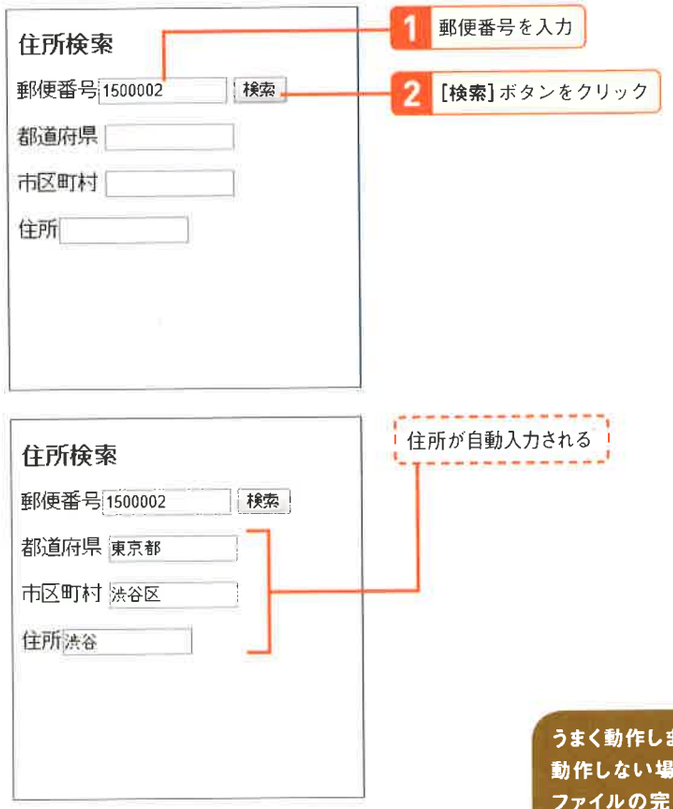
3 プログラムが完成した

プログラムが完成したら、内容を上書き保存して、 ⋮ う❶❷。
index.htmlをブラウザで開いて動作を確認しましょ



Chapter

12

YouTubeの動画ギャラリーを作ろう

この章では、学習の総まとめとして、Googleが提供するYouTube Data API（v3）を使った、ビデオギャラリーを作成します。いままでに学んだことを生かしてWebサイトを完成させましょう。

Lesson 64 [ゴールの確認]

ゴールを確認しましょう

このレッスンのポイント

この章では、学習の総まとめとして、Googleが提供するYouTube Data API(v3)を使って、ビデオギャラリーのWebサイトを作成します。まずは制作に必要となるYouTube Data APIの概要と、最終的なゴールイメージを確認しましょう。

ビデオギャラリーを作成しよう

数あるWeb APIの中で、特に実用性が高く学ぶ価値が高いものとして、Googleによって提供されているWeb API群があります。 今回はその中の1つ「YouTube Data API」を使用して、YouTube動画を一覧表示したビデオギャラリーを作成します。Googleが提供するAPIの使用方法を学びながら、これまでの集大成として、1つの作品を作り上げましょう。

▶完成イメージ

特定のテーマで動画を収集し、一覧表示する

動画を検索するためのAPIを利用して、取得したデータからギャラリーを作成します。

YouTube Data API(v3)について

インターネットで「動画」といえばYouTubeが有名ですよね。YouTubeの運営会社であるGoogleは、YouTubeの動画に関するさまざまなデータを利用するためのWeb API「YouTube Data API」を無料で公開しています。今回はこのWeb APIの最新バージョンである「v3」を利用して、ギャラリーで表示する動画を取得しましょう。このWeb APIを利用すれば、YouTubeのマイリストや動画の検索結果から、動画データを取得することができます。

▶YouTube

https://www.youtube.com/

制作の手順

まずYouTube Data API (v3) の利用準備として、APIの利用に必要な「APIキー」の取得を行います。その後、利用方法を確認した後、ビデオギャラリーの制作に取り掛かります。 制作ではまず、JavaScriptの記述を行って機能を完成させてから、CSSでスタイリングの記述を行い、ビデオギャラリーを完成させます。

