jQuery +jQuery UI +jQuery Mobile 逆引きハンドブック

古籏 一浩◆著

Web制作で圧倒的な人気を誇る
JavaScriptライブラリ「jQuery」「jQuery UI」
「jQuery Mobile」を網羅的に解説!

Web開発者の必携書!

jQuery 1.X/2.X 対応!

24時間無料でサンプルデータをダウンロードできます。

C&R研究所

jQuery +jQuery UI +jQuery Mobile 逆引きハンドブック

古籏 一浩◆著

C&R研究所

■権利について

- 本書に記述されている社名・製品名などは、一般に各社の商標または登録商標です。
- 本書では™、©、®は割愛しています。

■本書の内容について

- 本書は著者・編集者が実際に操作した結果を慎重に検討し、著述・編集しています。ただし、本書の記述内容に関わる運用結果にまつわるあらゆる損害・障害につきましては、責任を負いませんのであらかじめご了承ください。
- 本書で紹介しているコードの実行結果の画面などは、環境によって異なる場合がございますので、あらかじめ、ご了承ください。
- 本書の内容は、2014年9月現在の情報を基に記述しています。仕様の変更やブラウザのバージョンアップなどにより、サンプルの動作が変わったり、プログラムの書き換えが必要になったり、動作しなくなったりする場合があります。また、紹介しているURLなども変更になる場合があります。あらからじめ、ご了承ください。

■サンプルについて

- 本書で紹介しているサンプルは、C&R研究所のホームページ(http://www.c-r.com)からダウンロードすることができます。ダウンロード方法については、5ページを参照してください。
- サンプルデータの動作などについては、著者・編集者が慎重に確認しております。ただし、サンプルデータの運用結果にまつわるあらゆる損害・障害につきましては、責任を負いませんのであらかじめご了承ください。
- サンプルデータの著作権は、著者及びC&R研究所が所有します。許可なく配布・販売することは堅く禁止します。

●本書の内容についてのお問い合わせについて

この度はC&R研究所の書籍をお買いあげいただきましてありがとうございます。本書の内容に関するお問い合わせは、「書名」「該当するページ番号」「返信先」を必ず明記の上、C&R研究所のホームページ(http://www.c-r.com/)の右上の「お問い合わせ」をクリックし、専用フォームからお送りいただくか、FAXまたは郵送で次の宛先までお送りください。お電話でのお問い合わせや本書の内容とは直接的に関係のない事柄に関するご質問にはお答えできませんので、あらかじめご了承ください。

〒950-3122 新潟県新潟市北区西名目所4083-6　株式会社 C&R研究所　編集部
FAX 025-258-2801
『jQuery+jQuery UI+jQuery Mobile逆引きハンドブック』サポート係

PROLOGUE

JavaScript誕生から約10年後の2006年に産声を上げたjQueryは、瞬く間に普及していきました。今やWebサイト制作においてjQueryは欠かせないライブラリとなっています。

jQueryは手軽にページ内容の要素を操作でき、複雑な処理も短いプログラムで実現できます。このため、jQueryはプログラマ、Webデザイナーが多く利用しています。また、jQueryをベースにしたプラグインが数多くあり、自分でコードを書かなくても検索すれば必要なプラグインが出てきます。他にもUI構築に便利なjQuery UIもあります。

また、スマートフォンやタブレットに対応したjQuery Mobileも用意されています。これは、iPhone/Android/Windows Phoneなどで動作するモバイル用のUIライブラリです。モバイルと名前についていますが、スマートフォンやタブレットしか使えないわけではなく、パソコンでもプログラムを変更することなく利用できます。

jQueryは機種間の違いを吸収してくれる便利なライブラリですが、jQuery Mobileはさらにスマートフォンやタブレット間の動作の違いを吸収してくれます。jQuery/jQuery Mobileを使えば、より簡単に素早くWebサイトを制作することができます。

これだけ普及したjQueryですから、互換性を持つライブラリもいくつか存在します。最も有名なのがZepto.jsです。Zepto.jsはjQueryの機能の中で多く利用されるものだけをピックアップし、コンパクトなサイズにまとめています。このため、パソコンよりもスマートフォンで使用されることが多いライブラリとなっています。

本書ではjQueryとjQuery UI、jQuery Mobileの3つに関して、よく使われる項目を逆引きの形でまとめました。また、互換ライブラリであるZepto.jsで動作するかどうかや使用時の注意点を掲載しています。他にもApp Frameworkがありますが、このライブラリに関しても、なるべく記載するようにしました。

jQueryは1.x系から2.0系へと大きくバージョンアップされ、過去のコードが一部動かなくなる場面も出てきます。このため1.x系と2.0系との違いも記述するようにしました（多くはjQuery Migrate Pluginを使えば解決できます）。本書は執筆期間が長かったため、jQueryのバージョン1.6.4から最新の2.1.1まで複数のバージョンが混在しています。なるべく、新しい情報を入れるようにしましたが、一部古いバージョンでないと動作しないサンプルもあります（これに関しては本文中に記載しています）。

2014年に入ってからAngularJSなど大規模開発用のフレームワークも普及しつつあります。今後は大規模開発と、中小規模開発に分化してしまうのかもしれません。それにより使われるフレームワークやライブラリは変わっていくでしょう。もちろんjQueryも時代の変化に合わせていけるのであれば今後も使われ続けると思います。

本書がjQuery/jQuery Mobileを使ったサイト制作の手助けになれば幸いです。

2014年9月

古籏一浩

本書について

動作確認の環境について

本書ではWindows 7+Google Chrome、OS X（Mac OS X）+Google Chrome、およびiPhone（iOS 7）、Android（4.2以降）で動作確認をしています。動作するブラウザに関しては、jQueryおよびjQuery UI、jQuery Mobileが対応している種類とバージョンになります。なお、jQuery Mobileでは動作するブラウザをグレードで分けていますが、本書ではグレードAに対応した環境を前提にしています。グレードBおよびCに関しては本書で示している画面や動作とは異なる場合があります。

- jQuery Mobile 1.4 Supported Platforms

  URL http://jquerymobile.com/gbs/1.4/

jQueryおよびjQuery UI、jQuery Mobileには複数のバージョンがありますが、その中でバージョンアップにより機能が削除されたものもあります。本書では削除された一部の機能に関しても掲載しています。ただし、削除された機能は最新版では動作しませんので、そのような場合は本文中に記載しています。

また、本書ではサンプルコードが特定の統一されたライブラリのバージョンではなく、複数のバージョンが混在しています。jQueryの場合、1.6.4 ～ 2.1.1まで各バージョンが混在しています。なるべく、新しいバージョンに対応するようにしましたが、使用している機能の関係上、古いバージョンを指定して動作させるサンプルもいくつか含まれています。

jQuery互換ライブラリであるZepto.jsやApp Frameworkに関しては動作の違いなどを掲載していますが、バージョンアップにより動作が変わってしまう可能性があります。

CDN（Contents Delivery Network ／コンテンツデリバリネットワーク）について

本書に掲載されているサンプルはCDNを利用しています。このため、オフライン環境では正常に動作しないことがあります。ネットワークが利用できることが最低条件になります。また、非同期通信を使ったサンプルはローカル環境では動作せず、サーバーにファイルをアップロードする必要があります。

また、CDNを利用する際にプロトコルで次のようにhttpを指定しています。

```
<script src="http://ajax.googleapis.com/ajax/libs/jquery/2.1.0/jquery.min.js"></script>
```

httpだけでなくhttpsが混在するサイトの場合、次のように「http:」部分を削除して利用してください。

```
<script src="//ajax.googleapis.com/ajax/libs/jquery/2.1.0/jquery.min.js"></script>
```

本書の表記方法

本書の表記についての注意点は、次のようになります。

▶サンプルコードの中の▼について

本書に記載したサンプルコードは、誌面の都合上、1つのサンプルコードがページをまたがって記載されていることがあります。その場合は▼の記号で、1つのコードであることを表しています。

▶ブラウザとライブラリの対応バージョンについて

本書では項目のタイトル部分に、ブラウザとライブラリの対応バージョンの表を記載しています。表の右列にブラウザや各ライブラリが対応しているバージョンが記載されています。

サンプルファイルのダウンロードについて

本書のサンプルデータは、C&R研究所のホームページからダウンロードすることができます。本書のサンプルを入手するには、次のように操作します。

❶「http://www.c-r.com/」にアクセスします。
❷トップページ左上の「商品検索」欄に「149-8」と入力し、[検索]ボタンをクリックします。
❸検索結果が表示されるので、本書の書名のリンクをクリックします。
❹書籍詳細ページが表示されるので、[サンプルデータダウンロード]ボタンをクリックします。
❺下記の「ユーザー名」と「パスワード」を入力し、ダウンロードページにアクセスします。
❻「サンプルデータ」のリンク先のファイルをダウンロードし、保存します。

サンプルのダウンロードに必要な
ユーザー名とパスワード

ユーザー名	jQuim
パスワード	2c8wz

※ユーザー名・パスワードは、半角英数字で入力してください。また、「J」と「j」や「K」と「k」などの大文字と小文字の違いもありますので、よく確認して入力してください。

サンプルコードの利用方法

サンプルファイルは、CHAPTERごとのフォルダの中に、項目番号のフォルダに分かれています。サンプルはZIP形式で圧縮してありますので、解凍してお使いください。

それぞれのフォルダ内には、HTMLファイルやJavaScriptファイル、CSSファイル、画像ファイルなどが保存されています。

なお、サンプルはネットワーク接続されていることが大前提です。特にローカルディスク上（HTMLファイルを直接、ブラウザで開く方法）では動作しない場合があります。動作の確認を行うにはネットワーク環境（Webサーバー上）にサンプルファイルをアップロードしてください。Webサーバーについては、ローカルサーバーでも構いませんが、CDNを利用してjQueryライブラリなどを読み込んでいるためインターネットに接続されている必要があります。

CONTENTS

PART-1 jQuery

CHAPTER 01 jQueryの基礎知識

001 jQueryとは? …… 38
COLUMN ■jQueryのライセンスについて

002 jQueryのバージョンについて …… 45
COLUMN ■jQuery/jQuery UI/jQuery Mobileの最新バージョンを確認する

003 jQuery互換ライブラリについて …… 48

004 jQueryライブラリの読み込みについて …… 52
COLUMN ■CDNで使われているサーバーの応答速度を調べるには

005 jQueryとDOM(ドキュメントオブジェクトモデル)について …… 55

CHAPTER 02 jQuery Core(コア/基本)

006 jQueryオブジェクトを指定する …… 60
ONEPOINT ■jQueryオブジェクトへのアクセス方法について

007 「$」が他のライブラリと干渉しないようにする …… 62
ONEPOINT ■「$」の干渉を回避するには「noConflict()」メソッドを使う
COLUMN ■無名関数を利用して「$」を渡す

CHAPTER 03 セレクタ関係

008 すべての要素を指定する …… 66
ONEPOINT ■すべての要素を指定するには「$()」のパラメーターに「"*"」を指定する
COLUMN ■「document.getElementsByTagName()」を使って
サンプルと同じ処理を行う

009 ID名で指定する …… 69
ONEPOINT ■ID名で指定するには「$()」に「#ID名」を指定する
COLUMN ■より高速に処理するには

010 クラス名で指定する …… 71
ONEPOINT ■クラス名で指定するには「$()」に「.クラス名」を指定する
COLUMN ■jQueryを使わない場合のコード

011 属性名で指定する …… 73
ONEPOINT ■属性名で指定するには「$()」に「[属性名]」を指定する
COLUMN ■jQueryを使わない場合のコード

012 複数の要素を指定する …… 76
ONEPOINT■複数の要素を指定するには「,」(カンマ)で区切って列記する
COLUMN■jQueryを使わない場合のコード

013 複数の属性セレクタを指定する …… 78
ONEPOINT■複数の属性セレクタを指定するには
「[属性名1][属性名2]」のように列記する
COLUMN■jQueryを使わない場合のコード

014 特定の要素を指定する …… 80
ONEPOINT■特定の要素を指定するには
「$()」のパラメーターに要素名を文字列で指定する
COLUMN■jQueryを使わない場合のコード

015 空の要素を指定する …… 83
ONEPOINT■空の要素を指定するには「:empty」を指定する
COLUMN■jQueryを使わない場合のコード

016 見出し要素を指定する …… 85
ONEPOINT■見出し要素を指定するには「:header」を指定する
COLUMN■jQueryを使わない場合のコード

017 選択された要素を指定する …… 87
ONEPOINT■選択された要素を指定するには「:selected」を指定する
COLUMN■jQueryを使わない場合のコード

018 最初の要素を指定する …… 89
ONEPOINT■最初の要素を指定するには「:first」と指定する
COLUMN■jQueryを使わない場合のコード

019 最後の要素を指定する …… 91
ONEPOINT■最後の要素を指定するには「:last」と指定する
COLUMN■jQueryを使わない場合のコード

020 要素の前後関係を指定する …… 93
ONEPOINT■要素の前後関係を指定するには「+」を使う
COLUMN■jQueryを使わない場合のコード

021 指定した出現順番より小さい要素を指定する …… 95
ONEPOINT■指定した出現順番より小さい要素を指定するには
「:lt(出現位置)」を指定する
COLUMN■jQueryを使わない場合のコード

022 指定した出現順番より大きい要素を指定する …… 98
ONEPOINT■指定した出現順番より大きい要素を指定するには
「:gt(出現位置)」を指定する
COLUMN■jQueryを使わない場合のコード

023 指定した出現位置の要素を指定する …… 101
ONEPOINT■指定した出現位置の要素を指定するには「:eq(出現位置)」を指定する
COLUMN■jQueryを使わない場合のコード

026 偶数番目／奇数番目の要素を指定する……103
ONEPOINT ■偶数番目の要素を指定するには「:even」、奇数番目の要素を指定するには「:odd」を指定する
COLUMN ■jQueryを使わない場合のコード

025 親要素を指定する……106
ONEPOINT ■親要素を指定するには「:parent」を指定する
COLUMN ■jQueryを使わない場合のコード

026 子要素だけを指定する……109
ONEPOINT ■子要素だけを指定するには「:only-child」を指定する
COLUMN ■jQueryを使わない場合のコード

027 親子関係にある子要素を指定する……111
ONEPOINT ■親子関係にある子要素を指定するには「親要素名>子要素名」を指定する
COLUMN ■jQueryを使わない場合のコード

028 特定の要素を祖先に持つ要素を指定する……113
ONEPOINT ■特定の要素を祖先に持つ要素を指定するには「祖先要素名 子要素名」を指定する
COLUMN ■jQueryを使わない場合のコード

029 特定の子孫要素を持つ要素を指定する……116
ONEPOINT ■特定の子孫要素を持つ要素を指定するには「:has()」を使う

030 指定した順番の子要素を指定する……119
ONEPOINT ■指定した順番の子要素を指定するには「:nth-child(出現位置)」を指定する
COLUMN ■jQueryを使わない場合のコード

031 最初の子要素を指定する……122
ONEPOINT ■最初の子要素を指定するには「:first-child」を指定する
COLUMN ■jQueryを使わない場合のコード

032 最後の子要素を指定する……124
ONEPOINT ■最後の子要素を指定するには「:last-child」を指定する
COLUMN ■jQueryを使わない場合のコード

033 兄弟要素を指定する……126
ONEPOINT ■兄弟要素を指定するには「~」を使う
COLUMN ■jQueryを使わない場合のコード

034 表示されている要素を指定する……129
ONEPOINT ■表示されている要素を指定するには「:visible」を指定する
COLUMN ■jQueryを使わない場合のコード

035 表示されていない要素を指定する……132
ONEPOINT ■表示されていない要素を指定するには「:hidden」を指定する
COLUMN ■jQueryを使わない場合のコード

036 入力が有効になっている要素を指定する……135
ONEPOINT ■入力が有効になっている要素を指定するには「:enabled」を指定する
COLUMN ■jQueryを使わない場合のコード

042 入力が無効になっている要素を指定する ……138
ONEPOINT■入力が無効になっている要素を指定するには「:disabled」を指定する
COLUMN■jQueryを使わない場合のコード

043 内容が特定の単語で始まるか「-」で区切られている要素を指定する …141
ONEPOINT■特定の単語で始まるか「-」で区切られている要素を指定するには「[属性名|=単語]」とする
COLUMN■jQueryを使わない場合のコード

044 属性が指定した文字列から始まる要素を指定する ……143
ONEPOINT■属性が指定した文字列から始まる要素を指定するには「[属性名^=内容]」とする
COLUMN■「[属性^=値]」の再実行時の動作について
COLUMN■jQueryを使わない場合のコード

045 属性が指定した文字列で終わる要素を指定する ……147
ONEPOINT■属性が指定した文字列で終わる要素を指定するには「[属性名$=内容]」とする
COLUMN■jQueryを使わない場合のコード
COLUMN■HTML5で指定できる「type」属性の形式

046 属性が指定した内容になっている要素を指定する ……150
ONEPOINT■属性が指定した値になっている要素を指定するには「[属性名=内容]」とする
COLUMN■「[属性=値]」の再実行時の動作について
COLUMN■jQueryを使わない場合のコード

047 属性が指定した内容ではない要素を指定する ……154
ONEPOINT■属性が指定した値になっていない要素を指定するには「[属性名!=内容]」とする
COLUMN■jQueryを使わない場合のコード

048 属性が指定した単語を含む要素を指定する ……157
ONEPOINT■属性が指定した単語を含む要素を指定するには「[属性名~=内容]」とする
COLUMN■jQueryを使わない場合のコード

049 属性が指定した文字列を含む要素を指定する ……160
ONEPOINT■属性が指定した文字列を含む要素を指定するには「[属性名*=内容]」とする
COLUMN■jQueryを使わない場合のコード
COLUMN■テーブルで3の倍数の行に色を付ける

050 指定した内容が含まれる要素を指定する ……163
ONEPOINT■指定した内容が含まれる要素を指定するには「:contains()」と使う
COLUMN■jQueryを使わない場合のコード

051 一致しない要素を指定する ……165
ONEPOINT■一致しない要素を指定するには「:not(セレクタ)」を指定する
COLUMN■jQueryを使わない場合のコード

052 チェックボックスを指定する……168
ONEPOINT■チェックボックスを指定するには「:checkbox」とする
COLUMN■jQueryを使わない場合のコード

048 チェック済みのチェックボックスを指定する ……170
ONEPOINT ■チェック済みのチェックボックスを指定するには「":checkbox:checked"」を指定する
COLUMN ■jQueryを使わない場合のコード

049 ボタンを指定する ……173
ONEPOINT ■ボタンを指定するには「:button」とする
COLUMN ■jQueryを使わない場合のコード

050 ラジオボタンを指定する ……176
ONEPOINT ■ラジオボタンを指定するには「:radio」とする
COLUMN ■jQueryを使わない場合のコード

051 リセットボタンを指定する……178
ONEPOINT ■リセットボタンを指定するには「:reset」と指定する
COLUMN ■jQueryを使わない場合のコード

052 送信ボタンの要素を指定する ……180
ONEPOINT ■送信ボタンを指定するには「:submit」と指定する
COLUMN ■jQueryを使わない場合のコード

053 テキストフィールドを指定する……182
ONEPOINT ■テキストフィールドを指定するには「:text」と指定する
COLUMN ■jQueryを使わない場合のコード

054 入力フォームのimageを指定する ……184
ONEPOINT ■入力フォームのimageを指定するには「$()」に「:image」を指定する

055 入力フォームの「input」要素を指定する ……186
ONEPOINT ■入力フォームの「input」要素を指定するには「:input」と指定する
COLUMN ■jQueryを使わない場合のコード

056 ファイルアップロード要素を指定する……188
ONEPOINT ■ファイルアップロード要素を指定するには「:file」を指定する
COLUMN ■jQueryを使わない場合のコード

057 フォーカスしている要素を指定する ……190
ONEPOINT ■フォーカスしている要素を指定するには「:focus」を指定する
COLUMN ■jQueryを使わない場合のコード

058 アニメーションしている要素を指定する……192
ONEPOINT ■アニメーションしている要素を指定するには「:animated」を指定する
COLUMN ■アニメーションしている要素を停止させるには

CHAPTER 04 属性関係

059 要素の属性値を取得／設定する……196
ONEPOINT ■属性値を取得／設定するには「attr()」メソッドを使う
COLUMN ■「attr()」メソッドと「prop()」メソッドの結果の違い
COLUMN ■jQueryを使わない場合のコード

060 プロパティ値を取得／設定する ……199
ONEPOINT■プロパティ値を取得／設定するには「prop()」メソッドを使う
COLUMN■「attr()」メソッドと「prop()」メソッドの結果の違い
COLUMN■jQueryを使わない場合のコード

061 属性を削除する ……202
ONEPOINT■属性を削除するには「removeAttr()」メソッドを使う
COLUMN■「removeAttr()」メソッドと「removeProp()」メソッドについて

062 プロパティを削除する ……205
ONEPOINT■プロパティを削除するには「removeProp()」メソッドを使う
COLUMN■「removeProp()」メソッドの注意点
COLUMN■「removeAttr()」メソッドと「removeProp()」メソッドについて

063 入力フォームの値を取得／設定する ……208
ONEPOINT■入力フォームの値を取得／設定するには「val()」メソッドを使う
COLUMN■カスタムデータ属性へのアクセス

CHAPTER 05 Traversing(トラバース)

