = time.getDate();
9     const hour = time.getHours();
10    const minute = twodigit(time.getMinutes());
11    const second = twodigit(time.getSeconds());
12    const timeString = `${year}/${month}/${date} ${hour}:${minute}:${second}`;
13    return timeString;
14  }
15
16  // 2桁にする関数
17  const twodigit = function(num) {
18    if(num < 10) {
19      return '0' + num;
20    } else {
21      return num;
22    }
23  }
24
25  let times = [];
26  // 入出力
27  $('#enter').on('click', function(event) {
28    event.preventDefault();
29    // const now = new Date();
30    times = [];
31    times.push(new Date());
32    $('#enter-time').text(`${beautifulTime(times[0])}`);
33    $('#enter-time').text('');
34    $('#enter-exit').text('');
35  });

```

Sublime Text (Mac)

```

1 'use strict';
2
3 {
4   // 日時を整形する関数
5   const beautifulTime = function(time) {
6     const year = time.getFullYear();
7     const month = time.getMonth() + 1;
8     const date = time.getDate();
9     const hour = time.getHours();
10    const minute = twodigit(time.getMinutes());
11    const second = twodigit(time.getSeconds());
12    const timeString = `${year}/${month}/${date} ${hour}:${minute}:${second}`;
13    return timeString;
14  }
15
16  // 2桁にする関数
17  const twodigit = function(num) {
18    if(num < 10) {
19      return '0' + num;
20    } else {
21      return num;
22    }
23  }
24
25  let times = [];
26  // 入出力
27  $('#enter').on('click', function(event) {
28    event.preventDefault();
29    // const now = new Date();
30    times = [];
31    times.push(new Date());
32    $('#enter-time').text(`${beautifulTime(times[0])}`);
33    $('#enter-time').text('');
34    $('#enter-exit').text('');
35  });

```

Visual Studio Code (Windows)

索引

[凡例]

Ⓐ 同義語であることを意味しています。

jQuery jQuery のメソッドです。

ES6 ES6 で使える機能を意味しています。

Math Math オブジェクトのメソッドです。

window window オブジェクトのメソッドです。

配列 配列のメソッドです。

Date Date オブジェクトのメソッドです。

String String オブジェクトのメソッドです。

■ 記号

'	55
-	65
!	102
!==	102
"	55
\$() メソッド jQuery	174
\$(this) jQuery	182
\$.ajax メソッド jQuery	222
\${ } ES6	62
%	65
%s	40
&&	102
*	65
** ES6	68
,	40
/	65
;	28
`	62
	102
¥、/	56
¥n	61
+	
足し算	65
文字列の連結	57
+=	114
<	102
<=	102
<form> タグ	164
<option> タグ	165
<script> タグ	33
=	51
===	100, 102
>	102
>=	102

■ A

abs メソッド Math	208
---------------	-----

addClass メソッド jQuery	189
Ajax (Ⓐ非同期通信/ⒶXHR)	19, 173, 214
alert メソッド window	27, 29, 89
append メソッド jQuery	175
ATOM	24
attr メソッド jQuery	179

■ B

Babel	234
Boolean 型	72
Brackets	24

■ C

ceil メソッド Math	208
classList.add メソッド	158
classList.contains メソッド	158
classList.remove メソッド	158
classList.toggle メソッド	158
classList プロパティ	
取得した要素の一	158
clearInterval メソッド	145
concat メソッド 配列	143
confirm メソッド window	89
console (Ⓐコンソール)	38
console.log メソッド	37, 39
const ES6	108
cross-env パッケージ	246

■ D

data-* 属性	180
data メソッド jQuery	180
Date オブジェクト	97, 195
document.getElementById メソッド	69
DOM 操作	15, 153, 173

■ E

ECMAScript	20
else	92

else if	101
encodeURIComponent	225
ES2015	20
ES6	20, 105, 106, 234
event.currentTarget プロパティ	158, 181
event.preventDefault メソッド	157, 166

F

false (偽)	91
floor メソッド Math	208
for...of 文	134
forEach メソッド 配列	143
function	119, 182

G

getDate メソッド Date	199
getDay メソッド Date	199
getFullYear メソッド Date	199
getHours メソッド Date	98, 199
getMilliseconds メソッド Date	199
getMinutes メソッド Date	199
getMonth メソッド Date	199
getSeconds メソッド Date	199
getTime メソッド Date	199
Google Chrome	22

H

HTML5	180
HTTP リクエスト	213

I

id 属性	164
IE11	239
if 文	92
import 文	242
innerHTML プロパティ	84

J

join メソッド 配列	143
jQuery	173, 245
—オブジェクト	181
—を使う準備	176
JSON	214
JSONP	225

L

laravel-mix パッケージ	246