❶YouTube Data API (v3) の利用準備
❷YouTube Data API (v3) の利用方法の確認
❸ビデオギャラリーの制作：JavaScriptの記述
❹ビデオギャラリーの制作：スタイルの記述
❺完成

この章だけは「APIキー」がなければサンプルファイルが動きません。Lesson 65の手順にしたがって「APIキー」を取得する必要があります。

Lesson 65 [APIキーの発行]

YouTube Data API（v3）を利用する準備をしましょう

このレッスンのポイント

このレッスンではYouTubeから動画データを取得するために必要なWeb API「YouTube Data API(v3)」の利用準備をしましょう。YouTube Data API(v3) を利用するには、APIキーの発行が必要となります。

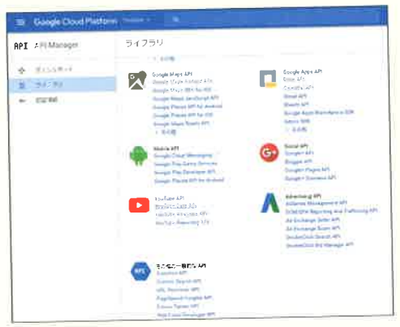
事前準備で行うこと

YouTube Data API (v3) を利用するための事前準備として、API利用に必要なAPIキーの発行を行います。APIキーは、APIが悪用されたり、規約に違反して利用されることがないように、誰が利用しているのか特定するための利用証明書の役割を担っています。

APIキーを利用するには、Googleの提供するWebサービス「Google Cloud Platform」にログインして、キーの発行を行う必要があります。このレッスンでは「Google Cloud Platform」にログインして、実際にAPIキーを発行していきましょう。

❶Google Cloud Platform にログイン
❷プロジェクトを作成する
❸APIキーを発行して、APIの利用準備を行う

▶ Google Cloud Platform



Google Cloud Platformでは、GoogleのあらゆるWeb APIを利用できます。

Google Cloud Platformを利用する

まずは「Google Cloud Platform」にログインしましょう。「https://console.cloud.google.com」にアクセスして、Googleアカウントでログインします。すでにGoogleアカウントを持っている人は、それを利用してログインできます。まだ持っていない場合は画面下部の［アカウントを作成］からアカウントを作成します。Googleアカウントは、基本的に無料で取得、利用できます。

1 ログインする

1 Cloud Platformのページ（https://console.cloud.google.com）を表示
2 Googleアカウントでログイン

2 利用規約に同意する

1 お知らせメールの受信の［はい／いいえ］を選択
2 利用規約の同意で［はい］を選択
3 ［同意して続行］をクリック

本書で利用するサービスは無料（2017年2月現在）ですが、サービスの中には有料のものもあります。利用する際は規約を確認しましょう。

プロジェクトを作成する

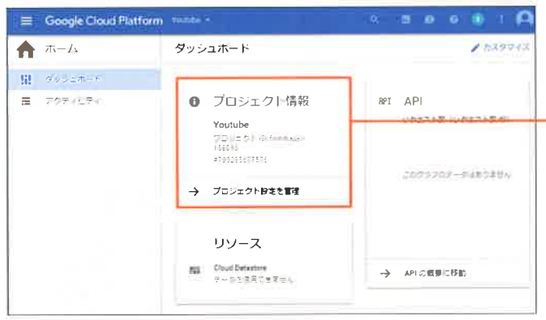
続いて、APIキーを発行するために「プロジェクト」を作成します。プロジェクトとは、Google Cloud Platformのサービス利用状況を管理するための単位です。青いメニューバーの「Project▼」と書かれた部分を選択して、表示されたメニューから「プロジェクトを作成」をクリックします。新しいプロジェクトの入力画面が表示されたら、任意のプロジェクト名を指定します。