064 繰り返し処理を行う ……212
ONEPOINT■繰り返し処理を行うには「each()」メソッドを使う

065 要素を検索する ……214
ONEPOINT■要素を検索するには「find()」メソッドを使う

066 要素をフィルタリングする ……216
ONEPOINT■要素をフィルタリングするには「filter()」メソッドを使う

067 条件にあった場合に処理する ……218
ONEPOINT■条件にあった場合に処理するには「is()」メソッドを使う

068 要素を配列に変換する ……220
ONEPOINT■要素を配列に変換するには「map()」メソッドを使う

069 処理対象を追加する ……223
ONEPOINT■処理対象を途中から追加するには「add()」メソッドを使う

070 自身の要素も対象にする ……225
ONEPOINT■自身の要素も対象にするには「andSelf()」メソッドを使う

071 最初の要素／最後の要素を取得する ……227
ONEPOINT■最初の要素を指定するには「first()」メソッド、
最後の要素を指定するには「last()」メソッドを使う

072 指定した要素を持つ場合に処理する ……229
ONEPOINT■指定した要素を持つ場合に処理するには「has()」メソッドを使う

073 指定した要素以外を対象にする ……231
ONEPOINT■指定した要素以外を対象にする「not()」メソッドを使う

074 指定した位置の要素を指定する ……233
ONEPOINT ■指定位置の要素を指定するには「eq()」メソッドを使う

075 親要素を取得する ……235
ONEPOINT ■親要素を取得するには「parent()」メソッドを使う

076 親要素や祖先要素をすべて取得する ……237
ONEPOINT ■親要素や祖先要素をすべて取得するには「parents()」メソッドを使う

077 指定した親要素直前までの親要素を取得する ……239
ONEPOINT ■指定した親要素直前までの親要素や祖先要素をすべて取得するには「parentsUntil()」メソッドを使う

078 要素から最も近い親要素を取得する ……241
ONEPOINT ■要素から最も近い親要素を取得するには「closest()」メソッドを使う
COLUMN ■「closest()」メソッドのその他の書式について

079 特定の「position」属性の親要素を取得する ……244
ONEPOINT ■特定の「position」属性の親要素を取得するには「offsetParent()」メソッドを使う

080 子要素をすべて取得する ……246
ONEPOINT ■子要素をすべて取得するには「children()」メソッドを使う

081 すべての子要素／テキスト要素を取得する ……248
ONEPOINT ■すべての子要素／テキスト要素を取得するには「contents()」メソッドを使う

082 兄弟要素をすべて取得する ……250
ONEPOINT ■兄弟要素をすべて取得するには「siblings()」メソッドを使う

083 前の兄弟要素を対象にする ……252
ONEPOINT ■前の兄弟要素を対象にするには「prev()」メソッドを使う

084 前の兄弟要素をすべて対象にする ……254
ONEPOINT ■前の兄弟要素をすべて対象にするには「prevAll()」メソッドを使う

085 次の兄弟要素を対象にする ……256
ONEPOINT ■次の兄弟要素を対象にするには「next()」メソッドを使う

086 後の兄弟要素をすべて対象にする ……258
ONEPOINT ■後の兄弟要素をすべて対象にするには「nextAll()」メソッドを使う

087 指定範囲の兄弟要素を取得する ……260
ONEPOINT ■指定範囲の兄弟要素を取得するには「slice()」メソッドを使う

088 指定した要素より前で特定の条件以外の兄弟要素を対象にする …262
ONEPOINT ■指定した要素より前で特定の条件以外の兄弟要素を対象にするには「prevUntil()」メソッドを使う

089 指定した要素より後で特定の条件以外の兄弟要素を対象にする …264
ONEPOINT ■指定した要素より後で特定の条件以外の兄弟要素を対象にするには「nextUntil()」メソッドを使う

090 1つ前の要素集合に戻す ……266
ONEPOINT ■1つ前の要素セットを指定するには「end()」メソッドを使う
COLUMN ■jQuery Deferredを使ったアニメーション

CHAPTER 06 操作

091 要素を複製する……270
ONEPOINT■要素を複製するには「clone()」メソッドを使う
COLUMN■「clone()」メソッドについての注意点

092 要素を切り離す……273
ONEPOINT■要素を切り離すには「detach()」メソッドを使う

093 要素を削除する……276
ONEPOINT■要素を削除するには「remove()」メソッドを使う

094 要素の内容を空にする……278
ONEPOINT■要素の内容を空にするには「empty()」メソッドを使う

095 要素を置き換える……280
ONEPOINT■要素を置き換えるには「replaceAll()」メソッドを使う

096 指定した要素で置き換える……282
ONEPOINT■要素を置き換えるには「replaceWith()」メソッドを使う

097 要素内に追加する……284
ONEPOINT■要素内に追加するには「appendTo()」メソッドを使う

098 要素の前に挿入する……286
ONEPOINT■要素の前に挿入するには「insertBefore()」メソッドを使う

099 要素の後に挿入する……288
ONEPOINT■要素の後に挿入するには「insertAfter()」メソッドを使う

100 要素の後に追加する……290
ONEPOINT■要素の後に追加するには「append()」メソッドを使う

101 要素の前に文字を挿入する……292
ONEPOINT■要素の前に文字を挿入するには「before()」メソッドを使う

102 要素の後に文字を挿入する……294
ONEPOINT■要素の後に文字を挿入するには「after()」メソッドを使う

103 各要素の先頭に挿入する……296
ONEPOINT■各要素の先頭に新たに要素を挿入するには「prepend()」メソッドを使う

104 セレクタを使って各要素の先頭に挿入する……298
ONEPOINT■セレクタを使って各要素の先頭に挿入するには
「prependTo()」メソッドを使う

105 要素の内側に入れ込む……300
ONEPOINT■要素の内側に入れ込むには「wrapInner()」メソッドを使う

106 要素内の文字列を取得／設定する……302
ONEPOINT■要素内の文字列を取得／設定するには「text()」メソッドを使う
COLUMN■jQueryを使わない場合のコード

107 HTML要素を取得／設定する……305
ONEPOINT■HTML要素を取得／設定するには「html()」メソッドを使う
COLUMN■jQueryを使わない場合のコード

108 フォーム関連要素の値を取得／設定する ……308
ONEPOINT ■フォーム関連要素の値を取得／設定するには「val()」メソッドを使う

109 要素の属性値を取得／設定する ……310
ONEPOINT ■要素の属性値を取得／設定するには「attr()」メソッドを使う

110 指定した要素で囲む ……312
ONEPOINT ■指定した要素で囲むには「wrap()」メソッドを使う

111 指定した要素をまとめて囲む ……314
ONEPOINT ■要素をまとめて囲むには「wrapAll()」メソッドを使う

112 囲んでいる要素を消去する ……316
ONEPOINT ■囲んでいる要素を消去するには「unwrap()」メソッドを使う
COLUMN ■古いイベントメソッド

CHAPTER 07 CSS（スタイルシート）

113 スタイルシートのクラスを追加する ……320
ONEPOINT ■スタイルシートのクラスを追加するには「addClass()」メソッドを使う

114 スタイルシートのクラスを削除する ……322
ONEPOINT ■スタイルシートのクラスを削除するには「removeClass()」メソッドを使う

115 クラスがあるかどうか調べる ……324
ONEPOINT ■クラスがあるかどうか調べるには「hasClass()」メソッドを使う

116 スタイルシートのクラスのON/OFFを切り替える ……326
ONEPOINT ■スタイルシートのクラスのON/OFFを切り替えるには
「toggleClass()」メソッドを使う

117 指定したCSSプロパティ値を取得／設定する ……328
ONEPOINT ■指定したCSSプロパティ値を取得／設定するには
「css()」メソッドを使う
COLUMN ■jQueryを使わずに要素のスタイルプロパティの値を取得する

118 要素の幅を取得／設定する ……332
ONEPOINT ■要素の幅を取得／設定するには「width()」メソッドを使う

119 要素の内側／外側の幅を取得する ……334
ONEPOINT ■要素の内側の幅を取得するには「innerWidth()」メソッド、
外側を幅を取得するには「outerWidth()」メソッド使う

120 要素のオフセット位置を取得／設定する ……337
ONEPOINT ■要素のオフセット位置を取得／設定するには「offset()」メソッドを使う

121 親要素からの相対位置を取得する ……339
ONEPOINT ■親要素からの相対位置を取得するには「position()」メソッドを使う

122 スクロール量を取得／設定する ……341
ONEPOINT ■スクロール量を取得／設定するには「scrollTop()」メソッドを使う

123 スタイルシートプロパティをフックする
（スタイルシートプロパティに処理を追加する）……344
ONEPOINT■スタイルシートプロパティをフックするには「$.cssHooks」を使う

CHAPTER 08 イベント関係

124 イベントを設定する ……348
ONEPOINT■イベントを設定するには「on()」メソッドを使う
COLUMN■jQueryのイベント設定について

125 1回だけ実行するイベントを設定する ……351
ONEPOINT■1回だけイベント処理を行うには「one()」メソッドを使う

126 要素にイベントを紐付ける（バインドする）……353
ONEPOINT■要素にイベントを紐付ける（バインドする）には「bind()」メソッドを使う

127 要素にイベントを送信する ……355
ONEPOINT■イベントを送信するには
「trigger()」メソッドまたは「triggerHandler()」メソッドを使う

128 イベントを解除する ……357
ONEPOINT■イベントを解除するには「off()」メソッドを使う
COLUMN■プラグインなどでイベントが重複しないようにする

129 設定したイベントを解除する ……360
ONEPOINT■イベントを解除するには「unbind()」メソッドを使う

130 イベントハンドラにデータを渡す……364
ONEPOINT■イベントハンドラにデータを渡すには
最初のパラメーターにオブジェクトリテラル形式を指定する

131 イベント間でデータを受け渡す ……366
ONEPOINT■イベント間でデータを受け渡すには
イベントハンドラ内で「return」を使う

132 オリジナルのイベントオブジェクトを参照する ……368
ONEPOINT■オリジナルのイベントオブジェクトを参照するには
「originalEvent」オブジェクトを使う

133 イベントの種類を取得する ……370
ONEPOINT■イベントの種類を取得するには「type」プロパティを読み出す

134 イベント発生時のキーやボタン番号を取得する ……372
ONEPOINT■押されたキーコードやボタン番号の取得するには
「which」プロパティを参照する

135 イベント発生時の要素を取得する ……374
ONEPOINT■イベント発生時の要素を取得するには
「currentTarget」プロパティを参照する

１３６ イベント発生時の要素だけを取得する ……376
ONEPOINT ■イベント発生時の要素だけを取得するには
「target」プロパティを参照する

１３７ イベント発生前の要素を取得する ……378
ONEPOINT ■イベント発生前の要素を取得するには
「relatedTarget」プロパティを参照する

１３８ イベント発生時の時間を取得する ……380
ONEPOINT ■イベント発生時の時間を取得するには
「timeStamp」プロパティを読み出す

１３９ イベント発生時の名前空間を取得する ……382
ONEPOINT ■イベント発生時の名前空間を取得するには
「namespace」プロパティを参照する

１４０ イベント発生時のデリゲート要素を取得する……385
ONEPOINT ■イベント発生時のデリゲート要素を取得するには
「delegateTarget」プロパティを使う

１４１ ブラウザのデフォルトの動作を禁止する ……388
ONEPOINT ■ブラウザのデフォルトの動作を禁止するには
「preventDefault()」メソッドを使う

１４２ ブラウザのデフォルトの動作を禁止したかどうか調べる ……391
ONEPOINT ■ブラウザのデフォルトの動作を禁止したかどうか調べるには
「isDefaultPrevented()」メソッドを使う

１４３ イベントの伝達を禁止する ……393
ONEPOINT ■イベントの伝達を禁止するには「stopPropagation()」メソッド
または「stopImmediatePropagation()」メソッドを使う

１４４ イベントの伝達禁止状態を調べる ……396
ONEPOINT ■イベントの伝達禁止状態を調べるには
「isPropagationStopped()」メソッドを使う

１４５ イベントの伝達を完全に禁止したかどうか調べる……399
ONEPOINT ■イベントの伝達を完全に禁止したかどうか調べるには
「isImmediatePropagationStopped()」メソッドを使う

１４６ エラー発生時の処理を行う ……402
ONEPOINT ■エラー発生時の処理を行うには「error()」メソッドを使う

１４７ クリックされた場合に処理する ……404
ONEPOINT ■クリック時に処理を行うには「click()」メソッドを使う
COLUMN ■jQueryを使わない場合のコード

１４８ クリックするたびに交互に処理を行う……407
ONEPOINT ■交互に処理を行うには「toggle()」メソッドを使う

１４９ ダブルクリックされた場合に処理する……409
ONEPOINT ■ダブルクリック時に処理を行うには「dblclick()」メソッドを使う
COLUMN ■jQueryを使わない場合のコード

１５０ マウスダウン／マウスアップされた場合に処理する……412
ONEPOINT■マウスダウン時に処理を行うには「mousedown()」メソッド、
マウスアップ時に処理を行うには「mouseup()」メソッドを使う
COLUMN■App Frameworkでの対応
COLUMN■jQueryを使わない場合のコード

１５１ マウスオーバー／マウスアウトの場合に処理する ……415
ONEPOINT■マウスオーバー時に処理を行うには「mouseover()」メソッド、
マウスアウト時に処理を行うには「mouseout()」メソッド使う
COLUMN■App Frameworkでの対応

１５２ マウスが要素の領域内に入った／出た場合に処理する ……420
ONEPOINT■マウスが要素の領域内に入った場合に処理を行うには「mouseenter()」
メソッド、出た場合に処理を行うには「mouseleave()」メソッドを使う
COLUMN■App Frameworkの対応

１５３ マウス移動時に処理する ……425
ONEPOINT■マウス移動時に処理を行うには「mousemove()」メソッドを使う
COLUMN■App Frameworkでの対応

１５４ ホバー（マウスオーバー／アウト）時の処理を行う ……427
ONEPOINT■ホバー（マウスオーバー／アウト）時の処理を行うには
「hover()」メソッドを使う

１５５ テキストフィールドの文字が選択された場合に処理する ……429
ONEPOINT■テキスト選択時に処理を行うには「select()」メソッドを使う

１５６ キーが押された／離された場合に処理する ……431
ONEPOINT■キー入力イベントの処理を行うには
「keydown()」「keypress()」「keyup()」メソッドを使う

１５７ フォーカスされた場合に処理する ……433
ONEPOINT■フォーカスされた場合に処理するには
「focus()」メソッドまたは「focusin()」メソッドを使う

１５８ フォーカスを外した場合に処理する ……435
ONEPOINT■フォーカスを外した場合に処理するには
「blur()」メソッドまたは「focusout()」メソッドを使う

１５９ 内容が変更された場合に処理する ……437
ONEPOINT■内容が変更された場合に処理するには「change()」メソッドを使う

１６０ 送信する前に処理する ……439
ONEPOINT■送信時に処理を行うには「submit()」メソッドを使う

１６１ ページ上の座標を取得する ……441
ONEPOINT■ページ上の座標を取得するには
「pageX」プロパティと「pageY」プロパティを参照する

１６２ ページが読み込まれた場合に処理する ……443
ONEPOINT■ページの読み込み完了時に処理を行うには「ready()」メソッドを使う

１６３ ページや画像の読み込みが完了した場合に処理する ……445
ONEPOINT■読み込み完了時に処理を行うには「load」イベントを捕捉する
COLUMN■jQueryを使わない場合のコード

164 ページから離脱する場合に処理する ……447
ONEPOINT ■ページから離脱時に処理を行うには「unload()」メソッドを使う
165 ウィンドウがリサイズされた場合に処理する ……449
ONEPOINT ■ウィンドウリサイズ時に処理を行うには「resize()」メソッドを使う
166 スクロールした場合に処理する ……451
ONEPOINT ■スクロールした場合に処理を行うには「scroll()」メソッドを使う

CHAPTER 09 エフェクト

167 アニメーションを行う ……454
ONEPOINT ■アニメーションを指定するには「animate()」メソッドを使う
168 アニメーションを停止する ……456
ONEPOINT ■アニメーションを停止させるには「stop()」メソッドを使う
169 アニメーション効果の処理間隔を指定する ……459
ONEPOINT ■アニメーション効果の間隔は「$.fx.interval」プロパティで指定する
170 アニメーション効果を有効にするか無効にするかを指定する ……461
ONEPOINT ■アニメーション効果のON/OFFを設定するには
「$.fx.off」プロパティを使う
171 待ち時間を指定する ……463
ONEPOINT ■待ち時間(ディレイ)を指定するには「delay()」メソッドを使う
172 要素の表示／非表示を切り替える ……465
ONEPOINT ■要素の表示と非表示を交互に切り替えるには「toggle()」メソッドを使う
173 要素をアニメーションしながら表示／非表示にする ……468
ONEPOINT ■要素の表示するには「show()」メソッド、
非表示にするには「hide()」メソッドを使う
174 要素をフェードイン／フェードアウトさせる ……471
ONEPOINT ■要素をフェードインするには「fadeIn()」メソッド、
フェードアウトするには「fadeOut()」メソッドを使う
COLUMN ■バージョンによるパラメータの違い
COLUMN ■jQueryを使わずにフェードアウトさせる
175 要素を交互にフェードさせながら表示／非表示を切り替える ……476
ONEPOINT ■要素を交互にフェードイン／フェードアウトさせるには
「toggleFade()」メソッドを使う
176 要素をスライドアップ／スライドダウンさせる ……479
ONEPOINT ■要素をスライドアップさせるには「slideUp()」メソッド、
スライドダウンさせるには「slideDown()」メソッドを使う

177 要素を交互にスライドアップ／スライドダウンさせる ……482
ONEPOINT■要素を交互にスライドアップ／スライドダウンさせるには「toggleSlide()」メソッドを使う
COLUMN■「toggleSlide()」メソッド使用時に要素が表示されているか非表示かを調べるには

178 キューの取得／設定を行う ……485
ONEPOINT■キューの取得／設定を行うには「queue()」メソッドを使う

179 キューから処理を取り出し実行する ……487
ONEPOINT■キューから処理を取り出し実行するには「dequeue()」メソッドを使う

180 キューを消去する ……489
ONEPOINT■キューを消去するには「clearQueue()」メソッドを使う

CHAPTER 10 非同期通信

181 非同期通信を使う ……492
ONEPOINT■非同期通信を使うには「$.ajax()」メソッドを使う

182 非同期通信時のオプションを設定する ……495
ONEPOINT■非同期通信時のオプションを設定するには「$.ajaxSetup()」メソッドを使う

183 非同期オプションをフィルタリングする ……497
ONEPOINT■非同期オプションをフィルタリングするには「$.ajaxPrefilter()」メソッドを使う

184 GET/POSTメソッドで通信する ……499
ONEPOINT■GETメソッドで通信するには「$.get()」メソッド、POSTメソッドで通信するには「$.post()」メソッドを使う

185 JSONファイルを読み込む ……501
ONEPOINT■JSONファイルを読み込むには「getJSON()」メソッドを使う

186 JavaScriptファイルを読み込む ……503
ONEPOINT■JavaScriptファイルを読み込むには「getScript()」メソッドを使う

187 サーバー上のファイルを読み込んで表示する ……505
ONEPOINT■サーバー上のファイルを読み込んで表示するには「load()」メソッドを使う
COLUMN■jQueryを使わない場合のコード

188 リクエスト送信時に処理する ……507
ONEPOINT■リクエスト送信時に処理するには「ajaxSend()」メソッドを使う

189 読み込み開始時／終了時に処理する ……509
ONEPOINT■読み込み開始時に処理するには「ajaxStart()」メソッド、終了時に処理するには「ajaxStop()」メソッドを使う

190 読み込み成功時に処理する……511
ONEPOINT ■読み込み成功時に処理するには「ajaxSuccess()」メソッドを使う
COLUMN ■Zepto.js 1.0の場合

191 読み込み完了時に処理する……514
ONEPOINT ■読み込み完了時に処理するには「ajaxComplete()」メソッドを使う

192 読み込みエラー時に処理する……516
ONEPOINT ■読み込みエラー時に処理するには「ajaxError()」メソッドを使う

193 オブジェクトをシリアライズする……518
ONEPOINT ■オブジェクトをシリアライズするには「param()」メソッドを使う

194 フォームデータをシリアライズする……520
ONEPOINT ■フォームデータをシリアライズするには「serialize()」メソッドを使う

195 jQuery Deferredで非同期処理が完了したときに処理する……523
ONEPOINT ■jQuery Deferredで非同期処理が完了したときに処理するには「always()」メソッドを使う

196 jQuery Deferredで非同期処理が成功したときに処理する……525
ONEPOINT ■jQuery Deferredで非同期処理が成功したときに処理するには「done()」メソッドを使う

197 jQuery Deferredで非同期処理が失敗したときに処理する……527
ONEPOINT ■jQuery Deferredで非同期処理が失敗したときに処理するには「fail()」メソッドを使う

CHAPTER 11 非同期処理

198 jQuery Deferredで非同期処理を行う……530
ONEPOINT ■jQuery Deferredを使って非同期処理を行うには「promise()」メソッドを使う
COLUMN ■jQuery Deferredの解説ページ

199 「Deferred」オブジェクトの状態を調べる……533
ONEPOINT ■「Deferred」オブジェクトの状態を取得するには「state()」メソッドを使う