-------------------------	-----

length プロパティ String	112, 124
let ES6	106

M

Math.round メソッド	68
Math オブジェクト	68, 208
max メソッド Math	208
min メソッド Math	208
Mozilla Firefox	22

N

name 属性	164
new	97, 198
Node.js	236
node_modules フォルダ	245
npm	238, 245
Number 型	72

O

Object 型	73
onclick イベント	156
onsubmit イベント	165
on メソッド jQuery	176, 181

P

package.json ファイル	245
package-lock.json ファイル	245
pop メソッド 配列	141
pow メソッド Math	208
push メソッド 配列	141, 207

R

random メソッド Math	208
removeClass メソッド jQuery	189
return	119
reverse メソッド 配列	143
round メソッド Math	208

S

Safari	39
setDate メソッド Date	199
setFullYear メソッド Date	199
setHours メソッド Date	100, 199
setIntervald メソッド	145
setMinutes メソッド Date	199
setMonth メソッド Date	199
setSeconds メソッド Date	199

shift メソッド 配列	142
split メソッド String	143
String 型	72
Sublime Text 3	24
substr メソッド String	112

■ T

textContent プロパティ	70, 84
text メソッド jQuery	175, 202
toString メソッド 配列	142
true (Ⓓ真)	91

■ U

unshift メソッド 配列	142
URL エンコード	225
'use strict'	44
UTC (Ⓓ協定世界時)	199
UTF-8	28

■ V

value 属性	165
value プロパティ	166
var	50
Visual Studio Code	24

■ W

webpack.mix.js ファイル	246
Web インスペクタ	38
while 文	111, 134
window オブジェクト	30

■ X

XHR (Ⓓ非同期通信／ⒹAjax)	19, 173, 214
XML	214

■ Y

Yahoo! JAPAN ID	215
Yahoo! デベロッパーネットワーク	216
YOLP	212

■ あ行

アプリケーション登録	215
余り	
割り算の一	65
アラートダイアログ	89
アロー式	128, 182
一の省略	129, 130

イベント	18, 155
イベントハンドラ	156
入れ子	44
インクリメンタルサーチ	214
インターバル ID	145
インデックス (Ⓓ添え字)	
配列の一	76
エスケープ	56, 61
演算子	65
オブジェクト	27, 29, 80

■ か行

改行	61
開発ツール	38
外部 JavaScript ファイル	33, 35
返り値	90, 120
確認ダイアログ	89
掛け算	65
画像ギャラリー	178
偽 (Ⓓfalse)	91
協定世界時 (ⒹUTC)	199
クライアント ID	218
誤差	67
コマンドプロンプト	238
コメントアウト	45
コメント文	43
子要素	
一を削除する	154, 175
一を挿入する	154, 175
コンソール (Ⓓconsole)	
.....	37, 38, 44, 66, 228
コンテンツ	
HTML 要素の一	16

■ さ行

最小値	208
最大値	208
三項演算子	130
参照	
変数の一	51
時刻	97
四捨五入	208
指数演算子 ES6	68
四則演算子	65
条件式	100
省略	30, 93
真 (Ⓓtrue)	91

数値	65, 72
—を丸める	68
スコープ	107
ストリクトモード	44
生成	198, 208
絶対値	208
宣言	
オブジェクトの—	81
定数の—	108
配列の—	76
変数の—	50, 105, 106
添え字 (Ⓜインデックス)	76
属性	17
—セクタ	57
—の値を書き換える	179

■た行

ターミナル	238
ダイアログボックス	26
代入	51
タイマー	145
足し算	65
定義	
ファンクションの—	119, 122
ファンクションの— ES6	128
定数	108
データ型	72
テキストエディタ	23
デフォルト動作	
イベント発生時の—	156, 166
デベロッパーツール	38
テンプレート文字列	61, 84, 114
同一オリジン	225
トランスパイル	21, 234, 239

■な行

日時	
現在の—	196
—を取得する	196
—を整形する	199

■は行

パース	13, 34
配列	75, 134, 140
パラメータ	29, 30, 120
複数の—	121
比較演算子	101

引き算	65
非同期通信 (ⓂAjax / ⓂXHR)	
	14, 19, 173, 214
ファンクション	118
ブール値 (Ⓜブーリアン値)	91
フォーム	164
ブラウザー	22
ブロックスコープ	107
プロパティ	29, 30, 80
—の書き換え	83
—の追加	83
—の読み出し	82
—名	80
べき乗	68
変数	49
ファンクションの—への代入	122
文字列中に—を埋め込む	62

■ま行

ミリ秒	207
メソッド	29
文字エンコーディング形式	28
文字列	55, 72
—連結	57

■や行

要素	
HTML—	15
—にイベントを設定する	155, 176
—のコンテンツを書き換える	70, 154, 175
—の取得	69, 174
—の属性を書き換える	154
呼び出し	
ファンクションの—	120
予約語	50

■ら行

ライブラリ	174
乱数	208
リクエスト	12, 213
利用規約	220
レスポンス	12, 213
論理演算子	102

■わ行

割り算	65
-----	----

[著者プロフィール]

狩野 祐東（かのう すけはる）

Web / アプリケーション UI デザイナー、エンジニア。執筆家。

アメリカ・サンフランシスコで UI デザイン理論を学ぶ。帰国後会社勤務を経てフリーランス。
2016 年に株式会社 Studio947 を設立。Web サイトやアプリケーションのインターフェース
デザイン、インタラクティブコンテンツの開発を数多く手がける。各種セミナーや研修講師
としても活動中。著書多数。

サンプル制作協力：狩野さやか（Studio947）

執筆アシスタント：青砥愛子（Studio947）

株式会社 Studio947 ホームページ：<https://studio947.net>

[スタッフ]

ブックデザイン：坂本真一郎（クオルデザイン）

本文イラスト：大野 文彰（大野デザイン事務所）

編集協力：片野 美都

イラスト協力：薄井久美子

DTP：本菌 直美（株式会社アクティブ）

知識ゼロからはじめる ゆっくり・ていねい JavaScript ES6 対応

2019 年 2 月 1 日 初版第 1 刷発行

著 者 狩野 祐東

発行人 片柳 秀夫

編集人 三浦 聡

発行所 ソシム株式会社

<http://www.socym.co.jp/>

〒101-0064 東京都千代田区神田猿楽町 1-5-15

猿楽町 SS ビル 3F

TEL 03-5217-2400（代表）

FAX 03-5217-2420

印刷・製本 シナノ印刷株式会社

定価はカバーに表示してあります。

落丁・乱丁は弊社販売部までお送りください。送料弊社負担にてお取り替えいたします。

ISBN978-4-8026-1124-4

Printed in Japan

©2019 Sukeharu Kanou