1 プロジェクトを作成する

1. [Project▼] をクリック
2. 表れたメニューから [プロジェクトを作成] をクリック

2 プロジェクト名を入力する

1. 任意のプロジェクト名を入力
2. [作成] をクリック



3 プロジェクトが作成された

作成したプロジェクトが表示された

YouTube Data APIを有効にする

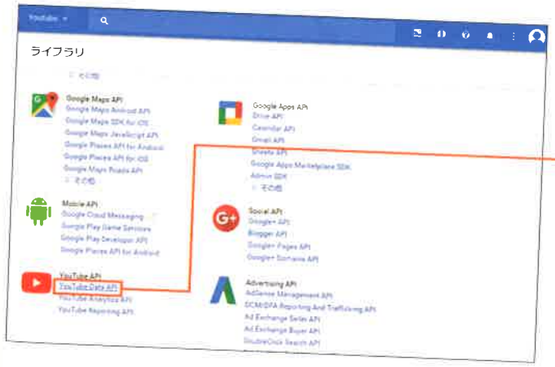
プロジェクトの作成が完了したら、いよいよ「YouTube Data API (v3)」を有効にします。「有効なAPI」のページに遷移して [APIを有効にする] をクリックします。利用するAPIを選択する画面が表示されるので「YouTube Data API」のリンクをクリックして、画面遷移後に [有効にする] をクリックします。

1 「有効なAPI」のページへ移動する

1. [APIの概要に移動] をクリック

2 APIの有効画面に切り替える

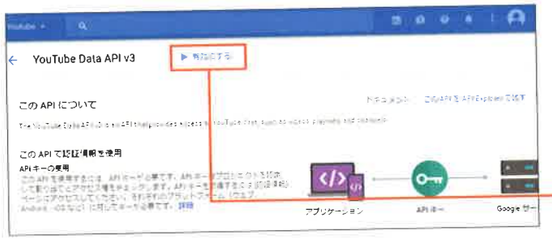
1. [APIを有効にする] をクリック



3 APIを選択する

1. [YouTube Data API] をクリック

NEXT PAGE



4 APIを有効にする

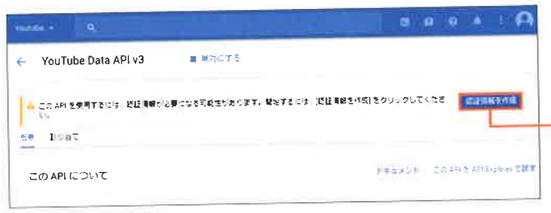
「YouTube Data API v3」のページが表示された

1 [有効にする]をクリック

APIキーを発行する

最後にAPIキーを発行します。JavaScript側でAPIを利用する際にこのAPIキーを指定しておくと、Googleによって認証されてサービスが利用可能になります。発行の際に「APIを呼び出す場所」や「アクセスするデータの種類」といった情報を設定します。