200 すべての非同期処理が完了してから処理する……537
ONEPOINT ■すべての非同期処理の完了を待つには「$.when()」メソッドを使う

201 成功／失敗に関係なく終了時の処理を行う……539
ONEPOINT ■成功／失敗に関係なく終了時の処理を行うには「always()」メソッドを使う

202 成功時に処理を行う……542
ONEPOINT ■成功時に処理を行うには「done()」メソッドを使う

203 失敗時に処理を行う……545
ONEPOINT ■失敗時に処理を行うには「fail()」メソッドを使う

２０４ 成功時と失敗時で処理を分岐させる……548
ONEPOINT■成功時と失敗時の処理を分岐させるには「then()」メソッドを使う

２０５ notifyで通知する／ progressで処理する……551
ONEPOINT■通知を行うには「notify()」メソッドと「progress()」メソッドを組み合わせる

２０６ notifyでコンテキストを渡す……554
ONEPOINT■コンテキストを渡すには「notifyWith()」メソッドを使う

CHAPTER 12 ユーティリティ

２０７ ブラウザの種類を調べる……558
ONEPOINT■ブラウザの種類を調べるには「$.browser」を使う

２０８ サポート状態を調べる……561
ONEPOINT■サポート状態を調べるには「$.support」を使う

２０９ オブジェクトの種類を調べる……564
ONEPOINT■オブジェクトの種類を調べるには「type()」メソッドを使う

２１０ プレーンオブジェクトかどうか調べる……566
ONEPOINT■プレーンオブジェクトかどうか調べるには「$.isPlainObject()」を使う

２１１ オブジェクトが列挙型／空かどうか調べる……568
ONEPOINT■内容が列挙型／空かどうか調べるには「$.isEmptyObject()」メソッドを使う

２１２ 「Window」オブジェクトかどうか調べる……570
ONEPOINT■「Window」オブジェクトかどうか調べるには「$.isWindow()」を使う

２１３ 「this」が指し示すオブジェクトを指定する……572
ONEPOINT■「this」が指し示すオブジェクトを指定するには「$.proxy()」メソッドを使う
COLUMN■「this」を固定する

２１４ 数値かどうか調べる……576
ONEPOINT■数値かどうか調べるには「$.isNumeric()」を使う

２１５ 関数かどうか調べる……578
ONEPOINT■関数かどうか調べるには「$.isFunction()」を使う

２１６ 配列かどうか調べる……580
ONEPOINT■配列かどうか調べるには「$.isArray()」を使う

２１７ 配列に該当要素があるかどうか調べる……582
ONEPOINT■配列に該当要素があるかどうか調べるには「$.inArray()」を使う

２１８ 要素がある要素のノードに含まれているかどうか調べる……584
ONEPOINT■要素がある要素のノードに含まれているかどうか調べるには「contains()」メソッドを使う

２１９ 指定されたノードがXMLノード内にあるか調べる ……586
ONEPOINT ■指定されたノードがXMLノード内にあるか調べるには「$.isXMLDoc()」を使う

２２０ 配列に変換する ……588
ONEPOINT ■配列に変換するには「makeArray()」メソッドを使う

２２１ 配列要素を処理して新たな配列を生成する ……590
ONEPOINT ■配列要素を処理して新たな配列を生成するには「map()」メソッドを使う

２２２ 2つの配列を連結する ……592
ONEPOINT ■2つの配列を連結するには「merge()」メソッドを使う

２２３ 配列要素のフィルタ処理を行う ……594
ONEPOINT ■配列要素のフィルタ処理を行うには「grep()」メソッドを使う

２２４ 要素にデータを保存する ……596
ONEPOINT ■要素にデータを保存するには「data()」メソッドを使う

２２５ 要素のデータを削除する ……598
ONEPOINT ■要素のデータを削除するには「removeData()」メソッドを使う

２２６ 文字列の左右の空白や改行を削除する ……602
ONEPOINT ■空白を削除するには「trim()」メソッドを使う

２２７ 重複していないものを取り出す ……604
ONEPOINT ■重複していないものを取り出すには「unique()」メソッドを使う

２２８ JSON形式のデータをJavaScriptオブジェクトに変換する ……606
ONEPOINT ■JSON形式のデータをJavaScriptオブジェクトに変換するには「$.parseJSON()」メソッドを使う

２２９ XML文字列をXMLオブジェクトに変換する ……608
ONEPOINT ■XML文字列をXMLオブジェクトに変換するには「$.parseXML()」メソッドを使う

２３０ jQueryオブジェクトだけに機能を追加する ……610
ONEPOINT ■jQueryオブジェクトだけに機能を追加するには「extend()」メソッドを使う

２３１ グローバルコンテキストで式を評価する ……612
ONEPOINT ■グローバルコンテキストで評価するには「$.globalEval()」メソッドを使う

２３２ 現在のミリ秒を取得する ……614
ONEPOINT ■現在のミリ秒を取得するには「$.now()」を使う

２３３ 「ready」イベントを保留する ……616
ONEPOINT ■「ready」イベントを保留するには「$.holdReady()」メソッドを使う
COLUMN ■アイコンや画像にSVGを使う

PART-2 jQuery UI

CHAPTER 01 インタラクション

234 要素をドラッグ可能にする ……622
ONEPOINT■要素をドラッグ可能にするには「draggable()」メソッドを使う
COLUMN■jQueryを使わずにドラッグ可能／ドラッグ禁止にする

235 要素をドロップできるようにする ……625
ONEPOINT■要素をドロップできるようにするには「droppable()」メソッドを使う

236 要素をリサイズ可能にする ……627
ONEPOINT■要素をリサイズ可能にするには「resizable()」メソッドを使う

237 リストを選択できるようにする ……629
ONEPOINT■リストを選択できるようにするには「selectable()」メソッドを使う

238 リストを並び替え(ソート)可能にする ……631
ONEPOINT■リストを並び替え(ソート)可能にするには「sortable()」メソッドを使う

CHAPTER 02 ウィジェット

239 アコーディオン機能を設定する ……634
ONEPOINT■アコーディオン機能を設定するには「accordion()」メソッドを使う
COLUMN■アコーディオンパネルをキーボードで操作する

240 アコーディオン展開時のアニメーションを禁止する ……637
ONEPOINT■アニメーション処理を禁止するには
パラメーターに「animate: false」を指定する

241 アコーディオンパネルを展開できないようにする ……640
ONEPOINT■アコーディオンを展開できないようにするには
パラメーターに「disabled: true」を指定する

242 アクティブになったパネルのヘッダー情報を取得する ……642
ONEPOINT■ヘッダー情報を取得するには
「ui」オブジェクト内にjQueryオブジェクトを参照する

243 パネルヘッダーのアイコンを指定する ……645
ONEPOINT■パネルヘッダーのアイコンを指定するには
パラメーターに「{ icon : { 種類 : アイコン名 }}」を指定する

244 オートコンプリート機能を設定する ……647
ONEPOINT■オートコンプリート機能を設定するには
「autocomplete()」メソッドを使う

245 入力候補に画像を表示する ……649
ONEPOINT■入力候補に画像を表示するにはjQueryプラグインを使う

246 入力候補の表示位置を指定する……651
ONEPOINT ■入力候補が表示されるポップアップの位置を設定するには「position」オプションを使う

247 入力候補が表示されるまでの時間を設定する……653
ONEPOINT ■入力候補が表示されるまでの時間を設定するには「delay」オプションを使う

248 入力候補が表示されるポップアップにフォーカスする……655
ONEPOINT ■ポップアップにフォーカスするには「autoFocus」オプションを使う

249 一定の文字数を入力しないと入力候補が表示されないようにする 657
ONEPOINT ■一定の文字数を入力したら候補が表示されるようにするには「minLength」オプションを使う

250 オートコンプリート機能を削除する……660
ONEPOINT ■オートコンプリート機能を削除するには「autocomplete("destroy")」とする

251 グラフィカルなボタンを設定する……663
ONEPOINT ■グラフィカルなボタンにするには「button()」メソッドを使う

252 ボタンを無効にする／有効にする……665
ONEPOINT ■ボタンの無効化と有効化を設定するには「disable」オプションを使う

253 ボタンに表示するアイコンを設定／変更する……667
ONEPOINT ■アイコンを設定／変更するには「icons」オプションを使う

254 ボタンに表示する文字を設定／変更する……670
ONEPOINT ■文字を設定／変更するには「label」オプションを使う

255 ボタン内の文字を非表示にする……672
ONEPOINT ■ボタン内の文字の表示／非表示を設定するには「text」オプションを使う

256 カレンダーで選択できるようにする……674
ONEPOINT ■テキストフィールドでカレンダーを表示するには「datepicker()」メソッドを使う

257 カレンダーをダイアログモードで表示する……676
ONEPOINT ■カレンダーをダイアログとして表示するには最初のパラメーターに「"dialog"」を指定する

258 カレンダーで選択可能な日付範囲を指定する……679
ONEPOINT ■選択可能な範囲を指定するには「minDate」オプションと「maxDate」オプションを指定する

259 ダイアログを表示する……682
ONEPOINT ■ダイアログを表示するには「dialog()」メソッドを使う

260 ダイアログの表示位置を指定する……685
ONEPOINT ■ダイアログの表示位置を指定するには「position」オプションを使う

261 ダイアログの幅を設定する……687
ONEPOINT ■横幅を指定するには「width」オプション、縦幅を指定する場合は「height」オプションを使う

262 ダイアログをドラッグ／リサイズできないようにする ……690
ONEPOINT■ダイアログのドラッグを禁止するには「draggable」オプションに「false」を、
リサイズを禁止するには「resizable」オプションに「false」を指定する

263 ダイアログ生成時に自動的に表示しないようにする……692
ONEPOINT■ダイアログの自動表示を禁止するには
「autoOpen」オプションに「false」を指定する

264 ダイアログを閉じる／開かれているか調べる ……694
ONEPOINT■ダイアログを閉じるには「dialog("close")」とする

265 ダイアログが表示された場合に処理する ……697
ONEPOINT■ダイアログが表示された場合に処理を行うには
「open」イベントを捕捉する

266 ダイアログがドラッグされた場合に処理する……699
ONEPOINT■ダイアログがドラッグされた場合に処理するには
「dragStart」「drag」「dragStop」イベントを補足する

267 ダイアログがリサイズされた場合に処理する ……702
ONEPOINT■ダイアログがリサイズされた場合に処理するには
「resizeStart」「resize」「resizeStop」イベントを補足する

268 プログレスバーを表示する ……705
ONEPOINT■プログレスバーを表示するには「progressbar()」メソッドを使う

269 プログレスバーの最大値を設定する ……707
ONEPOINT■最大値の設定するには「max」オプションを使う

270 プログレスバーの値を読み出す／設定する ……709
ONEPOINT■プログレスバーの値を読み出すには「progressbar()」メソッドを使う

271 プログレスバーの値が更新された／完了した場合に処理する ……712
ONEPOINT■プログレスバーの値が更新された場合に処理するには「change」イベント、
完了した場合に処理するには「complete」イベントを補足する

272 プログレスバーを無効にする／有効にする ……715
ONEPOINT■プログレスバーの無効にするには「progressbar("disable")」、
有効にするには「progressbar("enable")」とする

273 プログレスバーを破棄する ……718
ONEPOINT■プログレスバーを破棄するには「progressbar("destroy")」とする

274 スライダーを表示する ……721
ONEPOINT■スライダーを表示するには「slider()」メソッドを使う

275 範囲スライダーを表示する ……724
ONEPOINT■スライダーを範囲内で指定できるようにするには
「range」オプションに「true」を設定する

276 縦型のスライダーを表示する ……727
ONEPOINT■スライダーの表示方向を設定するには「orientation」オプションを使う

277 スライダーの1回の移動量（ステップ）を設定する……729
ONEPOINT■スライダーの移動量（ステップ）を設定するには「step」オプションを使う

278 スライダーを無効にする／有効にする ……731
ONEPOINT ■スライダーを無効にするには「slider("disable")」、
有効にするには「slider("enable")」とする

279 スピナーを設定する ……733
ONEPOINT ■スピナーを表示するには「spinner()」メソッドを使う

280 スピナーの最小値と最大値を設定する ……735
ONEPOINT ■スピナーの最小値は「min」オプション、
最大値は「max」オプションで設定する

281 スピナーの増減値を設定する ……737
ONEPOINT ■スピナーの増減値を設定するには「step」オプションを使う

282 スピナーの生成時に処理する ……739
ONEPOINT ■スピナーの生成時に処理するには「create」オプションを使う

283 スピナーのイベントを処理する ……741
ONEPOINT ■スピナーのイベントを処理するには
「start」「spin」「stop」「change」イベントを補足する

284 スピナーを破棄する ……744
ONEPOINT ■スピナーを破棄するには「spinner("destroy")」とする

285 タブ機能を設定する ……747
ONEPOINT ■タブ機能を設定するには「tabs()」メソッドを使う

286 表示したいタブを指定する ……750
ONEPOINT ■表示したいタブを指定するには「active」オプションを使う

287 タブを無効にする／有効にする ……753
ONEPOINT ■タブを無効にするには「disabled」オプションを使う

288 タブを削除する ……757
ONEPOINT ■タブを削除するには「tabs("destroy")」とする

289 ツールチップ機能を設定する ……760
ONEPOINT ■ツールチップ機能を設定するには「tooltip()」メソッドを使う

290 ツールチップの表示位置を指定する ……762
ONEPOINT ■ツールチップで位置を設定するには「position」オプションを使う

291 ツールチップの内容を変更する ……764
ONEPOINT ■ツールチップの内容を変更するには
「tooltip("option", "content", 表示する内容)」のように指定する

292 プログラムからツールチップの表示制御を行う……766
ONEPOINT ■ツールチップを表示するには「tooltip("open")」、
非表示にするには「tooltip("close")」とする

293 セレクトメニューを表示する ……769
ONEPOINT ■セレクトメニューを表示するには要素に対して「selectmenu()」メソッド
を呼び出す

294 メニュー項目にカスタムアイコンを表示する……771
ONEPOINT ■メニュー項目にカスタムアイコンを表示するには
カスタムメソッドを作成する

295 セレクトメニューを有効にする／無効にする……774
ONEPOINT■セレクトメニューの無効／有効を設定するには「disabled」オプションを使う

296 選択項目が変化した場合に処理する……777
ONEPOINT■選択項目が変化した場合に処理するには「change」イベントを捕捉する

297 セレクトメニューのUIを破棄する……780
ONEPOINT■セレクトメニューのUIを破棄するには「selectmenu("destroy")」とする
COLUMN■QUnitによるテスト

CHAPTER 03 エフェクト

298 ブラインド(blind)エフェクトを行う……784
ONEPOINT■blindエフェクトを行うには「toggle()」メソッドや「effect()」メソッドで最初のパラメーターに「blind」の文字を指定する

299 バウンド(bounce)エフェクトを行う……787
ONEPOINT■bounceエフェクトを行うには「toggle()」メソッドや「effect()」メソッドで最初のパラメーターに「bounce」の文字を指定する

300 クリップ(clip)エフェクトを行う……790
ONEPOINT■clipエフェクトを行うには「toggle()」メソッドや「effect()」メソッドで最初のパラメーターに「clip」の文字を指定する

301 ドロップ(drop)エフェクトを行う……793
ONEPOINT■dropエフェクトを行うには「toggle()」メソッドや「effect()」メソッドで最初のパラメーターに「drop」の文字を指定する

302 分割(explode)エフェクトを行う……796
ONEPOINT■explodeエフェクトを行うには「toggle()」メソッドや「effect()」メソッドで最初のパラメーターに「explode」の文字を指定する

303 フェードイン／フェードアウト(fade)を行う……799
ONEPOINT■fadeエフェクトを行うには「toggle()」メソッドや「effect()」メソッドで最初のパラメーターに「fade」の文字を指定する

304 折りたたみ(fold)エフェクトを行う……802
ONEPOINT■foldエフェクトを行うには「toggle()」メソッドや「effect()」メソッドで最初のパラメーターに「fold」の文字を指定する

305 ハイライト(highlight)エフェクトを行う……805
ONEPOINT■highlightエフェクトを行うには「toggle()」メソッドや「effect()」メソッドで最初のパラメーターに「highlight」の文字を指定する

306 吹きかけ(puff)エフェクトを行う……808
ONEPOINT■puffエフェクトを行うには「toggle()」メソッドや「effect()」メソッドで最初のパラメーターに「puff」の文字を指定する

307 パルス／点滅(pulsate)エフェクトを行う……811
ONEPOINT ■pulsateエフェクトを行うには、「toggle()」メソッドや「effect()」メソッドで最初のパラメーターに「pulsate」の文字を指定する

308 スケール(scale)エフェクトを行う……814
ONEPOINT ■scaleエフェクトを行うには「toggle()」メソッドや「effect()」メソッドで最初のパラメーターに「scale」の文字を指定する

309 シェイク(shake)エフェクトを行う……817
ONEPOINT ■shakeエフェクトを行うには「toggle()」メソッドや「effect()」メソッドで最初のパラメーターに「shake」の文字を指定する

310 サイズ変更(size)エフェクトを行う……820
ONEPOINT ■sizeエフェクトを行うには「toggle()」メソッドや「effect()」メソッドで最初のパラメーターに「size」の文字を指定する

311 スライド(slide)エフェクトを行う……823
ONEPOINT ■slideエフェクトを行うには「toggle()」メソッドや「effect()」メソッドで最初のパラメーターに「slide」の文字を指定する

312 移動(transfer)エフェクトを行う……826
ONEPOINT ■transferエフェクトを行うには「toggle()」メソッドや「effect()」メソッドで最初のパラメーターに「transfer」の文字を指定する

313 エフェクトを使う……829
ONEPOINT ■エフェクトを行うには「effect()」メソッドを使う

314 エフェクトを使って表示する……832
ONEPOINT ■エフェクトを使って要素を表示するには「show()」メソッドを使う

315 エフェクトを使って非表示にする……835
ONEPOINT ■エフェクトを使って非表示にするには「hide()」メソッドを使う

316 エフェクト付きでCSSクラスを追加する……838
ONEPOINT ■エフェクト付きでCSSクラスを追加するには「addClass()」メソッドを使う
COLUMN ■「easing」を指定したときの動き

317 エフェクト付きでCSSクラスを削除する……842
ONEPOINT ■CSSクラスの削除時にエフェクトを行うには「removeClass()」メソッドを使う

318 エフェクト付きでCSSクラスを切り替える……846
ONEPOINT ■CSSクラスの切り替え時にエフェクトを行うには「switchClass()」メソッドを使う

319 CSSクラスをエフェクト付きで交互に切り替える……850
ONEPOINT ■CSSクラスの追加と削除を交互に行うには「toggleClass()」メソッドを使う

320 エフェクト処理を交互に切り替える……854
ONEPOINT ■エフェクト処理の方向などを交互に切り替えるには「toggle()」メソッドを使う

PART-3 jQuery Mobile

CHAPTER 01 ページ構成

321 単ページを構成する……860
ONEPOINT■jQuery Mobileでのページ構成
COLUMN■1ページしかない場合

322 複数ページを構成する……863
ONEPOINT■複数ページの扱い

323 ヘッダー／フッターを固定する……865
ONEPOINT■ヘッダー／フッターを固定するには「data-position="fixed"」を指定する

324 共通のフッターとして使用する……868
ONEPOINT■共通のフッターを利用するには
「data-position="fixed"」と「data-id="共通のID"」を設定する

325 全画面（フルスクリーン）モードにする……871
ONEPOINT■全画面モードにするには「data-fullscreen="true"」を指定する

326 特定のテーマで表示する……874
ONEPOINT■テーマを指定するには「data-theme」属性を使う

327 ヘッダーに戻るボタンを付ける／戻るボタンのテキストを変更する 876
ONEPOINT■戻るボタンを表示するには「data-add-back-btn="true"」を指定する

328 ヘッダーにボタンを追加する……879
ONEPOINT■ヘッダーにボタンを表示するには「a」要素を記述する

329 フッターにボタンを追加する……883
ONEPOINT■フッターにボタンを表示するには「class="ui-bar"」を指定する

330 ナビゲーションバーを表示する……885
ONEPOINT■ナビゲーションバーを表示するには
「div」要素や「nav」要素にナビゲーション項目となるリストを記述する

CHAPTER 02 ダイアログ

331 ダイアログとして表示する……888
ONEPOINT■ダイアログを表示するには「a」要素に「data-rel="dialog"」を指定するか、
ダイアログとして表示する要素に「data-role="dialog"」を指定する

332 ダイアログの表示時のアニメーション方法を指定する……891
ONEPOINT■ダイアログ表示のアニメーションは
「a」要素の「data-transition」で指定する

333 ダイアログの閉じるボタンの文字を設定する……894
ONEPOINT■クローズボタンの文字を変更するには
「data-close-btn-text」属性を使う

334 ダイアログをスクリプトで閉じる……896
ONEPOINT ■ダイアログをスクリプトで閉じるには
「$(".ui-dialog").dialog("close")」とする

CHAPTER 03 ページリンク

335 他のページにリンクする……900
ONEPOINT ■他のページに移動するには

336 ページ移動時のアニメーション方法や移動方向を指定する……904
ONEPOINT ■移動時のエフェクトを指定するには
「a」要素の「data-transition」を使う