本書では扱いませんが[アクセスするデータの種類]で「ユーザーデータ」を選択すると、ユーザーにひもづく利用状況などのデータにアクセスできるようになります。



1 APIキーの作成を開始する

1 [認証情報を作成]をクリック

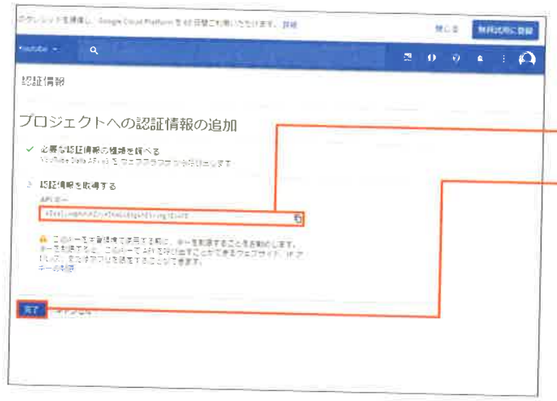
2 認証情報を追加する

1 [YouTube Data API v3]を選択

2 [ウェブブラウザ（Javascript）]を選択

3 [一般公開データ]を選択

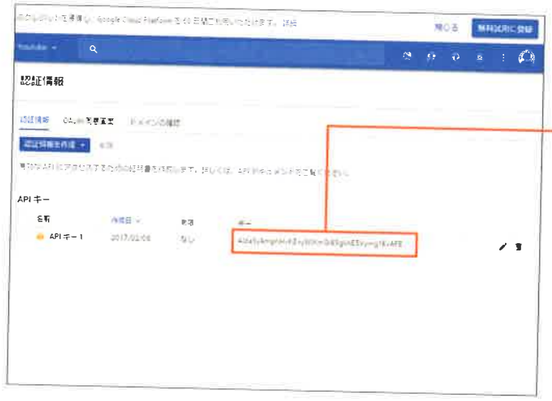
4 [必要な認証情報]をクリック



3 APIキーが発行された

認証情報（APIキー）が発行された

1 [完了]をクリック



4 認証情報の管理画面に戻る

1 APIキーが表示された

このAPIキーをコピーする

お疲れさまでした。ここで取得したAPIキーは今後の制作で使用します。APIキーはこのページでいつでも確認できますが、念のためコピーしてわかりやすい場所に保存しておくといいでしょう。

Lesson 66 [APIのパラメータ]

YouTube Data API (v3) の使い方を確認しましょう

このレッスンのポイント

APIキーを取得したら、さっそくYouTube Data API(v3) の使い方を確認していきましょう。このレッスンでは「検索」の方法を中心に確認していきます。キーワードを使ってギャラリーに表示する動画を検索、取得できるようにしましょう。

APIの利用方法

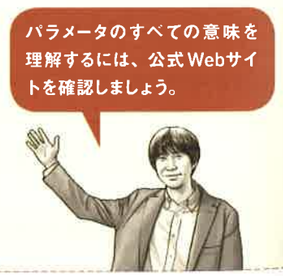
YouTube Data API (v3) を利用するには、「https://www.googleapis.com/youtube/v3/search」のURLの後に?(クエスチョンマーク) を付け、その後に「パラメータ名=値」の形式で検索条件などのパラメータを指定していきます。パラメータは&記号を使って複数指定でき、先ほど取得したAPIキーも「key=APIキー」の形式で指定します。

▶「music」をキーワードとした検索を行う

```
https://www.googleapis.com/youtube/v3/search?type=video&part=snippet&q=music&key=APIキー
```

▶ 公式Webサイト

https://developers.google.com/youtube/v3/docs/search/list?hl=ja



動画検索に利用するパラメータとレスポンス

今回の動画検索で指定するパラメータは下表の通りです。ビデオギャラリーを作成するため、musicという検索キーワードとAPIキーの他に、Webページに埋め込み可能な動画のみ取得する「videoEmbeddable」などのパラメータを追加しています。また、レスポンスからはさまざまなデータを取得できますが、今回はYouTubeのビデオにアクセスするために必要となる「videoId」を取得します。

▶ サンプルで使用するパラメータ

パラメータ	意味	備考
part	取得するリソースのプロパティを指定	必須項目。idかsnippet を指定。snippetにするとすべてのプロパティを取得できる
type	検索対象を特定のリソースに限定	video、channel、playlistのいずれかを指定
q	検索に用いる文字列を指定	「音楽」「動物」「ニュース」などの検索文字列を指定
videoEmbeddable	検索対象をWebページに埋め込み可能な動画のみに限定	trueまたはfalse
videoSyndicated	検索対象をyoutube.com 以外で再生できる動画のみに限定	trueまたはfalse
maxResults	一度に取得する検索数を指定	10、100...などの数字で指定
key	APIの利用に必要なAPIキーを指定	自分で取得したAPIキーが必要

▶ レスポンスの例

```
{
  "items": [  //検索結果を格納した配列
  {
    /* 中略 */
    "id": {        //検索結果のID
      "kind":      //リソースの種類,
      "videoId":   //動画ID...  /* 今回の動画表示に必要 */
      :
}
```

Lesson 67 [YouTube Data API(v3)の利用]

ビデオギャラリーを作成しましょう

このレッスンのポイント

ここから実際にビデオギャラリーを作成していきます。今回の実践にはYouTube Data API(v3) を利用するためのAPIキーが必要になります。まだAPIキーを取得していない人は、この章のLesson 65を参考にAPIキーを事前に取得しておいてください。

プログラムの流れを確認しよう

さあ、いよいよプログラムの作成に入っていきましょう。このレッスンでは、まずHTMLファイルを編集してビデオギャラリーの表示領域を作ります。次に、JavaScriptファイルを編集してリクエストパラメータの準備をし、YouTube Data API (v3) にリクエスト (Ajax通信) を行い、動画の検索データを取得します。そしてレスポンスの成否に応じた条件分岐を行い取得したデータをもとにビデオを表示して、ビデオギャラリーを実現します。またLesson 68でCSSファイルを編集し、スタイルを整えてビデオギャラリーを完成させます。

❶APIへのリクエスト準備

↓

❷APIへのリクエスト (Ajax通信)

↓

❸レスポンスに応じた条件分岐

↓

❹取得したデータを元にビデオを表示

HTML部分を記述する

1 HTMLファイルに記述する 12/youtube/practice/index.html

今回はWeb APIで得られたデータを元に、JavaScriptでHTMLを記述するので、HTMLファイルにはほとんどコードを記述しません。このレッスンのindex.htmlファイルをBracketsで開いて、以下のコードを記述してください。

HTMLファイルには、後からJavaScriptでギャラリーを表示する場所を指定するため、div要素を追加してid属性に「videoList」を指定しておきましょう❶。この時点では、ブラウザでHTMLファイルを開いても何も表示されていなくてOKです。

```
008 <body>
009 __<h3>Video_Gallery</h3>
010 __<div_id="videoList">
011 __</div>
012 __<script_src="js/jquery-3.1.1.js"></script>
013 __<script_src="js/app.js"></script>
014 </body>
```

❶ 要素を追加する

JavaScript部分を記述する

1 リクエストURLを準備する 12/youtube/practice/js/app.js

ここからはJavaScriptでプログラムを記述していきます。まずは、リクエスト先のURLに含める情報を記述していきましょう。このレッスンのapp.jsファイルをBracketsで開いて、以下のコードを記述してください。まずは、APIを使用するためのAPIキーの指定が必要です。APIキーはそのままでは長いので、変数「KEY」を用意して、値を代入しておきます。続いて、今回利用する検索APIのURLが必要です。こちらも長いので、変数「url」を要して、値を代入しておきましょう❶。

```
001 //_リクエストパラメータのセット
002 var_KEY_=_'あなたが取得したAPIキー';_____________________ //_API_KEY
003 var_url_=_'https://www.googleapis.com/youtube/v3/search?';_//_API_URL
```

APIキーを貼り付ける

❶ 変数に代入

APIキーは取得した人それぞれで異なるので、自分で取得したものを記入してください。間違えないようコピー&ペーストで貼り付けるといいでしょう。

NEXT PAGE

2 リクエストパラメータを指定する

続いて、検索APIに指定するパラメータを指定していきましょう。先に用意した変数「url」に文字列を追記して、パラメータを連結していきます❶。パラメータの意味はコードのコメントに記載した通りです。このサンプルでは検索ワードを「'q=music'」と指定していますが「music」の部分を好きな言葉に変えれば、検索結果を変えることができます。ぜひお好みに応じて変更してみてください。

```
001 //_リクエストパラメータのセット
002 var_KEY_=_'あなたが取得したAPIキー';                    //_API_KEY
003 var_url_=_'https://www.googleapis.com/youtube/v3/search?'; //_API_URL
004 url_+=_'type=video';                  //_動画を検索する
005 url_+=_'&part=snippet';               //_検索結果にすべてのプロパティを含む
006 url_+=_'&q=music';                    //_検索ワード_このサンプルでは_music_に指定
007 url_+=_'&videoEmbeddable=true';       //_Webページに埋め込み可能な動画のみを検索
008 url_+=_'&videoSyndicated=true';       //_youtube.com_以外で再生できる動画のみに限定
009 url_+=_'&maxResults=6';               //_動画の最大取得件数
010 url_+=_'&key='_+_KEY;                 //_API_KEY
```