337 スクリプトでページを移動させる……907
ONEPOINT ■スクリプトでページを移動するには
「$.mobile.changePage()」メソッドを使う

338 現在のページを参照する……911
ONEPOINT ■現在のページ情報を得るには
「$.mobile.activePage」プロパティを使う

339 ページ内容をあらかじめ読み込む／キャッシュする……914
ONEPOINT ■ページデータをキャッシュするには同一ドメインに「a」要素で
「data-prefetch」を指定するか、キャッシュしたいページに
「data-dom-cache="true"」を指定する

340 スクリプトでページデータをキャッシュする……918
ONEPOINT ■ページデータをスクリプトでキャッシュするには
「$.mobile.loadPage()」メソッドを使う

341 スクリプトでページをスクロールさせる……920
ONEPOINT ■スクリプトでページをスクロールさせるには
「$.mobile.silentScroll()」メソッドを使う

CHAPTER 04 リスト／段組／パネル

342 リストを表示する……924
ONEPOINT ■iPhone/Androidライクなリスト表示を行うには
「data-role="listview"」を指定する

343 階層化リストにする……926
ONEPOINT ■リストを階層化して表示するにはリスト要素内にリスト要素を記述する

344 リストを角丸にする……928
ONEPOINT ■リストを角丸にするには「ul」または「ol」要素に「data-role="listview"」
を指定し、さらに「data-inset="true"」を指定する

345 リストを分割して表示する ……930
ONEPOINT■リストを分割して表示するには「li」要素内に「a」要素を2つ入れる

346 分割リストのアイコンを指定する ……933
ONEPOINT■分割リストで表示されるアイコンを設定するには「data-icon」を使う

347 分割リストのテーマを設定する ……935
ONEPOINT■分割リストで表示されるアイコンのテーマを設定するには
「data-split-theme」を使う

348 説明付きのリストにする ……937
ONEPOINT■見出しと説明文を同時に表示するには
「li」要素内で「h1」要素と「p」要素を記述する

349 説明と補足説明付きのリストにする ……939
ONEPOINT■説明と補足説明付きのリストにするには
「li」要素内で「h1」要素と「p」要素を記述する

350 リストに項目の区切り(デバイダ)を設定する……941
ONEPOINT■項目の区切りを指定するには
「li」要素に「data-role="list-divider"」を指定する

351 リスト項目にカウンタを表示する……943
ONEPOINT■数値(カウンタ)を表示するにはリスト項目内で
「値」のように指定する

352 画像付きのリストにする ……945
ONEPOINT■リストに画像を表示するには「li」項目内に「img」要素を記述する

353 リスト項目を検索できるようにする……948
ONEPOINT■リスト項目を検索するには、「ul」要素に「data-filter="true"」を指定する

354 段組にする……951
ONEPOINT■段組にするには「class="ui-grid-a"」か「class="ui-grid-b"」を指定する

355 開閉パネルを表示する……953
ONEPOINT■開閉パネルを表示するには
「div」要素に「data-role="collapsible"」を指定する

356 アコーディオンパネルを表示する ……956
ONEPOINT■アコーディオンパネルを表示するには
「div」要素に「data-role="collapsible-set"」を指定する

CHAPTER 05 フォーム

357 テキストフィールドを表示する……960
ONEPOINT■デフォルトでテキストフィールドの外観が変更される

358 検索ボックスを表示する ……962
ONEPOINT■検索ボックスを表示するには
「input」要素の「type」属性に「search」の文字を指定する

359 電子メール専用のフィールドを表示する ……964
ONEPOINT ■電子メール専用の入力フィールドを表示するには
「input」要素の「type」属性に「email」の文字を指定する

360 電話番号入力用のフィールドを表示する ……966
ONEPOINT ■電話番号入力用のフィールドを表示するには
「input」要素の「type」属性に「tel」の文字を指定する

361 URL入力用のフィールドを表示する ……968
ONEPOINT ■URL入力用のフィールドを表示するには
「input」要素の「type」属性に「url」の文字を指定する

362 スライダーを表示する ……970
ONEPOINT ■スライダーを表示するには
「input」要素の「type」属性に「range」の文字を指定する

363 範囲スライダーを表示する ……972
ONEPOINT ■範囲スライダーを表示するには
「 」とする

364 範囲スライダーのハイライトやラベルを消す ……974
ONEPOINT ■ハイライトやラベルを消すには「 」に
「data-highlight="false"」を指定する

365 トグルスイッチを表示する ……976
ONEPOINT ■トグルスイッチ(フリップスイッチ)を表示するには
「select」要素と「option」要素を使う

366 ラジオボタンを表示する ……978
ONEPOINT ■ラジオボタンを表示するにはラジオボタン全体を「fieldset」要素で囲む

367 チェックボックスを表示する ……980
ONEPOINT ■チェックボックスを表示するには
チェックボックス全体を「fieldset」要素で囲む

368 セレクトメニューを表示する ……983
ONEPOINT ■セレクトメニューを表示するには
「select」要素と「option」要素を記述する

369 送信／リセットボタンを文字サイズに合わせて表示する ……987
ONEPOINT ■文字数に合わせて送信／リセットボタンを表示するには「input」要素に
「data-inline="true"」を指定する

370 アイコンとアイコンの表示位置を指定する ……989
ONEPOINT ■アイコンを表示するには「data-icon」属性を使う
COLUMN ■アイコンだけのボタンにする
COLUMN ■jQueryからSVGを操作する

CHAPTER 06 設定／イベント／テーブル／地図

371 デフォルトの設定を変更する ……994
ONEPOINT■デフォルトの設定を変更するには「mobileinit」イベントを補足する
COLUMN■jQuery Mobile 1.4でのボタン設定の変更点

372 jQuery Mobile専用イベントを処理する ……999
ONEPOINT■jQuery Mobile専用のイベント

373 レスポンシブ対応のテーブル（表組み）を表示する …… 1002
ONEPOINT■テーブルをレスポンシブ対応にするには
「data-role="table"」を指定する

374 ユーザーがテーブルで表示する列を選択できるようにする …… 1005
ONEPOINT■ユーザーにとって必要な列だけを表示できるようにするには
「table」要素に「data-mode="columntoggle"」を指定する

375 ページ全体に地図を表示する …… 1009
ONEPOINT■地図をページ全体に表示するにはスタイルシートを設定する
COLUMN■jQuery Mobileのテーマをカスタマイズする

APPENDIX

376 jQueryプラグインの作成…… 1014
377 jQuery 2.0.3のコードマップ …… 1038
378 jQueryメソッド・プロパティ対応バージョン一覧 …… 1044

●索引…… 1054

PART-1
jQuery

CHAPTER 01

jQueryの基礎知識

jQueryとは?

jQueryの概要

jQueryはJavaScriptライブラリの1つです。JavaScriptライブラリは数多くありますが、その中で最も多く使用されている標準的なライブラリです。2006年に登場し、瞬く間にスタンダードなライブラリとしての地位を確立しました。

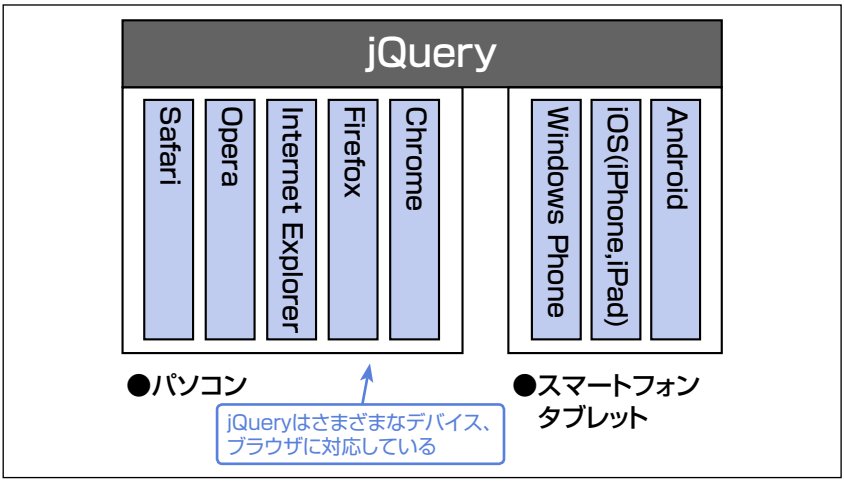
jQueryは主にブラウザ上のDOM操作(要素の操作)を行います。jQueryはJavaScript言語自体を拡張するためのライブラリではありません。また、大規模な開発ではロジックとモデル、ビューを分離して作成するため、jQueryはDOMの処理(MVCのV(View))用に使われることがあります。

なお、jQuery以前に登場したPrototype.jsライブラリはJavaScript言語自体を拡張します。jQueryではECMAScript ver 3、ver 5の違いは吸収しないので、JavaScriptネイティブで作成する場合は実行するブラウザの実行環境に注意する必要があります。

それではjQueryの特長を挙げてみます。

1. さまざまなブラウザやスマートフォン/タブレットなど多くの環境に対応
2. 処理を短いプログラムで記述できる
3. 豊富なjQueryプラグイン(ライブラリ)
4. スマートフォン/タブレットのUIに対応(統一感のあるUI)
5. 軽量でコンパクト
6. 非同期通信が簡単に処理できる(Ajax)

1ですが、jQueryは数多くの環境で動作するように作成されており、制作者がブラウザの違いやバージョンごとの違いをあまり気にせずに利用することができます。



もしjQueryがなかったら制作者はブラウザごとに異なる動作に対応し、さらにブラウザのバージョンアップによる変更、増え続けるデバイスへの対応に労力を注がなければなりません。jQueryが存在しなかった時代、ブラウザごとに異なる動作に対応するために多くの労力を費やす必要がありました。jQueryを使えば、このような労力を注がなくてもよくなります。これによりWeb制作者が本来注力すべき部分に労力を費やすことができるようになります。これは、制作物の品質向上にもなるはずです。

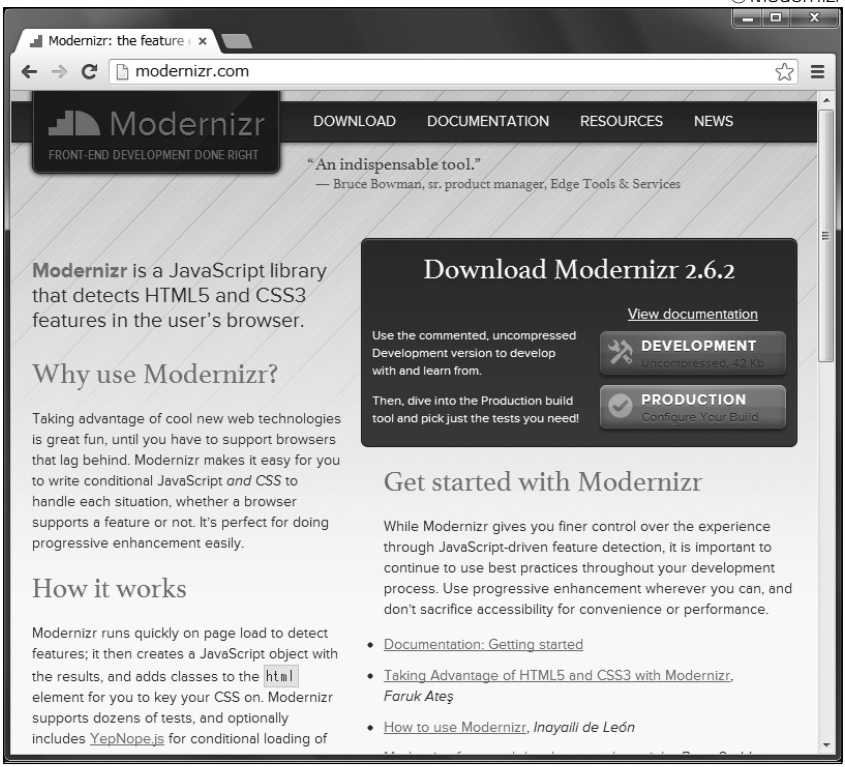
また、スマートフォン／タブレットは年々高精細な画面になっていますが、jQuery Mobile 1.4以降ではアイコンはPNG画像からSVG形式に変更されています。これにより、高精細な画面でも綺麗に表示されるように改良されています。このような変更が行われても既存のコードは変更することなく動くのもメリットの1つです。

ただし、jQueryを使ってもブラウザやスマートフォンの違いを完全に吸収できないこともあるので、その場合はModernizrなどを使って動作環境のチェックを行い、個別に対処することになります。

- Modernizr

  URL http://modernizr.com/

◉Modernizr



次に2です。jQueryは一般的なJavaScriptプログラムで同じ処理を作成した場合、よりシンプルで短いコードで記述できるようになっています。実際のプログラムで比較してみましょう。次のコードはテーブルのセルの背景色を奇数番目の場合はオレンジ色、偶数番目の場合は黄色に設定するプログラムです。

◉一般的なJavaScriptの場合

```
var tdList = document.querySelectorAll("#sale tbody tr");
for(var i=0; i<tdList.length; i++){
  if (i & 1){
    // 奇数番目の背景色をオレンジ色にする
    tdList[i].style.backgroundColor = "orange";
  }else{
    // 偶数番目の背景色を黄色にする
    tdList[i].style.backgroundColor = "yellow";
  }
}
```

同じ処理を行うプログラムをjQueryで作成すると、次のようになります。かなり短くなっているのがわかるでしょう。

◉jQueryの場合

```
// 偶数番目の背景色をオレンジ色にする
$("#sale tbody tr:even").css("background-color", "orange");
// 奇数番目の背景色を黄色にする
$("#sale tbody tr:odd").css("background-color", "yellow");
```

jQueryはCSS(スタイルシート)と同じように豊富なセレクタを持っているため、条件にあった要素だけを処理対象にすることができます。また、複数の要素に対して処理を行う場合、自動的に繰り返し処理を行います。このため、制作者側では繰り返しのための処理(ループ)を書かずに済みます。

また、jQueryではJavaScriptのメソッドチェーンという機能を利用して要素に対して連続して指示を与えることができます。たとえば、「h1」要素の内容を変更した後、文字の色を赤くした後、フェードアウトさせるには次のようになります。

```
$("h1").text("Android");
$("h1").css("color", "red");
$("h1").fadeOut(2000);
```

これだと冗長であり、毎回、「h1」要素を検索するため、速度も低下してしまいます。メソッドチェーンを使えば、次のように1行で簡潔に記述できます。

```
$("h1").text("Android").css("color", "red").fadeOut(2000);
```

行いたい処理を順番に記述すればよいのでわかりやすくなります。一部例外はありますが、jQueryはメソッドの戻り値をjQueryオブジェクトに統一しているため、メソッドチェーンを活用し

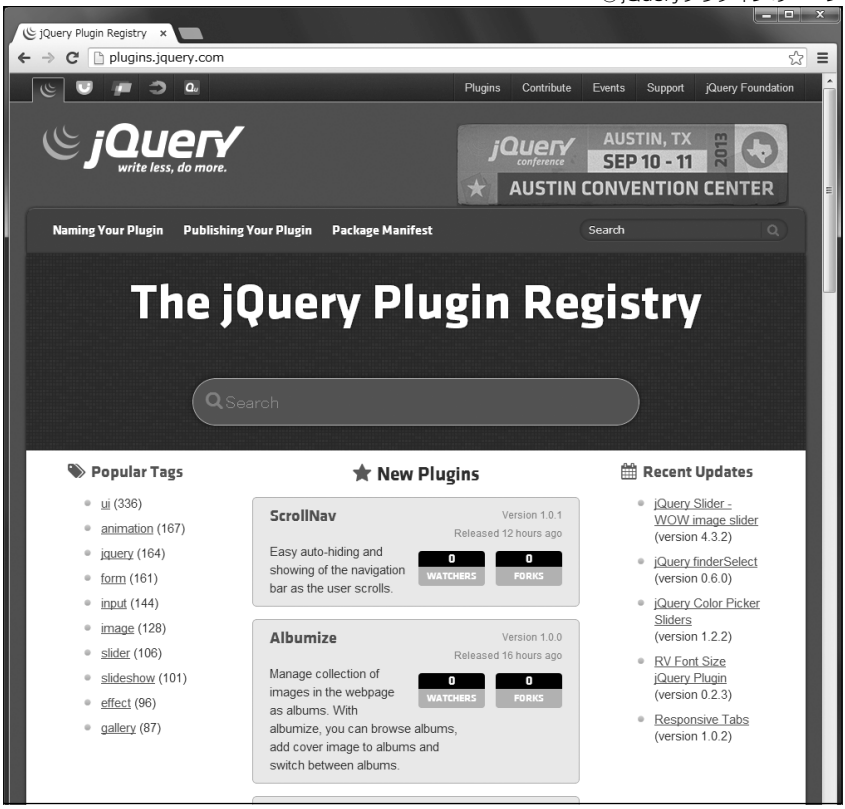
やすくなっているのです。

セレクタで複数要素を指定でき、ループの記述がなくメソッドチェーンで短く記述できるため、jQueryを使えば見通しの良いプログラムが出来上がります。プログラムが短く見通しがよければバグ(不具合)が入り込む余地も減り、制作期間の短縮(=コスト削減)にもなります。

3ですが、jQueryには非常に多くのプラグインがあります。自分で制作するより検索した方が早い場合も多々あります。jQueryプラグインは公式サイトからたどれるものと、そうでないものがあります。公式サイトにないものはGoogleの検索で「jQuery 求める機能」のようなキーワードで検索すれば該当するプラグインが表示されます。

公式サイトではプラグインはカテゴリごとに分けられています。また、プラグイン名などで検索することもできます。jQueryでは複数のプラグインを同時に使用しても、あまり不具合が発生しないようになっています。

◉jQueryプラグインのページ

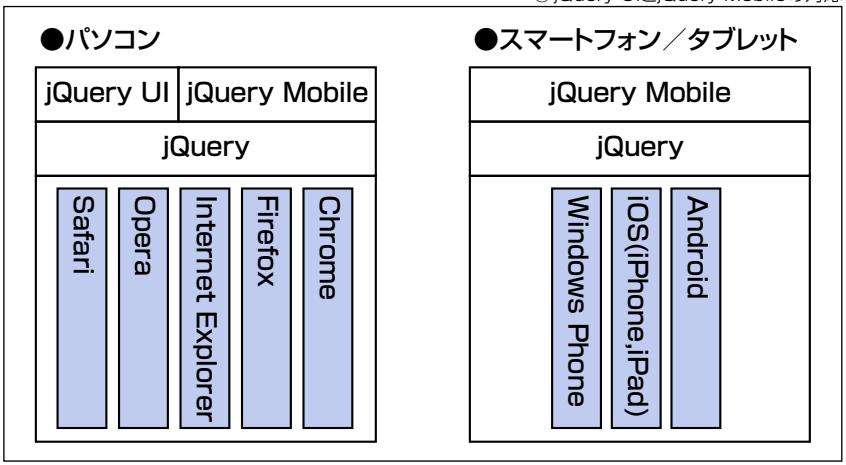


また、jQueryでは、統一感を持つUI(ユーザーインターフェース)が用意されています。これはjQuery UIとして用意されており、パソコン上で操作するインターフェースになっています。スマートフォンの場合はjQuery UIでなくjQuery Mobileを使用します。

◉jQuery UIのページ

◉jQuery Mobileのページ

◉jQuery UIとjQuery Mobileの対応



4のスマートフォン／タブレットへの対応です。jQueryはスマートフォン／タブレットに関してはjQuery Mobileが用意されています。スマートフォンの場合はjQuery Mobileを使うことで簡単にスマートフォン用のUIを持つページを作成することができます。なお、jQuery Mobileはパソコンでも使用することができます。jQuery MobileはjQueryライブラリをベースにしており、使用する場合はjQueryライブラリを読み込ませた後でjQuery Mobileのライブラリを読み込ませることになります。

jQuery Mobileを使えば、タッチ操作によるインターフェースをプログラムを作成することなく利用できます。jQuery、jQuery UIはJavaScriptプログラムを使ってインターフェースを構築する必要がありました。jQuery MobileではHTML5のカスタムデータ属性を利用することでプログラムを作成しなくてもモバイル用のUIを生成・使用することができるようになっています。もちろん、プログラムによって処理を書くこともできます。

5の軽量でコンパクトというのはファイルサイズのことです。一般的なJavaScriptライブラリのファイルは数百KB～数MBにまでなることがあります。jQueryでは最新の2.1.1でも82KB程度という非常に少ないファイルサイズになっています(ZIP圧縮した場合、さらに小さくなり33KBほどになる)。

jQueryは非常にファイルサイズが少ないのですが、スマートフォンの普及により、さらに少ないファイルサイズが求められることがあります。jQueryをZIP圧縮しても限界があります。そこで、登場したのがjQuery互換ライブラリです。jQueryとほぼ同じ機能を持ち、より少ないファイルサイズになっています。代表的なのがZepto.jsです。ファイルサイズは27KBほどでZIP圧縮すると10KB程度になります。jQueryの約1/3です。ファイルサイズが減れば、読み込み速度も速くなります。

Zepto.js以外にもApp Framework(旧名jqmobi)もあります。こちらは、スマートフォン専用でより限定された環境下で動作するライブラリです。なお、互換ライブラリに関しては別途