1 パラメータを連結

Point +=で結合と代入を同時に行う

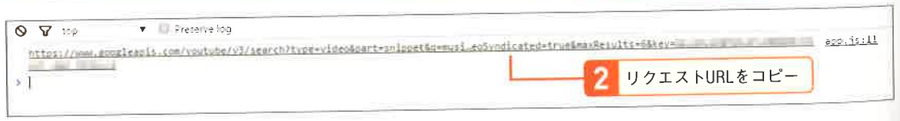
+=は変数に値を結合（加算）するとともに代入する演算子で、「変数=変数+値」と書くのに相当します。他に-=、*=、/=などの演算子もあります。

3 リクエストURLが正常に動作するか確認する

ここまでのプログラムでリクエストURLが完成し、変数urlに代入された状態になっているはずです。ここで動作確認しておきましょう。変数urlをコンソールに表示するコードを追記して、index.htmlをブラウザで開いてください❶。コンソールにリクエストURLが表示されるはずです❷。

```
011 //_動作確認が終わったら消すこと
012 console.log(url);
```

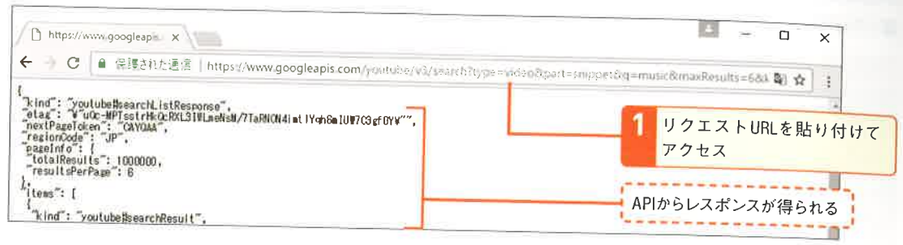
1 コンソールに表示



2 リクエストURLをコピー

4 確認したURLをブラウザでリクエストしてみる

コンソールに表示されたリクエストURLをコピーして、ブラウザのアドレスバーに貼り付けて読み込んでみましょう❶。うまく通信できればブラウザの画面にレスポンスのJSONが表示されます。



Ajaxを利用する

1 Ajaxでリクエストする

リクエストURLとパラメータの準備ができたら実際にAPIを利用する「リクエスト」の処理を記述していきます。以下のコードを追記して、上書き保存してください。

JavaScriptでリクエストを行う方法は、11章で学びましたね。今回もjQueryを用いたAjaxでリクエストを行います。まずは全体を「$(function() { ... })」で囲んでHTMLが読み込まれてから処理が動くようにして「{}」の中に具体的な処理を記述していきます❶。ajaxメソッドの引数に先ほど準備したurlを指定して、リクエストを実行します❷。

```
011
012 //_HTMLが読み込まれてから実行する処理
013 $(function()_{
014 __//_youtubeの動画を検索して取得
015 __$.ajax({
016 ____url:_url,
017 ____dataType_:_'jsonp'
018 __})
019 });
```

1 読み込み完了後に実行

2 リクエストを実行

2 doneメソッドとfailメソッドを追加する

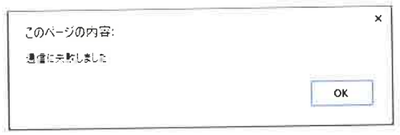
doneメソッド部分には、データ取得が成功したときの処理を記述します❶。ただし、その中の処理を書くのは後まわしにして先にfailメソッドを記述しましょう。failメソッドの部分には、通信に失敗したときの処理として、メッセージを表示するようにしましょう❷。

```
012 //_HTMLが読み込まれてから実行する処理
013 $(function()_{
014 __//_youtubeの動画を検索して取得
015 __$.ajax({
016 ____url:_url,
017 ____dataType_:_'jsonp'
018 __}).done(function(data)_{
019 ____//_データ取得が成功したときの処理
020 __}).fail(function(data)_{
021 ____alert('通信に失敗しました');
022 __});
023 });
```

1 doneメソッドを追加

2 failメソッドを追加

▶ 通信に失敗した時の表示



3 レスポンスごとの処理を記述する

通信が成功しても別の問題でデータが取得できないこともあります。その場合はdata.itemsという値がなくなるので、その存在をif文でチェックし❶、見つからない場合は警告のメッセージを表示します❷。また、取得できなかった際のレスポンスがどのようになっていたのか後で確認できるように、dateの値をコンソールに表示しておきましょう。

```
012 //_HTMLが読み込まれてから実行する処理
013 $(function()_{
014 __//_youtubeの動画を検索して取得
015 __$.ajax({
016 ____url:_url,
017 ____dataType_:_'jsonp'
018 __}).done(function(data)_{
019 ____if_(data.items)_{
020 ______//_データ取得が成功したときの処理
021 ____}_else_{
022 ______console.log(data);
023 ______alert('該当するデータが見つかりませんでした');
024 ____}
025 __}).fail(function(data)_{
026 ____alert('通信に失敗しました');
027 __});
028 });
```

1 データをチェック

2 警告メッセージを表示

NEXT PAGE

4 取得したデータをHTMLに反映する関数を作る

取得したJSONをHTMLに反映する処理は長くなるので、わかりやすくするためにsetData関数として記述しましょう。引数として取得したJSONを丸ごと受け取ることにします❶。

JSONのitemsプロパティには、動画の情報が配列としてまとめられています。これをfor文で1データずつ取り出します❷。そしてそこからビデオIDを取り出し、動画を表示するiframe要素のタグを作成します❸。

最後に作成したHTMLを、videoListというIDを持つHTML要素の子にします❹。

```
029
030 //_データ取得が成功したときの処理
031 function_setData(data)_{                          ← 1 関数を定義
032 __var_result_=_'';
033 __var_video__=_'';
034 __//_動画を表示するHTMLを作る                      ← 2 繰り返し処理
035 __for_(var_i_=_0;_i_<_data.items.length;_i++)_{
036 ____video__=_'<iframe_src="https://www.youtube.com/embed/';
037 ____video__+=__data.items[i].id.videoId;
038 ____video__+=_'"_allowfullscreen></iframe>';
039 ____result_+=_'<div_class="video">'_+_video_+_'</div>'
040 __}                                               ← 3 iframe要素のタグを作る
041 __//_HTMLに反映する
042 __$('#videoList').html(result);                    ← 4 要素を作成
043 }
```

Point iframe要素を使って動画を表示する

YouTubeの動画をWebページに埋め込んで再生する場合、iframe要素を利用して「https://www.youtube.com/embed/ビデオID」というURLを取り込みます。今回のサンプルではiframe要素をdiv要素の子として追加しています。

```
<div class="video">
<iframe  src="https://www.youtube.com/embed/ビデオID"
allowfullscreen></iframe>
</div>
```

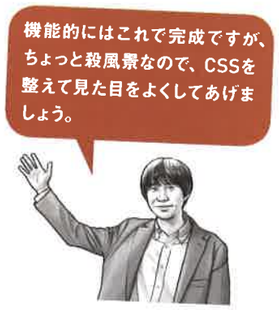

5 関数を呼び出す

関数ができあがったら、doneメソッド内のデータ取得処理が完了した部分で、setData関数を呼び出すようにします❶。

```
018 __}).done(function(data)_{
019 ____if_(data.items)_{
020 ______setData(data);_//_データ取得が成功したときの処理   ← 3 関数を呼び出す
021 ____}_else_{
022 ______console.log(data);
023 ______alert('該当するデータが見つかりませんでした');
024 ____}
025 __}).fail(function(data)_{
026 ____alert('通信に失敗しました');
027 __});
028 });
```