後述します。

6の非同期通信処理はJavaScriptがブレークする(広く使われる)きっかけになったものです。非同期通信処理はAjax(Asynchronous JavaScript + XML)と呼ばれページを切り替えることなく必要なデータを読み込み画面を表示することができます。この機能が多く使われ始めたことでjQueryなどのJavaScriptライブラリが登場したともいえます。

この非同期通信処理もjQueryでは簡単にできます。たとえば、サーバー上にあるファイルを読み込ませ表示するだけならば、「$("#msg").load("data.txt")」とするだけです。これだけで自動的にdata.txtファイルの内容が読み込まれ表示されます。また、後からJavaScriptプログラムを動的に追加することもできます。

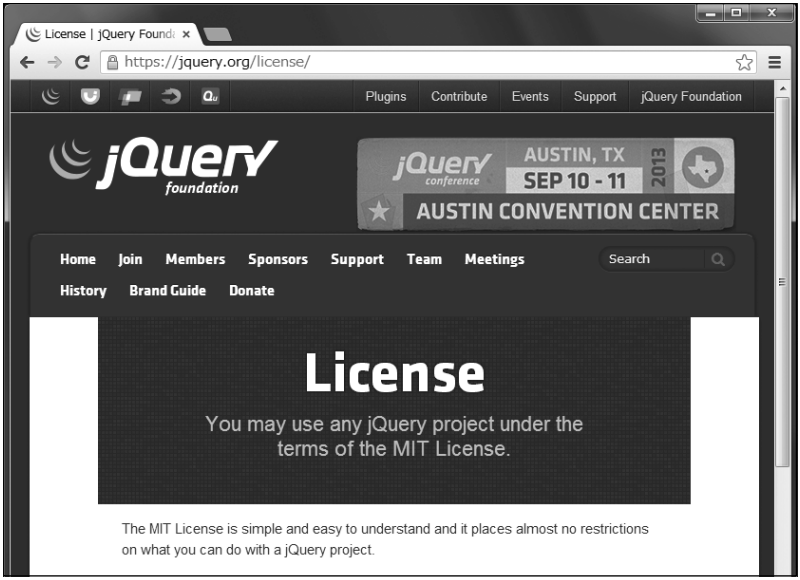
ただ、非同期通信処理はプログラムが複雑(入れ子/ネスト)になってしまうことがあります。コールバック地獄と呼ばれることもありますが、非同期通信処理を連続して行う場合は非常に面倒です。この面倒な処理もjQueryを使えば、かなりシンプルに記述し、利用することができます。HTML5関連の技術(API)を利用した場合、非同期処理はかなり多くなります。そのような非同期処理をシンプルに記述できるjQueryは非常に便利です。

COLUMN jQueryのライセンスについて

jQueryは1.8以降はMITライセンスですが、それ以前のバージョンではMITライセンスとGPL ver 2のデュアルライセンスになっています。

- jQuery License

  URL https://jquery.org/license/

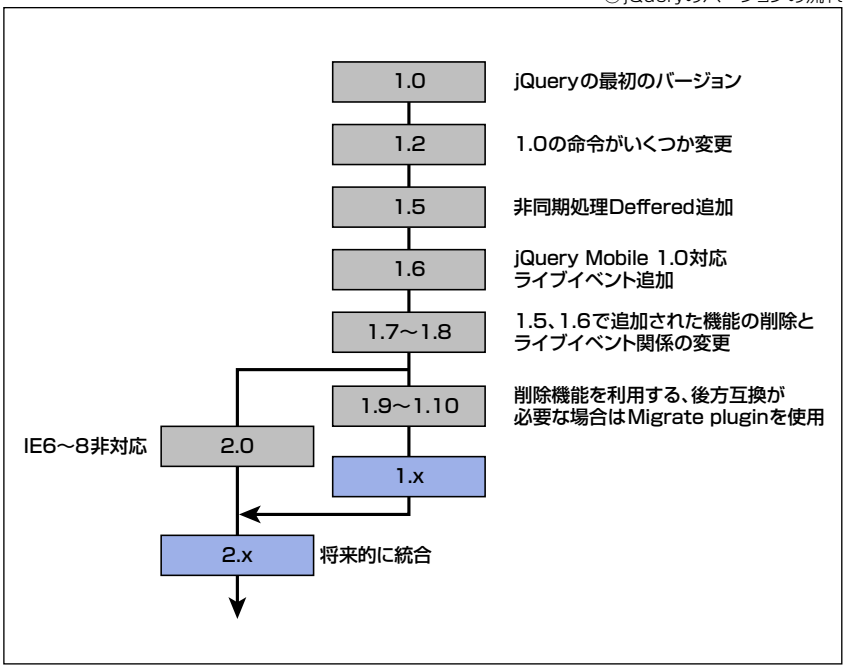


jQueryのバージョンについて

jQueryのバージョンの概要

jQueryにはいくつものバージョンがあります。2014年時点では、大きく分けると、2.0より前のバージョンと2.0以降のバージョンになります。2.0以降のバージョンではIE6～8には対応していません。IE8で動作させる場合には1.10など1.x系のバージョンを使う必要があります。

◉jQueryのバージョンの流れ

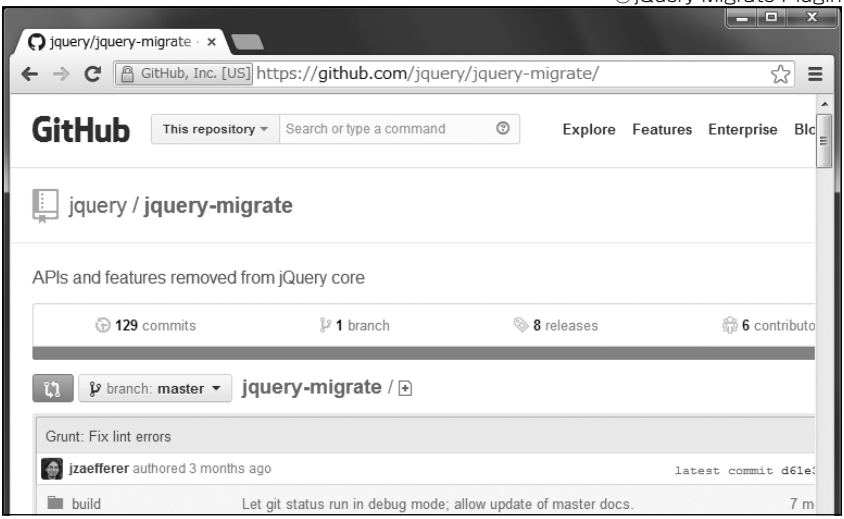


1.0系でも1.3、1.7～1.9で機能が追加・削除されており、使用する際には注意が必要です。特に1.9では将来的に2.0に統合するために廃止となった機能は完全に削除されています。削除された機能を利用したい場合や後方互換性が必要な場合にはjQuery Migrateプラグインが必要になります。

- jQuery Migrate Plugin

  URL https://github.com/jquery/jquery-migrate/

◉jQuery Migrate Plugin



IE6～8への対応が必要になる場合や過去のプログラムをバージョンアップして動作させる場合は、1.0系を使用することになります。場合によってはMigrateプラグインを加えて使用する必要があります。IE6～8への対応が不要で新規に作成する場合は、2.0系を使用するのがよいでしょう。jQueryのどのバージョンを使用するかは、jQuery UIやjQuery Mobileなど使用するプラグインなどにもよって異なります。プラグインによっては新しいバージョンでは動作しない場合があるためです。

なお、同じ命令（機能）でも1.0系と2.0系では動作が異なる場合があります。本書では動作が異なる命令の場合は、注意事項として記載しています。次にjQueryで廃止された機能一覧を示します。

◉廃止メソッド／プロパティ一覧

メソッド／プロパティ	廃止になったバージョン
jQuery.boxModel	1.3で廃止
jQuery.browser	
deffered.isRejected()	1.7で廃止
deffered.isResolved()	
.die()	
.sub()	
.live()	
.selector	
.andSelf()	1.8で廃止
deffered.pipe()	
.error()	
.load()	
.size()	
.toggle() [Mouse Events]	
.unload()	
.context	1.10で廃止

COLUMN **jQuery/jQuery UI/jQuery Mobileの最新バージョンを確認する**

jQuery、jQuery UI、jQuery Mobileの最新バージョンは、次のページで確認することができます。

- 最新バージョン

  URL https://code.jquery.com/

- すべてのバージョン

  URL https://code.jquery.com/jquery/

なお、以前は最新版としてjquery-latest.jsファイルが配布されていましたが、現在ではver 1.11.1に固定されています。つまり、以前と異なりjquery-latest.jsを指定しても最新版にはなりません。

- Don't Use jquery-latest.js

  URL http://blog.jquery.com/2014/07/03/dont-use-jquery-latest-js/

HTMLファイル内で、次のような指定は行わないようにしてください。このように指定するのではなく、バージョンを指定して読み込むようにしてください。

```
<script src="http://ajax.googleapis.com/ajax/libs/jquery/1/jquery.js"></script>
```

jQuery互換ライブラリについて

jQuery互換ライブラリの概要

jQueryには互換ライブラリが存在します。最も有名なのがZepto.jsです。jQueryもコンパクトなサイズですが、Zepto.jsではさらにコンパクトなファイルサイズになっています。カスタムビルドも可能で必要なモジュールだけを組み込んで使用することができます。

- Zepto.js

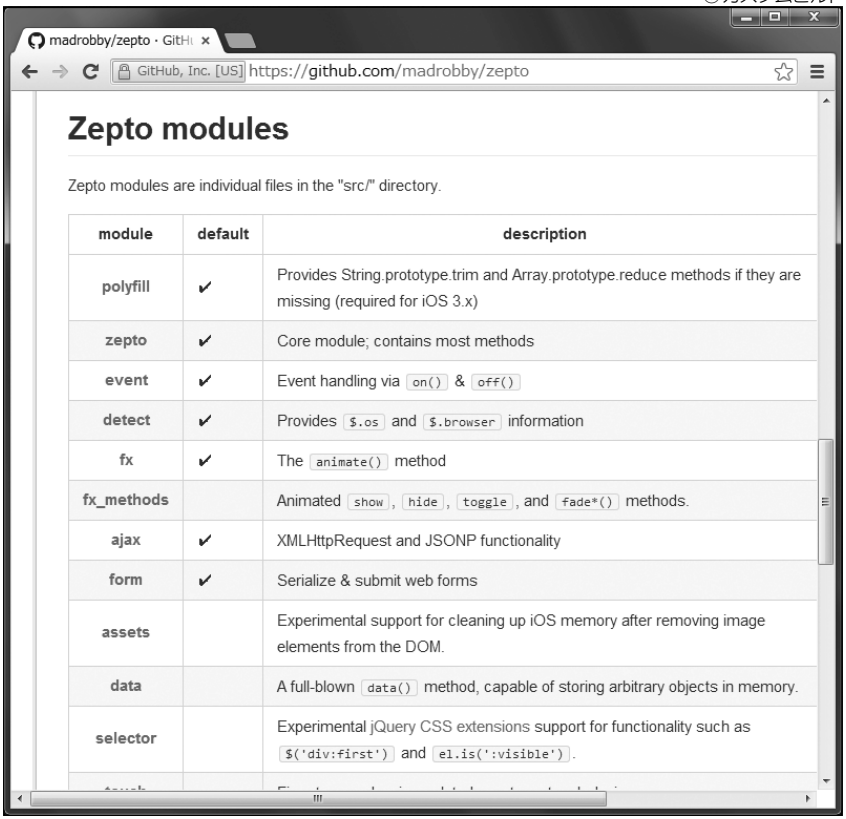
  URL http://zeptojs.com/

◉Zepto.js

- カスタムビルド

  URL https://github.com/madrobby/zepto

◉カスタムビルド



Zepto.jsはjQuery互換ですが、すべての機能に対応しているわけではありません。高度なアニメーション機能はありませんし、セレクタもすべて対応しているわけではありません（下表参照）。このため、現在はjQueryを使用しているが将来的にZepto.jsを使用する可能性がある場合は、何が対応していて何が対応していないかを把握しておく必要があります。また、豊富にあるjQueryライブラリもZepto.jsでは動作しないものがあります。

◉Zepto.jsでは使えない主なセレクタ

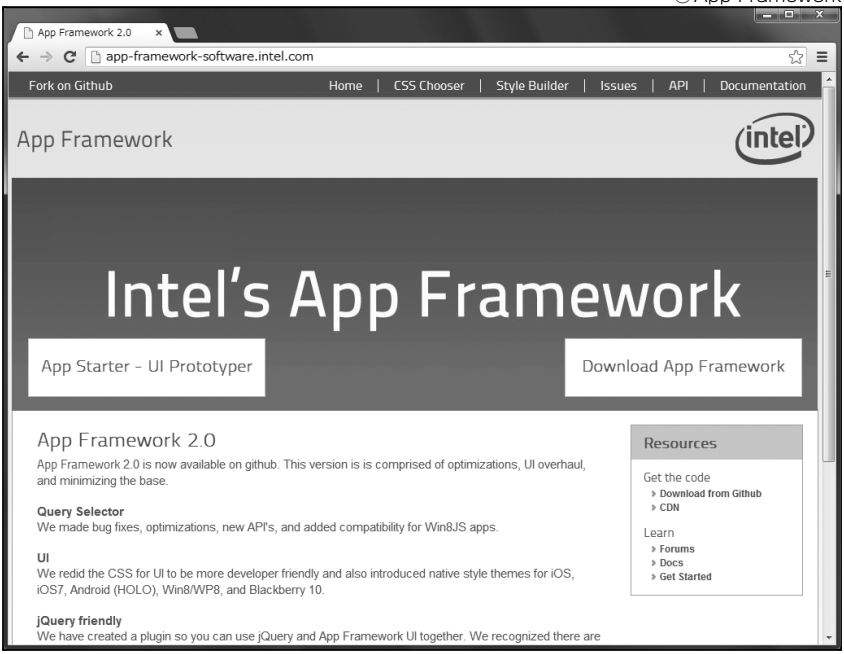
セレクタ	説明
$(":button")	buttonが非対応
$("h1:eq(0)")	eq()が非対応
$(":input:first")	firstが非対応

Zepto.jsよりも小さいファイルサイズの互換ライブラリがApp Framework（旧名jqmobi）です。App Frameworkはスマートフォン（iOS/Android）のために作成されたjQuery互換ライブラリです。

- App Framework

  URL http://app-framework-software.intel.com/

◉App Framework



App FrameworkはZepto.jsよりもjQueryとの互換性が少なくなっています。本当に基本的なセレクタ、DOM処理に限定されています。スマートフォンではCSS3アニメーションが使えるので、アニメーションやエフェクトなどはCSS3で処理し、jQueryと同じように手軽にDOM操作をするだけであればApp Frameworkでも充分です。

jQuery互換ライブラリへのスムーズな移行を行うには

jQuery互換ライブラリは、今後、増えるかもしれません。jQueryで作成したプログラムをコンパクトなZepto.js、App Frameworkで動作させなければならない場合もあるでしょう。将来的にjQuery互換ライブラリへの対応も考慮するのであれば、最初からHTML、JavaScript（jQuery）のコードに配慮が必要です。ここでは移行しやすくするために、どこに配慮したらよいかについて説明します。

まず、jQuery互換ライブラリといっても対応しているセレクタが異なっています。「class」属性で指定したクラス名や「id」属性で指定したクラス名に関しては互換性があり、いずれの互換ライブラリでも問題なく使用できます。これらに関しては他のjQuery互換ライブラリが登場しても問題なく使用できると思われます。

次に使用できるメソッドです。動作環境を調べる機能に関しては、互換性が低いためなるべく使用しない方が無難です。動作環境を調べたい場合にはjQueryではなく、Modernizrライブラリを使用する方がよいでしょう。

● Modernizr

URL http://modernizr.com/

◉Modernizr

次にパラメーターの違いです。jQuery互換ライブラリではサイズ軽減のためメソッドは特定のパラメーターしか対応していない場合があります。この場合、実行するとエラーにならないもののプログラムが動作しないか期待する動作にならない状態が発生します。対処方法としては各互換ライブラリのリファレンスを参照し、どのパラメーターが使えるのかを確認して、必要に応じてプログラムを書き換えます。

jQueryにはさまざまなイベントに関するメソッドがあります。「click()」や「mouseover()」などがありますが、イベント名をそのままメソッドにしているものは使用しないようにします。つまり、イベントの設定には「on()」メソッドを、イベントの解除には「off()」メソッドを使うということです。「on()」「off()」メソッドを使うことでさまざまなイベントに対応させることができます。

jQueryには簡単なアニメーションやエフェクト機能があります。Zepto.jsでは機能を組み込むことで最低限のエフェクトの対応はできます。エフェクトに関してはCSS3のアニメーション／変形機能などを使って行うようにします。その際、「$(～).css(プロパティ名, 値)」のように「css()」メソッドで直接CSSプロパティを指定してはいけません。エフェクトはあらかじめCSSファイルにクラスを定義しておき、そのクラス名を指定することでエフェクトを行うようにします。

すべて自前で作成したプログラムであれば対処のしようもありますが、プラグインを使用している場合は問題が発生することがあります。DOM要素を操作するなど簡単なものであれば動作するかもしれません。そうでない場合はプラグインの使用をあきらめ自前でプログラムを作成することになります。最初からプラグインを使用しない、互換ライブラリでの動作の確認がとれているものを使用するのが無難でしょう。

jQuery UIはユーザーインターフェースを扱うライブラリです。また、jQuery Mobileはモバイル用に作成されたフレームワークです。これらは互換ライブラリでは動作しません。なお、App Frameworkの場合はAF UIが用意されていますので、これを使用するようにします。

jQueryライブラリの読み込みについて

jQueryライブラリの読み込み方法

jQueryライブラリを読み込ませるにはHTMLの「script」要素を使います。HTML5であれば、次のように記述します。この場合、HTMLファイルと同じフォルダ／ディレクトリ内にある「jquery-2.1.0.min.js」ファイルが読み込まれます。以後の「script」要素で指定しているURLのバージョン表記部分は使用するバージョンに合わせて読み替えてください。

```
<script src="jquery-2.1.0.min.js"></script>
```

また、jQueryはCDN（Contents Delivery Network）でも配信されているため、直接サーバー上にあるjQueryライブラリのURLを指定して使用することもできます。なお、httpとhttps両方に対応する場合は「http://code.jquery.com/jquery-2.1.0.min.js」でなく、「http:」部分を省略して「//code.jquery.com/jquery-2.1.0.min.js」と記述する必要があります。なお、このように記述した場合はローカル上でのテストはできないので、サーバー上で動作テストを行う必要があります。

◉jQuery CDN

```
<script src="http://code.jquery.com/jquery-2.1.0.min.js"></script>
```

◉Google CDN

```
<script src="http://ajax.googleapis.com/ajax/libs/jquery/2.1.0/jquery.min.js"></script>
```

◉Microsoft CDN

```
<script src="http://ajax.aspnetcdn.com/ajax/jQuery/jquery-2.1.0.min.js"></script>
```

◉CDNJS CDN

```
<script src="http://cdnjs.cloudflare.com/ajax/libs/jquery/2.1.0/jquery.min.js"></script>
```

ユーザーインターフェース処理を行うjQuery UIライブラリを読み込む場合は、次のようになります。なお、jQueryファイルはjQuery UI関連ファイルよりも先に読み込ませておく必要があります。次の場合はHTMLファイルと同じフォルダ／ディレクトリを基準にして読み込みます。

```
<script src="themes/redmond/jquery-ui.css"></script>
<script src="jquery-ui.min.js"></script>
<script src="i18n/jquery-ui-i18n.min.js"></script>
```

CDNを利用する場合は、次のようになります。なお、CDNの場合は一部ファイルが提供されていないホストもあります。

◉jQuery CDN

```
<link rel="stylesheet" href="http://code.jquery.com/ui/1.10.3/themes/redmond/jquery-ui.css" />
<script src="http://code.jquery.com/ui/1.10.3/jquery-ui.min.js"></script>
```

◉Google CDN

```
<link rel="stylesheet"
  href="http://ajax.googleapis.com/ajax/libs/jqueryui/1.10.3/themes/redmond/jquery-ui.css" />
<script src="http://ajax.googleapis.com/ajax/libs/jqueryui/1.10.3/jquery-ui.min.js"></script>
<script src="http://ajax.googleapis.com/ajax/libs/jqueryui/1.10.3/i18n/jquery-ui-i18n.min.js">
</script>
```

◉Microsoft CDN

```
<link rel="stylesheet"
  href="http://ajax.aspnetcdn.com/ajax/jquery.ui/1.10.3/themes/redmond/jquery-ui.css" />
<script src="http://ajax.aspnetcdn.com/ajax/jquery.ui/1.10.3/jquery-ui.js"></script>
<script src="http://ajax.aspnetcdn.com/ajax/jquery.ui/1.10.3/i18n/jquery-ui-i18n.min.js">
</script>
```

◉CDNJS CDN

```
<script src="http://cdnjs.cloudflare.com/ajax/libs/jqueryui/1.10.3/jquery-ui.min.js">
</script>
```

jQuery Mobileの場合は、次のようになります。次の場合はHTMLファイルと同じフォルダ／ディレクトリにあるファイルを読み込みます。

```
<link rel="stylesheet" href="jquery.mobile-1.3.2.min.css" />
<script src="jquery-1.9.1.min.js"></script>
<script src="jquery.mobile-1.3.2.min.js"></script>
```

CDNを利用する場合は、次のようになります。

◉jQuery CDN

```
<link rel="stylesheet" href="http://code.jquery.com/mobile/1.3.2/jquery.mobile-1.3.2.min.css" />
<script src="http://code.jquery.com/jquery-1.9.1.min.js"></script>
<script src="http://code.jquery.com/mobile/1.3.2/jquery.mobile-1.3.2.min.js"></script>
```

◉Microsoft CDN

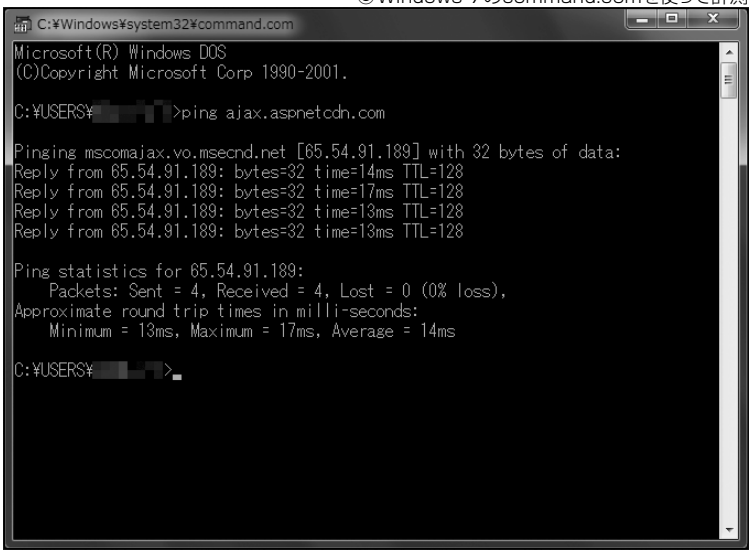
```
<link rel="stylesheet"
  href="http://ajax.aspnetcdn.com/ajax/jquery.mobile/1.3.1/jquery.mobile-1.3.1.min.css" />
<script src="http://ajax.aspnetcdn.com/ajax/jQuery/jquery-1.9.1.min.js"></script>
<script src="http://ajax.aspnetcdn.com/ajax/jquery.mobile/1.3.1/jquery.mobile-1.3.1.min.js">
</script>
```

COLUMN CDNで使われているサーバーの応答速度を調べるには

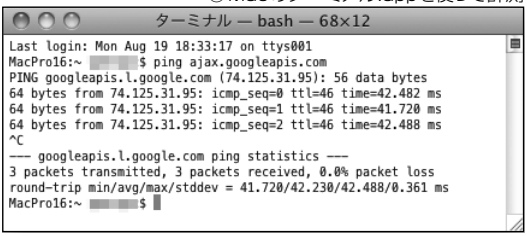
Macであればターミナル.appから、Windowsの場合はcommand.comから次のコマンドを実行すると、応答速度を調べることができます。

```
ping 調べたいサーバー名
```

◉Windows 7のcommand.comを使って計測



◉Macのターミナル.appを使って計測



SECTION-005

jQueryとDOM(ドキュメントオブジェクトモデル)について

jQueryを使ったDOM操作

jQueryはページ内の要素に簡単にアクセスして操作することができます。つまり、jQueryを使えば、DOM(ドキュメントオブジェクトモデル)操作を簡単にできるようになります。

ここでは簡単にjQueryを使ったDOM操作・イベント処理の基本について説明します。使用するHTMLはすべて共通で、次のようになっています。

SAMPLE CODE HTMLのコード(index.html)

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>Sample</title>
    <style>
      table { width: 400px; }
      .minus { color: red; }
      .minus2 { background-color:red; color: white; }
    </style>
    <script src="http://ajax.googleapis.com/ajax/libs/jquery/2.0.3/jquery.min.js"></script>
  </head>
  <body>
    <h1>昨年度の売り上げ</h1>
    <table border="1" id="sale">
      <thead><tr><th>日付</th><th>売り上げ</th></tr></thead>
      <tbody>
      <tr><td>1月</td><td>100万円</td></tr>
      <tr><td >2月</td><td class="minus">-80万円</td></tr>
      <tr><td>3月</td><td>140万円</td></tr>
      <tr><td>4月</td><td>1100万円</td></tr>
      <tr><td>5月</td><td class="minus">-300万円</td></tr>
      <tr><td>6月</td><td>900万円</td></tr>
      <tr><td>7月</td><td>450万円</td></tr>
      <tr><td>8月</td><td>1000万円</td></tr>
      <tr><td>9月</td><td>1900万円</td></tr>
      <tr><td>10月</td><td>2100万円</td></tr>
      <tr><td>11月</td><td>3500万円</td></tr>
      <tr><td>12月</td><td>5700万円</td></tr>
      </tbody>
    </table>
    <script src="js/sample.js"></script>
  </body>
</html>
```

jQueryの基本は「$()」

jQueryでは要素へのアクセスは「$()」を使います。「$」はjQueryオブジェクトのエイリアスです。「$()」の代わりに「jQuery()」と記述しても同じです。jQueryや$ではなく他の名前に変更することもできます。「$()」のパラメーターにはCSSで指定できるセレクタを文字列で指定します。最も簡単なところで、見出し要素である「h1」の内容(文字)を書き換えてみましょう。CSSで「h1」要素の色を変える場合は、次のように記述します。

```
h1 { color: red; }
```

この「h1」がセレクタです。この「h1」を「$()」のパラメーターに指定します。つまり、「$("h1")」と記述します。これで「h1」要素を指定したことになります。「h1」要素の文字を書き換えるには「text()」メソッドを使います。「text()」メソッドはパラメーターに指定した文字を要素内に出力します。次のようにすると、「h1」要素に表示される文字が「2012年度の売り上げ」になります。

```
$("h1").text("2012年度の売り上げ");
```

このHTMLでは「h1」要素は1つですが、複数の要素の内容を書き換える場合も同じように書きます。次のようにするとテーブルの「td」要素の内容が「現在処理中」の文字に書き換わります。

```
$("td").text("現在処理中");
```

すべて書き換えるのではなく特定の要素を指定することもできます。jQueryのセレクタは強力なので要素の出現順で指定したり奇数、偶数順で指定することもできます。次のように「eq()」を使うと出現順を指定することができ、1月の売り上げの文字だけが書き換わります。

```
$("td:eq(1)").text("現在処理中");
```

売り上げを示す右側の「td」要素をまとめて書き換える場合は、次のようになります。

```
$("td:nth-child(2n)").text("現在処理中");
```

「nth-child(2n)」のように、計算式を指定して要素の順番を指定することもできます。

スタイルシートの変更

jQueryでは要素のスタイルシートを変更することもできます。スタイルシートを変更するには「css()」メソッドを使います。「css()」メソッドの最初のパラメーターに変更したいスタイルシート名を、2番目のパラメーターに設定したい値を指定します。次の例ではID名が「sale」の要素(今回の場合はテーブル全体)の背景色がグレー(灰色)になります。

```
$("#sale").css("background-color", "gray");
```

表には売り上げがマイナスになってる箇所があります。赤色の文字になるようにスタイルシートクラスが割り当てられています。このスタイルシートを削除して新たなスタイルシートを設定することができます。スタイルシートを削除するには「removeClass()」メソッドを使い、パラメーターには

削除したいスタイルシートのクラス名を指定します。スタイルシートを追加するには「addClass()」メソッドを使い、パラメーターには追加したいスタイルシートのクラス名を指定します。

次の場合では、「minus」クラスを持つ要素の「minus」クラスを削除後に「minus2」クラスを追加しています。

```
$(".minus").removeClass("minus").addClass("minus2");
```

スタイルシートクラスを切り替えて使う場合には「toggleClass()」メソッドが用意されています。パラメーターには追加・削除したいスタイルシートクラス名を指定します。「toggleClass()」メソッドは呼び出すたびに自動的に判断し切り替えてくれます。

次の場合では「minus2」クラスを1秒ごとに追加・削除します。

```
setTimeout(function(){
  $(".minus").toggleClass("minus2");
  setTimeout(arguments.callee, 1000);
}, 10);
```

イベントの処理

jQueryではクリックなどのイベントも手軽にハンドリング(組み込み)できるようになっています。クリックされたら何か処理を行うには「click()」メソッドを使います。パラメーターにはクリック時に呼び出す関数(イベントハンドラ)を指定します。「click(xyz)」とすると、クリック時に「xyz」関数が呼び出されます。同じ関数を他から呼び出さないのであれば、「click()」のパラメーターに無名関数を指定します。

次の例では「td」要素がクリックされた場合に、その要素内の文字をアラートダイアログに表示します。

```
$("td").click(function(){
  alert($(this).text());
});
```

マウスが要素の上に乗った場合、離れた場合に処理を行うこともできます。この場合は「hover()」メソッドを使うことで簡単に実現できます。「hover()」メソッドの最初のパラメーターにマウスが乗った場合に呼び出す関数(イベントハンドラ)を、2番目のパラメーターにマウスが要素から離れた場合に呼び出す関数を指定します。

次の例ではマウスが「tr」要素に乗った場合にスタイルシートクラス名「minus2」を追加し、マウスが離れたら削除しています。

```
$("tr").hover(function(){
  $(this).addClass("minus2");
}, function(){
  $(this).removeClass("minus2");
});
```

jQueryでは1.6以降で後から追加した要素にも自動的にイベントを付与する機能が追加されました。動的に追加されるイベントに関してはメソッドの追加や廃止がありましたが、1.8／2.0以降では「on()」メソッドと「off()」メソッドに統合されています。

最初のイベントの例として「$("td").click(～)」と記述していました。要素が少ない場合はよいのですが、要素の数が多くなると要素ごとにイベントを割り当てるため、メモリや処理効率が悪くなります。そこで、個別の要素でイベントを処理するのではなく、親要素でイベントを一括して処理するようにします。「on()」メソッドは親要素に対してイベントを割り当てます。

次の例では親要素である「table」(ID名は「sale」)に「click」イベントを割り当て、「click」イベントを補足する要素として「td」を指定しています。このようにすると動的に「td」要素を増やしても処理も遅くならずメモリ消費も抑えられます。

```
$("#sale").on("click", "td", function(){
  alert($(this).text());
});
```

CHAPTER 02

jQuery Core（コア／基本）

jQueryオブジェクトを指定する

ブラウザの対応	IE6以降、Chrome、Safari 5以降、Firefox 4以降、iOS4以降、Android 2以降
jQueryの対応	1.0以降
Zepto.jsの対応	1.0以降
App Frameworkの対応	1.2以降

ここでは、jQueryオブジェクトを指定する方法について解説します。

SAMPLE CODE HTMLのコード(index.html)

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>Sample</title>
    <script src="http://ajax.googleapis.com/ajax/libs/jquery/2.1.0/jquery.min.js"></script>
    <script src="js/sample.js"></script>
  </head>
  <body>
    <h1>jQueryオブジェクトへのアクセス</h1>
    <output></output>
  </body>
</html>
```

SAMPLE CODE JavaScriptのコード(sample.js)

```
// jQueryオブジェクトを使って処理
jQuery(function(){
  // $オブジェクトを使って処理
  $("output").text("処理は完了しました");
  // jQueryオブジェクトを使って処理
  jQuery("output").css("border", "2px solid red");
});
```

jQueryオブジェクトを利用してページ上の要素にアクセスし、文字の出力、スタイルシートの変更を行っています。

ONEPOINT　jQueryオブジェクトへのアクセス方法について

jQueryを利用する場合、jQueryオブジェクトを介して各種オブジェクト、DOM要素へのアクセスを行います。また、jQueryオブジェクトだけでなく、「$」を使っても同様にアクセスできます。「$」は「$.～」のようにして使用する場合と「$()」メソッドとして使う場合があります。「$.～」は動作環境のチェックなどに使われます。「$()」メソッドはDOMを指定するために使われます。

「$()」メソッドには文字列でセレクタを指定するだけでなく、「this」や「document」など、JavaScriptのオブジェクトおよびDOM要素（とその集合）を指定することもできます。たとえば、次のような指定が可能です。

```
$("div#cr")
$(this)
$(document)
$(document.getElementsByTagName("span"))
```

なお、jQuery互換ライブラリのzepto.jsを使用する場合、サンプルは動作しません。これはjQueryオブジェクトが定義されていないためで、「jQuery（～）」の代わりに「$（～）」として「$」を使用してください。また、「$」が他のライブラリ（Prototype.js、MooToolsなど）で使用されており、他の名前で代用する必要がある場合は「jQuery.noConflict()」を使います。詳しい解説については62ページを参照してください。

関連項目 ▶▶▶
● 「$」が他のライブラリと干渉しないようにする …… p.62

「$」が他のライブラリと干渉しないようにする

ブラウザの対応	IE6以降、Chrome、Safari 5以降、Firefox 4以降、iOS4以降、Android 2以降
jQueryの対応	1.0以降
Zepto.jsの対応	×
App Frameworkの対応	×

ここでは、「$」が他のライブラリと干渉しないようにする方法について解説します。

SAMPLE CODE HTMLのコード(index.html)

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>Sample</title>
    <script src="http://ajax.googleapis.com/ajax/libs/jquery/2.1.0/jquery.min.js"></script>
    <script src="js/sample.js"></script>
  </head>
  <body>
    <h1>$の干渉を回避してアクセス</h1>
    <output id="result"></output>
  </body>
</html>
```

SAMPLE CODE JavaScriptのコード(sample.js)

```
// jQueryオブジェクトを$でなくjqを使ってアクセスする
var jq = jQuery.noConflict();
// jqオブジェクトを使って処理
jq(function(){
  // jqオブジェクトを使って処理
  jq("#result").text("処理は完了しました");
  // jQueryオブジェクトを使って処理
  jQuery("#result").css("border", "2px solid red");
});
```

HINT

Zepto.jsでは、「noConflict()」メソッドは用意されていません。

「$」でなく、新たに割り当てた「jq」オブジェクトを利用してページ上の要素にアクセスし、文字の出力、スタイルシートの変更を行っています。

ONEPOINT 「$」の干渉を回避するには「noConflict()」メソッドを使う

jQueryはデフォルトの状態では、「jQuery」と「$」がグローバル領域に割り当てられています。jQueryライブラリ単体で使用する場合は問題になりませんが、prototype.jsやMooToolsなど他のライブラリを併用した場合、同じ「$」が使われているため、プログラムが正しく動作しなくなります。この場合、jQueryでは「noConflict()」メソッドを使うことで「$」の干渉を回避できます。また、ライブラリの読み込み順序は関係なく、先にjQueryライブラリを読み込んでも、後から読み込んでも問題なく動作します。

「noConflict()」メソッドは戻り値として「jQuery」オブジェクトを返します。つまり、「var xyz = jQuery.noConflict()」とすると、変数「xyz」に「jQuery」オブジェクトが入ります。その後は「$」の代わりに「xyz」を「jQuery」オブジェクトとして利用することができます。「noConflict()」メソッドのパラメーターに「true」を指定するとグローバル領域に残っている「jQuery」オブジェクトも削除されます。

COLUMN 無名関数を利用して「$」を渡す

jQueryを利用しているライブラリによっては「$」が内部で使用されていることがあります。この場合、次のように無名関数を利用して「$」を渡す方法があります。

```
(function($) {
  $(function() {
  各種処理／ライブラリの利用
  });
})(jQuery.noConflict(true));
```

実際の例としては、次のようになります。

```
// $をalertとして割り当てる
var $ = alert;
// jQueryオブジェクトを$として渡す
(function($) {
  $(function() {
  // jqオブジェクトを使って処理
  $("output").text("処理は完了しました");
  // prototype.jsの$を使って処理
  $("#result").css("color", "red");
  });
})(jQuery.noConflict(true));
// アラートダイアログを表示
$("衝突回避");
```

関連項目 ▶▶▶

● jQueryオブジェクトを指定する …… p.60

CHAPTER 03

セレクタ関係

すべての要素を指定する

ブラウザの対応	IE6以降、Chrome、Safari 5以降、Firefox 4以降、iOS4以降、Android 2以降
jQueryの対応	1.0以降
Zepto.jsの対応	1.0以降
App Frameworkの対応	1.2以降

ここでは、すべての要素を指定する方法を解説します。

SAMPLE CODE HTMLのコード(index.html)

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>Sample</title>
    <script src="http://ajax.googleapis.com/ajax/libs/jquery/2.1.0/jquery.min.js"></script>
    <script src="js/sample.js"></script>
  </head>
  <body>
    <h1>全ての要素を指定する</h1>
    <table border="1" id="sale">
      <thead><tr><th>日付</th><th>売り上げ</th></tr></thead>
      <tbody>
      <tr><td>1月</td><td>100円</td></tr>
      <tr><td>2月</td><td>200円</td></tr>
      <tr><td>3月</td><td>140円</td></tr>
      <tr><td>4月</td><td>1100円</td></tr>
      <tr><td>5月</td><td>1300円</td></tr>
      <tr><td>6月</td><td>900円</td></tr>
      <tr><td>7月</td><td>450円</td></tr>
      <tr><td>8月</td><td>1000円</td></tr>
      <tr><td>9月</td><td>1900円</td></tr>
      <tr><td>10月</td><td>2100円</td></tr>
      <tr><td>11月</td><td>3500円</td></tr>
      <tr><td>12月</td><td>5700円</td></tr>
      </tbody>
    </table>
  </body>
</html>
```

SAMPLE CODE JavaScriptのコード(sample.js)

```
$(function(){
  $("*").css("border", "1px dotted red");
});
```

ページが表示されると、「body」要素以下すべての要素のまわりに赤い点線枠が表示されます。

ONEPOINT　すべての要素を指定するには「$()」のパラメーターに「"*"」を指定する

すべての要素を指定するには「$()」のパラメーターに「"*"」を指定します。CSSの全称セレクタの「*」と同じです。「$("*")」とすると、「html」要素や「head」要素、「meta」要素なども対象になります（下図参照）。

$("*")	②～㉓行までのすべての要素が対象になる
$("body *")	⑩～㉑行までのすべての要素が対象になる
$("tbody *")	⑯～⑳行までのすべての要素が対象になる
$("tbody tr:eq(1) *")	⑰～⑱行までのすべての要素が対象になる

```
① <!DOCTYPE html>
② <html>
③   <head>
④     <meta charset="utf-8">
⑤     <title>Sample</title>
⑥     <script src="http://ajax.googleapis.com/ajax/libs/jquery/2.0.3/jquery.min.js"></script>
⑦     <script src="js/sample.js"></script>
⑧   </head>
⑨   <body>
⑩     <h1> 全ての要素を指定する </h1>
⑪     <table>
⑫       <thead>
⑬         <tr><th> 日付 </th><th> 売り上げ </th></tr>
⑭       </thead>
⑮       <tbody>
⑯         <tr><td>1 月 </td><td>100 円 </td></tr>
⑰         <tr><td>2 月 </td><td>
⑱           <span style="color:red">200 円 </span></td></tr>
⑲         <tr><td>3 月 </td><td>140 円 </td></tr>
⑳       </tbody>
㉑     </table>
㉒   </body>
㉓ </html>
```

COLUMN 「document.getElementsByTagName()」を使って サンプルと同じ処理を行う

jQueryを使わずに純粋なJavaScriptのみで書いた場合、「document.getElementsByTagName("*")として指定できます。「document.getElementsByTagName()」を使ってサンプルと同じ処理を行う場合、次のようになります。

```
var all = document.getElementsByTagName("*");
for(var i=0; i<all.length; i++){
  all[i].style.border = "1px dotted red";
}
```

「*」はページ内すべての要素を対象にするのではなく、次のようにすると「tbody」要素内にあるすべての要素が対象になります。

```
$("tbody *").css("border", "1px dotted red");
```

関連項目 ▶▶▶

● 特定の要素を指定する …… p.80

SECTION-009

ID名で指定する

ブラウザの対応	IE6以降、Chrome、Safari 5以降、Firefox 4以降、iOS4以降、Android 2以降
jQueryの対応	1.0以降
Zepto.jsの対応	1.0以降
App Frameworkの対応	1.2以降

ここでは、ID名で指定する方法について解説します。

SAMPLE CODE HTMLのコード(index.html)

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>Sample</title>
    <script src="http://ajax.googleapis.com/ajax/libs/jquery/2.1.0/jquery.min.js"></script>
    <script src="js/sample.js"></script>
  </head>
  <body>
    <h1>ID名で要素を指定する</h1>
    <form>
      <input type="button" value="IDがnoteの背景色を変更">
    </form>
    <div id="comment">特にコメントはありませんでした</div>
    <div id="note">一部には反対意見もあったとのことです</div>
    <div class="note">全体としては賛成意見の方が多かったようです</div>
  </body>
</html>
```

SAMPLE CODE JavaScriptのコード(sample.js)

```
$(function(){
  $(":button").click(function(){
    // noteのID名を持つ要素の背景色をオレンジ色にする
    $("#note").css("background", "orange");
  });
});
```

ボタンをクリックすると、ID名が「note」の要素がオレンジ色の背景色になります。

ONEPOINT　ID名で指定するには「$()」に「#ID名」を指定する

ID名で要素を指定するには、「$()」に「#ID名」を指定します。複数のID名を指定する場合は「$("#abc,#def,#xyz")」のように「,」(カンマ)で区切って指定します。

COLUMN　より高速に処理するには

ID名で指定すると、より高速に処理されます。さらに高速に処理する必要がある場合はJavaScriptネイティブで記述する方法もあります。たとえば、ID名が「abc」であれば、次のように記述できます。