6 プログラムが完成した

プログラムが完成したので、ブラウザでindex.htmlを開いて動作テストしてみましょう。問題がなければYouTubeの動画が表示されます。



Lesson 68 [CSSの設定]

スタイルを整えて Webサイトを完成させましょう

このレッスンのポイント

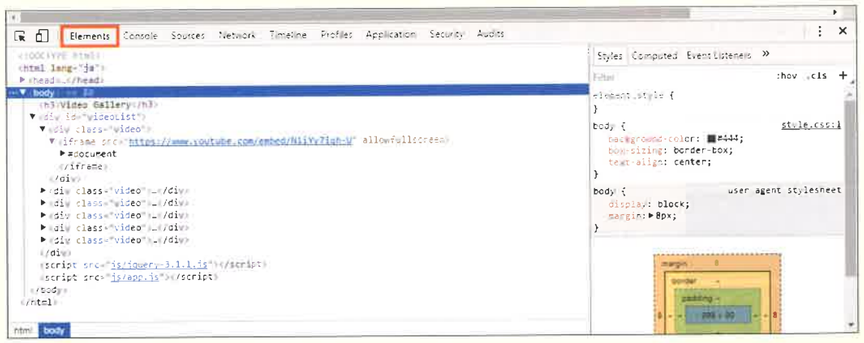
最後にCSSでスタイルを指定して、ビデオギャラリーとしての見た目を整えていきましょう。記述するCSSはChapter 9でフォトギャラリーに適用したものと基本的には同じです。完成までもうひと頑張りです！

ビデオギャラリーのHTML構造を確認する

JavaScriptで作り変えたHTMLに対してCSSを適用するには、HTMLファイルの内容ではなく、JavaScriptが実行された後に、実際にブラウザに表示されているHTMLの構造を想定する必要があります。

Chromeで表示しているWebページのHTMLを確認するには、デベロッパーツールを表示して［Elements］パネルを選択します。

▶ デベロッパーツールの［Elements］パネル



全体のスタイルを整える

1 全体のスタイルとタイトルを整える

このレッスンのstyle.cssファイルを開いて、以下のコードを記述してください。背景色をグレー（#444）にして、文字を中央ぞろえにしています❶。見出しは目立つように、明るめの色で、文字を大きく太くしておきましょう❷。

```
001 body {
002   background-color: #444;
003   box-sizing: border-box;
004   text-align: center;
005 }
006
007 h3 {
008   color: #bbb;
009   font-size: 20px;
010   font-weight: bold;
011 }
```

1 背景色を設定して中央そろえにする

2 タイトルを整える

2 ビデオリストの幅と余白を調整する

ビデオギャラリーのビデオが横並びになるようにレイアウトを整えていきます。YouTubeの動画は「幅560px、高さ315px」で表示しやすいように作られているので、今回はそれにあわせて560px×2程度の幅を指定しています❶。

```
013 #videoList {
014   margin: auto;
015   padding-top: 40px;
016   width: 1216px;
017 }
```

1 幅と余白を調整

NEXT PAGE

3 動画に枠を付ける

動画を際だたせるため、Chapter 9で「フォトギャラリー」を作成した際と同様に動画に枠線を付け、シャドーの効果を適用します❶。

```
#videoList .video {
border: 4px solid #fff;
box-shadow: 0px 0px 14px #000;
float: left;
height: 315px;
margin: 20px;
width: 560px;
}
```

1 枠を付ける

4 iframe要素を整える

最後に、iflame側の幅、高さも動画の枠と同じサイズに指定して、隙間が生まれないようにします❶。以上でビデオギャラリーの制作は完了です。ここまでのコードを記述できたら、ファイルを上書き保存してindex.htmlをブラウザで再読み込みしてください。

```
#videoList .video_iframe {
  border: none;
  height: 315px;
  width: 560px;
}
```

1 iframe要素のサイズを調整する

Chapter 13 独学する技術を身につけよう