```
document.getElementById("abc")
```

この要素の背景色をオレンジ色にするには、次のようになります。

```
document.getElementById("abc").style.backgroundColor = "orange";
```

もし、この要素に何度もアクセスするような場合、さらに高速化することができます。これは次のように要素への参照を変数(オブジェクト)に入れておきます。

```
var ele = document.getElementById("abc");
```

このようにしておけば、次のようにして要素のスタイルシートなどにアクセスできます。

```
ele.style.backgroundColor = "orange";
```

関連項目 ▶▶▶
- すべての要素を指定する …… p.66
- クラス名で指定する …… p.71
- 属性名で指定する …… p.73

クラス名で指定する

ブラウザの対応	IE6以降、Chrome、Safari 5以降、Firefox 4以降、iOS4以降、Android 2以降
jQueryの対応	1.0以降
Zepto.jsの対応	1.0以降
App Frameworkの対応	1.2以降

ここでは、クラス名で指定する方法について解説します。

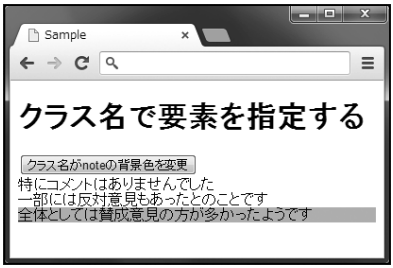
SAMPLE CODE HTMLのコード(index.html)

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>Sample</title>
    <script src="http://ajax.googleapis.com/ajax/libs/jquery/2.1.0/jquery.min.js"></script>
    <script src="js/sample.js"></script>
  </head>
  <body>
    <h1>クラス名で要素を指定する</h1>
    <form>
      <input type="button" value="クラス名がnoteの背景色を変更">
    </form>
    <div id="comment">特にコメントはありませんでした</div>
    <div id="note">一部には反対意見もあったとのことです</div>
    <div class="note">全体としては賛成意見の方が多かったようです</div>
  </body>
</html>
```

SAMPLE CODE JavaScriptのコード(sample.js)

```
$(function(){
  // ボタンにclickイベントを割り当てる
  $(":button").click(function(){
    // noteのクラス名を持つ要素の背景色をオレンジ色にする
    $(".note").css("background", "orange");
  });
});
```

ボタンをクリックすると、クラス名が「note」の要素がオレンジ色の背景色になります。



ONEPOINT クラス名で指定するには「$()」に「.クラス名」を指定する

クラス名で要素を指定するには「$()」に「.クラス名」を指定します。複数のクラス名を指定する場合は「$(".abc,.def,.xyz")」のように「,」(カンマ)で区切って指定します。

クラス名で指定した要素内にある「div」要素を指定する場合は「$(".note div")」のようにクラス名の後に半角空白で区切って要素名を指定します。また、指定したクラス名の要素内にあるすべての要素を対象にしたい場合は「$(".note *")」のように「*」(全称セレクタ)と組み合わせます。

COLUMN jQueryを使わない場合のコード

サンプルをjQueryを使わずに純粋なJavaScriptのみで書いた場合は、次のようになります。

```
window.addEventListener("load", function(){
  document.querySelectorAll("input[type=button]")[0]
    .addEventListener("click", function(){
      var ele = document.querySelectorAll(".note");
      for(var i=0; i<ele.length; i++){
        ele[i].style.background = "orange";
      }
  });
});
```

また、「document.querySelectorAll(".note");」の代わりに、「document.getElementsByClassName("note");」を使うこともできます。ただし、「getElementsByClassName()」で得られる要素のノードは動的になっているので使用する際には注意が必要です。

関連項目 ▶▶▶

- すべての要素を指定する …… p.66
- ID名で指定する …… p.69
- 属性名で指定する …… p.73

属性名で指定する

ブラウザの対応	IE6以降、Chrome、Safari 5以降、Firefox 4以降、iOS4以降、Android 2以降
jQueryの対応	1.0以降
Zepto.jsの対応	1.0以降
App Frameworkの対応	1.2以降

ここでは、属性名で指定する方法について解説します。

SAMPLE CODE HTMLのコード(index.html)

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>Sample</title>
    <script src="http://ajax.googleapis.com/ajax/libs/jquery/2.1.0/jquery.min.js"></script>
    <script src="js/sample.js"></script>
  </head>
  <body>
    <h1>特定の属性名を持つ要素を指定する</h1>
    <form>
      <input type="button" value="src属性がある要素に枠を付ける">
    </form>
    <img src="images/0.jpg" width="128" height="72" alt="">
    <img src="images/1.jpg" width="128" height="72" alt="">
    <div>Sample text (srcなし)</div>
    <div src="dummy">Sample text (srcあり)</div>
  </body>
</html>
```

SAMPLE CODE JavaScriptのコード(sample.js)

```
$(function(){
  // ボタンにイベントを割り当て
  $(":button").click(function(){
    // src属性がある要素に赤い枠をつける
    $("[src]").css("border", "4px solid red");
  });
});
```

ボタンをクリックすると、「src」属性がある「img」要素（画像）と「div」要素に赤い枠が追加されます。

ONEPOINT　属性名で指定するには「$()」に「[属性名]」を指定する

jQueryで属性名で指定するには、「$()」に「[属性名]」を指定します。「$("[src]")」であれば、「src」属性を持つすべての要素が対象になります。「$("img[src]")」のように指定すると、「img」要素で「src」属性を持つ要素だけが対象になります。

COLUMN **jQueryを使わない場合のコード**

サンプルをjQueryを使わずに純粋なJavaScriptのみで書いた場合は、次のようになります。

```
window.addEventListener("load", function(){
  document.querySelectorAll("input[type=button]")[0]
    .addEventListener("click", function(){
      var ele = document.querySelectorAll("[src]");
      for(var i=0; i<ele.length; i++){
        ele[i].style.border = "4px solid red";
      }
  });
});
```

関連項目 ▶▶▶

- 複数の属性セレクタを指定する …… p.78
- 属性が指定した文字列から始まる要素を指定する …… p.143
- 属性が指定した文字列で終わる要素を指定する …… p.147
- 属性が指定した内容になっている要素を指定する …… p.150
- 属性が指定した内容ではない要素を指定する …… p.154
- 属性が指定した単語を含む要素を指定する …… p.157
- 属性が指定した文字列を含む要素を指定する …… p.160

複数の要素を指定する

ブラウザの対応	IE6以降、Chrome、Safari 5以降、Firefox 4以降、iOS4以降、Android 2以降
jQueryの対応	1.0以降
Zepto.jsの対応	×
App Frameworkの対応	1.2以降

ここでは、複数の要素を指定する方法について解説します。

SAMPLE CODE HTMLのコード(index.html)

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>Sample</title>
    <script src="http://ajax.googleapis.com/ajax/libs/jquery/2.1.0/jquery.min.js"></script>
    <script src="js/sample.js"></script>
  </head>
  <body>
    <h1>複数の要素を指定する</h1>
    <div>
      <div id="cr">C&R研究所</div>
      <div class="address">新潟県</div>
      <div>業務：<span>出版</span></div>
      <div>マスコット：<span>Lucky</span></div>
    </div>
  </body>
</html>
```

SAMPLE CODE JavaScriptのコード(sample.js)

```
$(function(){
  // #cr, .class, spanの文字の色を赤色にする
  $("#cr, .address, span").css("color", "red");
});
```

複数の要素（#cr、.class、span）の文字の色が赤色になります。

ONEPOINT　複数の要素を指定するには「,」（カンマ）で区切って列記する

複数の要素を指定するには、「,」（カンマ）で区切って列記します。たとえば、「#mz」と「h1」要素を同時に指定するには、「$("#mz, h1")」のように指定します。複数のセレクタを「$("#cr, div, span.note, input[type='radio']")」のようにまとめて指定することもできます。複数のセレクタを指定した場合、該当する要素がない場合があります。この場合、対象となる要素が返されるだけで、対象となる要素が存在しなくてもエラーにはなりません。

COLUMN　jQueryを使わない場合のコード

サンプルをjQueryを使わずに純粋なJavaScriptのみで書いた場合は、次のようになります。

```
window.addEventListener("load", function(){
  var ele = document.querySelectorAll("#cr, .address, span");
  for(var i=0; i<ele.length; i++){
    ele[i].style.color = "red";
  }
});
```

関連項目 ▶▶▶

- 複数の属性セレクタを指定する …… p.78
- 特定の要素を指定する …… p.80

SECTION-013

複数の属性セレクタを指定する

ブラウザの対応	IE6以降、Chrome、Safari 5以降、Firefox 4以降、iOS4以降、Android 2以降
jQueryの対応	1.0以降
Zepto.jsの対応	1.0以降
App Frameworkの対応	1.2以降

ここでは、複数の属性セレクタを指定する方法について解説します。

SAMPLE CODE HTMLのコード(index.html)

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>Sample</title>
    <script src="http://ajax.googleapis.com/ajax/libs/jquery/2.1.0/jquery.min.js"></script>
    <script src="js/sample.js"></script>
  </head>
  <body>
    <h1>複数の属性セレクタを指定する</h1>
    <img src="images/0.jpg" width="128" height="72" alt="">
    <img src="images/1.jpg" width="128" height="72" class="note" alt="dam">
    <img src="images/2.jpg" width="128" height="72" alt="">
    <img src="images/3.jpg" width="128" height="72" class="note" alt="">
  </body>
</html>
```

SAMPLE CODE JavaScriptのコード(sample.js)

```
$(function(){
  // img要素でclass属性を持ち、なおかつalt属性がdamの要素に赤枠を追加する
  $("img[class][alt='dam']").css("border", "4px solid red");
});
```

「img」要素で「class」属性があり、なおかつ「alt」属性の内容が「dam」の要素だけ赤枠が表示されます。

ONEPOINT　複数の属性セレクタを指定するには「[属性名1][属性名2]」のように列記する

jQueryで複数の属性セレクタを指定するには、「[属性名1][属性名2]」のように列記します。条件なども記述できます。すべての条件を満たした要素だけが対象になります。

COLUMN　jQueryを使わない場合のコード

サンプルをjQueryを使わずに純粋なJavaScriptのみで書いた場合は、次のようになります。

```
window.addEventListener("load", function(){
  var ele = document.querySelectorAll("img[class][alt='dam']");
  for(var i=0; i<ele.length; i++){
    ele[i].style.border = "4px solid red";
  }
});
```

関連項目 ▶▶▶

- 複数の要素を指定する …… p.76
- 特定の要素を指定する …… p.80

特定の要素を指定する

ブラウザの対応	IE6以降、Chrome、Safari 5以降、Firefox 4以降、iOS4以降、Android 2以降
jQueryの対応	1.0以降
Zepto.jsの対応	1.0以降
App Frameworkの対応	1.2以降

ここでは、特定の要素を指定する方法について解説します。

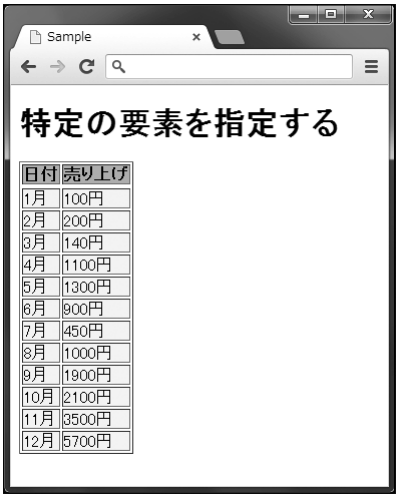
SAMPLE CODE HTMLのコード(index.html)

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>Sample</title>
    <script src="http://ajax.googleapis.com/ajax/libs/jquery/2.1.0/jquery.min.js"></script>
    <script src="js/sample.js"></script>
  </head>
  <body>
    <h1>特定の要素を指定する</h1>
    <table border="1" id="sale">
      <thead>
        <tr><th>日付</th><th>売り上げ</th></tr>
      </thead>
      <tbody>
      <tr><td>1月</td><td>100円</td></tr>
      <tr><td>2月</td><td>200円</td></tr>
      <tr><td>3月</td><td>140円</td></tr>
      <tr><td>4月</td><td>1100円</td></tr>
      <tr><td>5月</td><td>1300円</td></tr>
      <tr><td>6月</td><td>900円</td></tr>
      <tr><td>7月</td><td>450円</td></tr>
      <tr><td>8月</td><td>1000円</td></tr>
      <tr><td>9月</td><td>1900円</td></tr>
      <tr><td>10月</td><td>2100円</td></tr>
      <tr><td>11月</td><td>3500円</td></tr>
      <tr><td>12月</td><td>5700円</td></tr>
      </tbody>
    </table>
  </body>
</html>
```

SAMPLE CODE JavaScriptのコード(sample.js)

```
$(function(){
  // th要素を選択する
  $("th").css("background-color", "orange");
  // td要素を選択する
  $("td").css("background-color", "yellow");
});
```

ページが表示されると、テーブル内の「th」要素の背景色がオレンジ色に、「td」要素が黄色の背景色になります。



ONEPOINT 特定の要素を指定するには「$()」のパラメーターに要素名を文字列で指定する

特定の要素を指定するには、「$()」のパラメーターに要素名を文字列で指定します。たとえば、「div」要素であれば「$("div")」のように指定します。複数の要素を指定する場合には「$("div, h1, span")」のように、「,」(カンマ)で区切って列記します。

また、「$("div, table:first")」のように組み合わせて指定することもできます。

COLUMN jQueryを使わない場合のコード

サンプルをjQueryを使わずに純粋なJavaScriptのみで書いた場合は、次のようになります。なお、次のコードでは要素を取得するのに「getElementsByTagName()」メソッドを使っています。これ以外にも「querySelector()」や「querySelectorAll()」メソッドを使って要素を取得することもできます。

```
window.addEventListener("load", function(){
  var ele = document.getElementsByTagName("th");
  for(var i=0; i<ele.length; i++){
    ele[i].style.background = "orange";
  }
  var ele = document.getElementsByTagName("td");
  for(var i=0; i<ele.length; i++){
    ele[i].style.background = "yellow";
  }
});
```

関連項目 ▶▶▶

● すべての要素を指定する…… p.66

空の要素を指定する

ブラウザの対応	IE6以降、Chrome、Safari 5以降、Firefox 4以降、iOS4以降、Android 2以降
jQueryの対応	1.0以降
Zepto.jsの対応	1.0以降
App Frameworkの対応	1.2以降

ここでは、空の要素を指定する方法について解説します。

SAMPLE CODE HTMLのコード(index.html)

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>Sample</title>
    <script src="http://ajax.googleapis.com/ajax/libs/jquery/2.1.0/jquery.min.js"></script>
    <script src="js/sample.js"></script>
  </head>
  <body>
    <h1>空の要素を指定する</h1>
    <table border="1" id="sale">
      <thead>
        <tr><th>日付</th><th>売り上げ</th></tr>
      </thead>
      <tbody>
      <tr><td>1月</td><td></td></tr>
      <tr><td>2月</td><td></td></tr>
      <tr><td>3月</td><td></td></tr>
      <tr><td>4月</td><td>1100円</td></tr>
      <tr><td>5月</td><td>1300円</td></tr>
      <tr><td>6月</td><td>900円</td></tr>
      <tr><td>7月</td><td>450円</td></tr>
      <tr><td>8月</td><td>100円</td></tr>
      <tr><td>9月</td><td>70円</td></tr>
      <tr><td>10月</td><td>20円</td></tr>
      <tr><td>11月</td><td>0円</td></tr>
      <tr><td>12月</td><td></td></tr>
      </tbody>
    </table>
  </body>
</html>
```

SAMPLE CODE JavaScriptのコード(sample.js)

```
$(function(){
  // td要素が空の場合に内容を変更する
  $("#sale tbody td:empty").text("売り上げデータなし");
});
```

ページが表示されると、テーブル内の空の要素の内容が「売り上げデータなし」に書き換えられます。

ONEPOINT 空の要素を指定するには「:empty」を指定する

jQueryでは、セレクタに「:empty」を指定することで空の要素を指定することができます。空の要素を削除したい場合には、「remove()」メソッドを使います。サンプルの場合、テーブルの行ごと削除するには、「$("#sale tbody td:empty").parent().remove();」とします。

COLUMN jQueryを使わない場合のコード

サンプルをjQueryを使わずに純粋なJavaScriptのみで書いた場合は、次のようになります。

```
window.addEventListener("load", function(){
  var ele = document.querySelectorAll("#sale tbody td");
  for(var i=0; i<ele.length; i++){
    if(ele[i].innerHTML === ""){
      ele[i].innerHTML = "売り上げデータなし";
    }
  }
});
```

関連項目 ▶▶▶
● 指定した内容が含まれる要素を指定する ……p.163

見出し要素を指定する

ブラウザの対応	IE6以降、Chrome、Safari 5以降、Firefox 4以降、iOS4以降、Android 2以降
jQueryの対応	1.0以降
Zepto.jsの対応	×
App Frameworkの対応	×

ここでは、見出し要素を指定する方法について解説します。

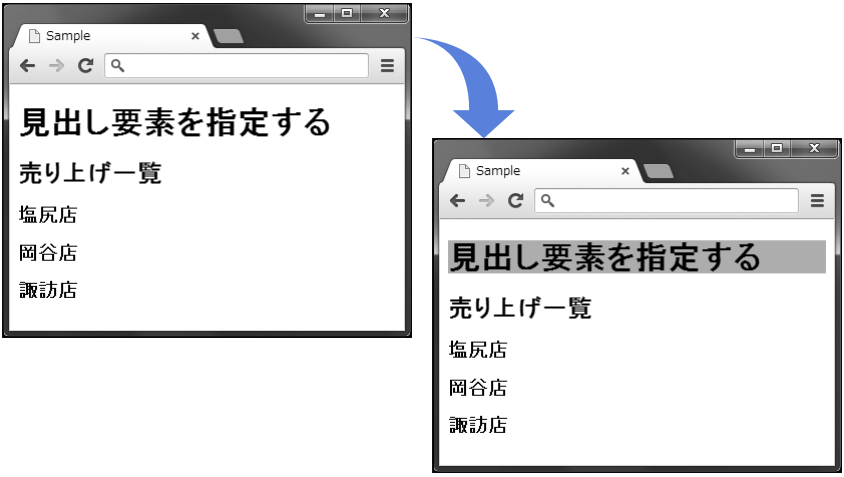
SAMPLE CODE HTMLのコード(index.html)

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>Sample</title>
    <script src="http://ajax.googleapis.com/ajax/libs/jquery/2.1.0/jquery.min.js"></script>
    <script src="js/sample.js"></script>
  </head>
  <body>
    <h1>見出し要素を指定する</h1>
    <h2>売り上げ一覧</h2>
    <h3>塩尻店</h3>
    <h3>岡谷店</h3>
    <h3>諏訪店</h3>
  </body>
</html>
```

SAMPLE CODE JavaScriptのコード(sample.js)

```
$(function(){
  // 見出し要素にマウスが乗った／離れた場合に処理
  $(":header").hover(function(){
    // マウスが乗ったら背景色をオレンジ色にする
    $(this).css("background-color", "orange");
  }, function(){
    // マウスが離れたら背景色を白色にする
    $(this).css("background-color", "white");
  });
});
```

見出し要素の上にマウスを重ねると、背景色がオレンジ色になります。マウスを離すと背景色は白色になります。



ONEPOINT 見出し要素を指定するには「:header」を指定する

jQueryでは、セレクタに「:header」を指定することで見出し要素を指定することができます。見出し要素は「h1」「h2」「h3」「h4」「h5」「h6」になります。「head」要素や「header」要素、「thead」要素、「hgroup」要素（HTML5.1では削除されている）は対象にはなりません。

COLUMN jQueryを使わない場合のコード

サンプルをjQueryを使わずに純粋なJavaScriptのみで書いた場合は、次のようになります。

```
window.addEventListener("load", function(){
  var ele = document.querySelectorAll("h1,h2,h3,h4,h5,h6");
  for(var i=0; i<ele.length; i++){
    ele[i].addEventListener("mouseover", function(){
      this.style.backgroundColor = "orange";
    });
    ele[i].addEventListener("mouseout", function(){
      this.style.backgroundColor = "white";
    });
  }
});
```

関連項目 ▶▶▶

- 複数の要素を指定する …… p.76
- 特定の要素を指定する …… p.80

選択された要素を指定する

ブラウザの対応	IE6以降、Chrome、Safari 5以降、Firefox 4以降、iOS4以降、Android 2以降
jQueryの対応	1.0以降
Zepto.jsの対応	×
App Frameworkの対応	×

ここでは、選択された要素を指定する方法について解説します。

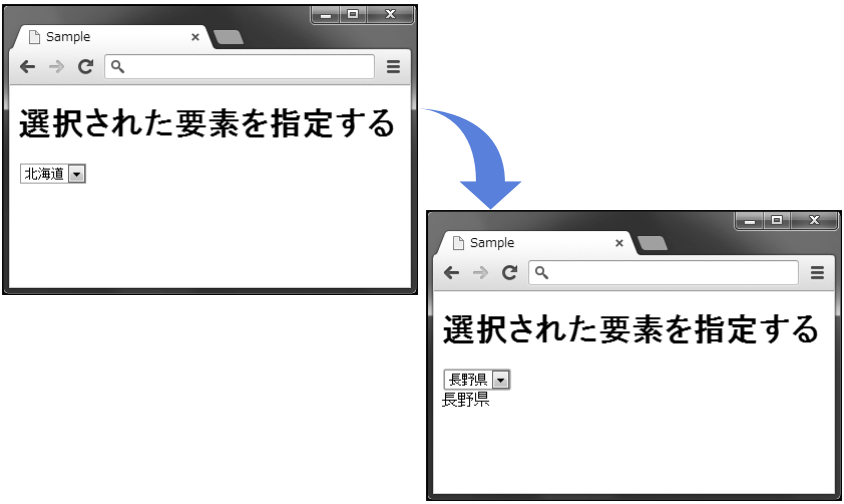
SAMPLE CODE HTMLのコード(index.html)

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>Sample</title>
    <script src="http://ajax.googleapis.com/ajax/libs/jquery/2.1.0/jquery.min.js"></script>
    <script src="js/sample.js"></script>
  </head>
  <body>
    <h1>選択された要素を指定する</h1>
    <form>
      <select id="pref">
        <option>北海道</option>
        <option>東京都</option>
        <option>長野県</option>
        <option>愛知県</option>
        <option>大阪府</option>
        <option>福岡県</option>
        <option>沖縄県</option>
      </select>
    </form>
    <output></output>
  </body>
</html>
```

SAMPLE CODE JavaScriptのコード(sample.js)

```
$(function(){
  // ID名prefのセレクトメニューにchangeイベント設定する
  $("#pref").change(function(){
    // 選択された項目名を表示する
    $("output:first").text($("#pref option:selected").text());
  });
});
```

セレクトメニューで項目を選択すると、選択された都道府県名が表示されます。



ONEPOINT　選択された要素を指定するには「:selected」を指定する

jQueryでは、セレクタに「:selected」を指定することで選択されている要素を指定することができます。セレクトメニューで複数項目を選択可能にしている場合は、「each()」メソッドを使って選択された項目を1つずつ処理することができます。テキストフィールドに入力した文字を選択した場合は、「:selected」で選択文字を取得することはできません。

なお、Zepto.js 1.0でサンプルを動作させるには、selector.jsを入れる必要があります。

COLUMN　jQueryを使わない場合のコード

サンプルをjQueryを使わずに純粋なJavaScriptのみで書いた場合は、次のようになります。

```
window.addEventListener("load", function(){
  document.getElementById("pref").addEventListener("change", function(){
    document.getElementsByTagName("output")[0].innerHTML = this.value;
  });
});
```

最初の要素を指定する

ブラウザの対応	IE6以降、Chrome、Safari 5以降、Firefox 4以降、iOS4以降、Android 2以降
jQueryの対応	1.0以降
Zepto.jsの対応	×
App Frameworkの対応	×

ここでは、最初の要素を指定する方法について解説します。

SAMPLE CODE HTMLのコード(index.html)

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>Sample</title>
    <script src="http://ajax.googleapis.com/ajax/libs/jquery/2.1.0/jquery.min.js"></script>
    <script src="js/sample.js"></script>
  </head>
  <body>
    <h1>最初のinput要素を選択する</h1>
    <form>
      <input type="text" value="iPhone"><br>
      <input type="text" value="iPad"><br>
      <input type="text" value="Nexus"><br>
      <input type="text" value="Nexus 7"><br>
      <input type="text" value="Galaxy Nexus"><br>
      <input type="text" value="Kindle"><br>
      <input type="text" value="Kindle Fire HD"><br>
      <input type="button" value="最初のinput要素の背景色を変更">
    </form>
  </body>
</html>
```

SAMPLE CODE JavaScriptのコード(sample.js)

```
$(function(){
  // ボタンにclickイベントを割り当てる
  $(":button").click(function(){
    // 最初のinput要素を指定する
    $("input:first").each(function(){
      // 背景色をオレンジ色にする
      $(this).css("background", "orange");
    });
  });
});
```

ボタンをクリックすると、最初の「input」要素がオレンジ色の背景色になります。

ONEPOINT　最初の要素を指定するには「:first」と指定する

jQueryでは、セレクタに「:first」と指定することで最初の要素を指定することができます。最後の要素を指定したい場合は、「:last」を指定します。

なお、Zepto.js 1.0でサンプルのように「:first」を使えるようにするには、selector.jsを入れる必要があります。

COLUMN　jQueryを使わない場合のコード

サンプルをjQueryを使わずに純粋なJavaScriptのみで書いた場合は、次のようになります。次のコードでは「getElementsByTagName()」メソッドを使っていますが、「querySelector()」や「querySelectorAll()」メソッドを使って要素を取得することもできます。

```
window.addEventListener("load", function(){
  document.querySelectorAll("input[type=button]")[0]
    .addEventListener("click", function(){
      var ele = document.getElementsByTagName("input");
      ele[0].style.background = "orange";
  });
});
```

関連項目 ▶▶▶

- 最後の要素を指定する …… p.91
- 最初の子要素を指定する …… p.122
- 最後の子要素を指定する …… p.124

最後の要素を指定する

ブラウザの対応	IE6以降、Chrome、Safari 5以降、Firefox 4以降、iOS4以降、Android 2以降
jQueryの対応	1.0以降
Zepto.jsの対応	×
App Frameworkの対応	×

ここでは、最後の要素を指定する方法について解説します。

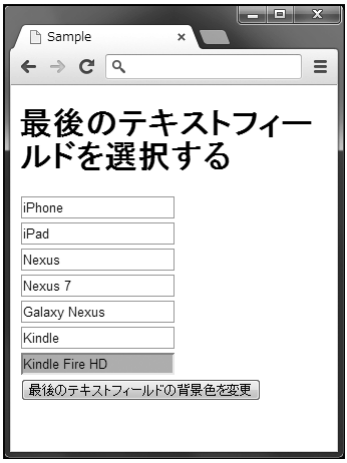
SAMPLE CODE HTMLのコード(index.html)

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>Sample</title>
    <script src="http://ajax.googleapis.com/ajax/libs/jquery/2.1.0/jquery.min.js"></script>
    <script src="js/sample.js"></script>
  </head>
  <body>
    <h1>最後のテキストフィールドを選択する</h1>
    <form>
      <input type="text" value="iPhone"><br>
      <input type="text" value="iPad"><br>
      <input type="text" value="Nexus"><br>
      <input type="text" value="Nexus 7"><br>
      <input type="text" value="Galaxy Nexus"><br>
      <input type="text" value="Kindle"><br>
      <input type="text" value="Kindle Fire HD"><br>
      <input type="button" value="最後のテキストフィールドの背景色を変更">
    </form>
  </body>
</html>
```

SAMPLE CODE JavaScriptのコード(sample.js)

```
$(function(){
  // ボタンにclickイベントを割り当てる
  $(":button").click(function(){
    // 背景色をオレンジ色にする
    $("input[type=text]:last").css("background", "orange");
  });
});
```

ボタンをクリックすると、最後のテキストフィールドがオレンジ色の背景色になります。



ONEPOINT　最後の要素を指定するには「:last」と指定する

jQueryでは、セレクタに「:last」と指定することで最後の要素を指定することができます。最後の要素でなく、最初の要素を指定したい場合は、「:first」を使います。

なお、Zepto.js 1.0でサンプルを動作させるには、selector.jsを入れる必要があります。

COLUMN　jQueryを使わない場合のコード

サンプルをjQueryを使わずに純粋なJavaScriptのみで書いた場合は、次のようになります。

```
window.addEventListener("load", function(){
  document.querySelectorAll("input[type=button]")[0]
    .addEventListener("click", function(){
      var ele = document.querySelectorAll("input[type='text']");
      ele[ele.length-1].style.background = "orange";
  });
});
```

関連項目 ▶▶▶

● 最初の要素を指定する …… p.89

SECTION-020

要素の前後関係を指定する

ブラウザの対応	IE6以降、Chrome、Safari 5以降、Firefox 4以降、iOS4以降、Android 2以降
jQueryの対応	1.0以降
Zepto.jsの対応	1.0以降
App Frameworkの対応	1.2以降

ここでは、要素の前後関係を指定する方法について解説します。

SAMPLE CODE HTMLのコード(index.html)

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>Sample</title>
    <script src="http://ajax.googleapis.com/ajax/libs/jquery/2.1.0/jquery.min.js"></script>
    <script src="js/sample.js"></script>
  </head>
  <body>
    <h1>特定の要素の次にある要素を指定する</h1>
    <ol>
      <li><span>Item 1</span></li>
      <li>Item 2</li>
      <li>Item 3
        <ol>
          <li><span>Item 3-1</span></li>
          <li>Item 3-2</li>
        </ol>
      </li>
      <li>Item 4</li>
    </ol>
    <ol>
      <li>Item B1</li>
      <li><span>Item B2</span></li>
    </ol>
  </body>
</html>
```

SAMPLE CODE JavaScriptのコード(sample.js)

```
$(function(){
  // h1要素の次にあるol要素の枠を赤色にする
  $("h1+ol").css("border", "1px solid red");
});
```

「h1」の次にある「ol」要素だけに赤い枠が表示されます。「h1」の次ではない「ol」要素には枠は表示されません。

ONEPOINT 要素の前後関係を指定するには「+」を使う

jQueryで要素の前後関係を指定するには、セレクタに「+」を使って「要素名+要素名」のように指定します。「+」の前後に空白が入っていても問題ありません。

たとえば、「$("h1+div")」とすると、「h1」要素の次にある「div」要素だけが対象になります。

COLUMN jQueryを使わない場合のコード

サンプルをjQueryを使わずに純粋なJavaScriptのみで書いた場合は、次のようになります。

```
window.addEventListener("load", function(){
  var ele = document.querySelectorAll("h1+ol");
  for(var i=0; i<ele.length; i++){
    ele[i].style.border = "1px solid red";
  }
});
```

関連項目 ▶▶▶

- 親要素を指定する …… p.106
- 子要素だけを指定する …… p.109
- 親子関係にある子要素を指定する …… p.111
- 特定の要素を祖先に持つ要素を指定する …… p.113
- 兄弟要素を指定する …… p.126

SECTION-021

指定した出現順番より小さい要素を指定する

ブラウザの対応	IE6以降、Chrome、Safari 5以降、Firefox 4以降、iOS4以降、Android 2以降
jQueryの対応	1.0以降
Zepto.jsの対応	×
App Frameworkの対応	×

ここでは、指定した出現順番より小さい要素を指定する方法について解説します。

SAMPLE CODE HTMLのコード(index.html)

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>Sample</title>
    <script src="http://ajax.googleapis.com/ajax/libs/jquery/2.1.0/jquery.min.js"></script>
    <script src="js/sample.js"></script>
  </head>
  <body>
    <h1>指定番目より前の要素を指定する</h1>
    <table border="1" id="sale">
      <thead>
        <tr><th>日付</th><th>売り上げ</th></tr>
      </thead>
      <tbody>
      <tr><td>1月</td><td>100円</td></tr>
      <tr><td>2月</td><td>200円</td></tr>
      <tr><td>3月</td><td>140円</td></tr>
      <tr><td>4月</td><td>1100円</td></tr>
      <tr><td>5月</td><td>1300円</td></tr>
      <tr><td>6月</td><td>900円</td></tr>
      <tr><td>7月</td><td>450円</td></tr>
      <tr><td>8月</td><td>1000円</td></tr>
      <tr><td>9月</td><td>1900円</td></tr>
      <tr><td>10月</td><td>2100円</td></tr>
      <tr><td>11月</td><td>3500円</td></tr>
      <tr><td>12月</td><td>5700円</td></tr>
      </tbody>
    </table>
  </body>
</html>
```

SAMPLE CODE JavaScriptのコード(sample.js)

```
$(function(){
  // 6番目よりも前のtr要素の背景色をオレンジ色にする
  $("#sale tbody tr:lt(6)").css("background-color", "orange");
});
```

ページが表示されると、テーブル内のセルの先頭から6番目までがオレンジ色の背景色になります。

ONEPOINT 指定した出現順番より小さい要素を指定するには「:lt(出現位置)」を指定する

jQueryでは、セレクタに「:lt(出現位置)」を指定することで指定した出現位置よりも前の要素を指定することができます。出現位置は「0」が最初の要素になる点に注意が必要です。

COLUMN jQueryを使わない場合のコード

サンプルをjQueryを使わずに純粋なJavaScriptのみで書いた場合は、次のようになります。

```
window.addEventListener("load", function(){
  var ele = document.querySelectorAll("#sale tbody tr");
  for(var i=0; i<ele.length; i++){
    if (i < 6){
      ele[i].style.background = "orange";
    }
  }
});
```

関連項目 ▶▶▶

- 指定した出現順番より大きい要素を指定する …… p.98
- 指定した出現位置の要素を指定する …… p.101

指定した出現順番より大きい要素を指定する

ブラウザの対応	IE6以降、Chrome、Safari 5以降、Firefox 4以降、iOS4以降、Android 2以降
jQueryの対応	1.0以降
Zepto.jsの対応	×
App Frameworkの対応	×

ここでは、指定した出現順番より大きい要素を指定する方法について解説します。

SAMPLE CODE HTMLのコード(index.html)

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>Sample</title>
    <script src="http://ajax.googleapis.com/ajax/libs/jquery/2.1.0/jquery.min.js"></script>
    <script src="js/sample.js"></script>
  </head>
  <body>
    <h1>指定番目以降の要素を指定する</h1>
    <table border="1" id="sale">
      <thead>
        <tr><th>日付</th><th>売り上げ</th></tr>
      </thead>
      <tbody>
      <tr><td>1月</td><td>100円</td></tr>
      <tr><td>2月</td><td>200円</td></tr>
      <tr><td>3月</td><td>140円</td></tr>
      <tr><td>4月</td><td>1100円</td></tr>
      <tr><td>5月</td><td>1300円</td></tr>
      <tr><td>6月</td><td>900円</td></tr>
      <tr><td>7月</td><td>450円</td></tr>
      <tr><td>8月</td><td>1000円</td></tr>
      <tr><td>9月</td><td>1900円</td></tr>
      <tr><td>10月</td><td>2100円</td></tr>
      <tr><td>11月</td><td>3500円</td></tr>
      <tr><td>12月</td><td>5700円</td></tr>
      </tbody>
    </table>
  </body>
</html>
```

SAMPLE CODE JavaScriptのコード(sample.js)

```
$(function(){
  // 6番目よりも後のtr要素の背景色をオレンジ色にする
  $("#sale tbody tr:gt(5)").css("background", "orange");
});
```

ページが表示されると、ヘッダーを除いたテーブル内のセルの7番目以降がオレンジ色の背景色になります。

ONEPOINT 指定した出現順番より大きい要素を指定するには「:gt(出現位置)」を指定する

jQueryでは、セレクタに「:gt(出現位置)」を指定することで指定した出現位置よりも後の要素を指定することができます。出現位置は「0」が最初の要素になる点に注意が必要です。

COLUMN jQueryを使わない場合のコード

サンプルをjQueryを使わずに純粋なJavaScriptのみで書いた場合は、次のようになります。

```
window.addEventListener("load", function(){
  var ele = document.querySelectorAll("#sale tbody tr");
  for(var i=0; i<ele.length; i++){
    if (i > 5){
      ele[i].style.background = "orange";
    }
  }
});
```

関連項目 ▶▶▶

- 指定した出現順番より小さい要素を指定する …… p.95
- 指定した出現位置の要素を指定する …… p.101

SECTION-023

指定した出現位置の要素を指定する

ブラウザの対応	IE6以降、Chrome、Safari 5以降、Firefox 4以降、iOS4以降、Android 2以降
jQueryの対応	1.0以降
Zepto.jsの対応	×
App Frameworkの対応	×

ここでは、指定した出現位置の要素を指定する方法について解説します。

SAMPLE CODE HTMLのコード(index.html)

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>Sample</title>
    <script src="http://ajax.googleapis.com/ajax/libs/jquery/2.1.0/jquery.min.js"></script>
    <script src="js/sample.js"></script>
  </head>
  <body>
    <h1>指定番目の要素を指定する</h1>
    <table border="1" id="sale">
      <thead>
        <tr><th>日付</th><th>売り上げ</th></tr>
      </thead>
      <tbody>
      <tr><td>1月</td><td>100円</td></tr>
      <tr><td>2月</td><td>200円</td></tr>
      <tr><td>3月</td><td>140円</td></tr>
      <tr><td>4月</td><td>1100円</td></tr>
      <tr><td>5月</td><td>1300円</td></tr>
      <tr><td>6月</td><td>900円</td></tr>
      <tr><td>7月</td><td>450円</td></tr>
      <tr><td>8月</td><td>1000円</td></tr>
      <tr><td>9月</td><td>1900円</td></tr>
      <tr><td>10月</td><td>2100円</td></tr>
      <tr><td>11月</td><td>3500円</td></tr>
      <tr><td>12月</td><td>5700円</td></tr>
      </tbody>
    </table>
  </body>
</html>
```

SAMPLE CODE JavaScriptのコード(sample.js)

```
$(function(){
  // 7番目のtr要素の背景色をオレンジ色にする
  $("#sale tbody tr:eq(6)").css("background-color", "orange");
});
```

ページが表示されると、テーブル内の7月のセルがオレンジ色の背景色になります。

ONEPOINT 指定した出現位置の要素を指定するには「:eq(出現位置)」を指定する

jQueryでは、セレクタに「:eq(出現位置)」を指定することで指定した出現の要素を指定することができます。出現位置は「0」が最初の要素になります。また、1.8.1以降では「eq(-2)」のように負数を指定することもできます。この場合、末尾からの順番になります。

なお、Zepto.js 1.0でサンプルを動作させるには、selector.jsを入れる必要があります。

COLUMN jQueryを使わない場合のコード

サンプルをjQueryを使わずに純粋なJavaScriptのみで書いた場合は、次のようになります。

```
window.addEventListener("load", function(){
  var ele = document.querySelectorAll("#sale tbody tr");
  ele[6].style.background = "orange";
});
```

関連項目 ▶▶▶

- 指定した出現順番より小さい要素を指定する …… p.95
- 指定した出現順番より大きい要素を指定する …… p.98

偶数番目／奇数番目の要素を指定する

ブラウザの対応	IE6以降、Chrome、Safari 5以降、Firefox 4以降、iOS4以降、Android 2以降
jQueryの対応	1.0以降
Zepto.jsの対応	×
App Frameworkの対応	×

ここでは、偶数番目／奇数番目の要素を指定する方法について解説します。

SAMPLE CODE HTMLのコード(index.html)

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>Sample</title>
    <script src="http://ajax.googleapis.com/ajax/libs/jquery/2.1.0/jquery.min.js"></script>
    <script src="js/sample.js"></script>
  </head>
  <body>
    <h1>奇数番目と偶数番目の要素を指定する</h1>
    <table id="sale">
      <thead>
        <tr><th>日付</th><th>売り上げ</th></tr>
      </thead>
      <tbody>
      <tr><td>1月</td><td>100円</td></tr>
      <tr><td>2月</td><td>200円</td></tr>
      <tr><td>3月</td><td>140円</td></tr>
      <tr><td>4月</td><td>1100円</td></tr>
      <tr><td>5月</td><td>1300円</td></tr>
      <tr><td>6月</td><td>900円</td></tr>
      <tr><td>7月</td><td>450円</td></tr>
      <tr><td>8月</td><td>1000円</td></tr>
      <tr><td>9月</td><td>1900円</td></tr>
      <tr><td>10月</td><td>2100円</td></tr>
      <tr><td>11月</td><td>3500円</td></tr>
      <tr><td>12月</td><td>5700円</td></tr>
      </tbody>
    </table>
  </body>
</html>
```

SAMPLE CODE JavaScriptのコード(sample.js)

```
$(function(){
  // 偶数番目の背景色をオレンジ色にする
  $("#sale tbody tr:even").css("background-color", "orange");
  // 奇数番目の背景色を黄色にする
  $("#sale tbody tr:odd").css("background-color", "yellow");
});
```

ページが表示されると、テーブル内のセルの奇数番目の背景色が黄色に、偶数番目の要素がオレンジ色の背景色になります。

ONEPOINT **偶数番目の要素を指定するには「:even」、奇数番目の要素を指定するには「:odd」を指定する**

jQueryでは、セレクタに「:even」を指定することで偶数番目の要素を指定することができます。偶数番目ではなく奇数番目の要素を指定したい場合には、「:odd」を指定します。サンプルのようにすると、簡単に表のセルに交互に色を指定することができます。

COLUMN **jQueryを使わない場合のコード**

サンプルをjQueryを使わずに純粋なJavaScriptのみで書いた場合は、次のようになります。「i & 1」が偶数か奇数かを調べている部分になります。「&」は論理積で次の場合、数値の最下位ビットが「1」なら「1」（奇数）、「0」なら「0」（偶数）になります。

```
window.addEventListener("load", function(){
  var ele = document.querySelectorAll("#sale tbody tr");
  for(var i=0; i<ele.length; i++){
    if (i & 1){
      ele[i].style.background = "yellow";
    }else{
      ele[i].style.background = "orange";
    }
  }
});
```

関連項目 ▶▶▶

- 指定した出現順番より小さい要素を指定する …… p.95
- 指定した出現順番より大きい要素を指定する …… p.98
- 指定した出現位置の要素を指定する …… p.101
- 指定した順番の子要素を指定する …… p.119