



JS

# KINDAI CAMP 2021 Autumn

APIを使って無限の可能性を広げよう

# このアプリの目的



今回のアプリ制作を通じて  
「API」というものを皆さんに知って  
いただきたいと思います！



# 今回のアプリ制作の目的

YouTube JP

検索

ホーム  
急上昇  
登録チャンネル  
ライブラリ  
履歴

Katty60

いいね!

チャンネル登録 8975

メジャーリーグベースボール - トピック あなたにおすすめの動画

「API」というものを、今回のアプリ制作を通じて知っていただき、プログラミングの可能性をより感じてほしいなと思います！

めざましテレビ イチロー選手引退 信玄米食 3.2万回視聴・1週間前

Ichiro vs Cooperstown Tecknowledgist 397万回視聴・1週間前

イチロー 軌跡 稲葉篤紀 20190329 8:17 5万回視聴・1週間前

A fan is very excited to touch Ichiro 292万回視聴・3年前

イチロー選手引退 海外でも大きな反響 NEW BREAKING 7.9万回視聴・1週間前

Ichiro Suzuki - Japan Highlights Tecknowledgist 7.6万回視聴・1年前

チャンネル登録 3433

qWertyWonka  
iamOTHER  
penpalsofficial  
YUTO MIZUNO  
ピアノ弾き語り...  
Ayaka Ikezawa  
他 25 件を表示

イチロー - トピック あなたにおすすめの動画

【貴重】イチロー ICHIRO イチロー vs 北野武 スペシャ... 43:35 8:19 2:39 1:02 4:51 9:58

イチロー オールスターで史上初のランニングホームラン 48万回視聴・2年前

イチローを突き飛ばし乱闘に発展 MLB Mariners ICHIRO 28万回視聴・2年前

イチロー人生初の退場処分の瞬間 MLB Mariners ICHIRO 115万回視聴・5日前

イチローサプライズ出演 浜田雅功仰天 smile design CH 386万回視聴・4年前

MLB2012 イチロー 8打数7安打4盗塁&決勝タイムリーの... 武田梅太郎 258万回視聴・5年前

# APIとは？？



とはいえ…

APIとは、世の中にある様々なサービスが提供しているデータを、自分の制作物で使えるようにするための決まりごとのことを指します。

決まりごとに従ってプログラミングを使って様々なデータを扱うことによって、制作するサービスをより高機能にすることができます。

# APIの事例



## Google Maps API

Google Mapsが提供する地図や地図に関する  
データをプログラミングで扱うことができます。

# APIの事例



## LINE Messaging API

自分だけのLINE botを作ることができます。

# APIの事例



ぐるなび Web サービス はじめての方へ API 仕様 ▾

新規アカウント発行

› 利用規約  
› 設定変更



ぐるなびが提供するグルメ情報に関する  
データを扱うことができます。

# APIの事例

ホーム モーメント 通知 メッセージ

いまどうしてる?

井上慎也@Webエンジニア @ino\_dev · 3時間  
何故政府が新元号の発表う?

あ、これ言っちゃダメな

Takkyu-san @Takkyu\_sa  
平日残業で忙しいと週末で遊ぶけど、新しい予定ぞ!(白目)

やまもとりゅうけん/嫁がヤバ

遅咲プログラマー (天然)凄い!

と素直に褒めてもくれな

稼いだ自慢するなよー

とフランクにいじっても

やまもとりゅうけん/嫁

会社設立したことある報告させられたんだって、居然が流れていのちやへつやえよりかづた。

twitter.com/yoshito410kam/...

ぐるたかさんと他のユーザーにフォローされています

本多大和 Yamato Honda

フォローする

知り合いを見つけましょう

Gmailからアドレス帳をインポートする

ケインコスゲ@バン...  
@keinkosuge

ツイート 5,251 フォロー 506 フォロワー 456

日本のトレンド 変更する

#アベマTVにポプテピックch開設  
本日4月1日、アベマTVに「ポプテピック」のみを放送する、ポプテピックchが開設。  
AbemaTV(アベマTV)@今日の番組表からによるプロモーション

エイプリルフール  
814,911件のツイート

#ラブライブ入学試験  
16,048件のツイート

#FGOQ  
91,873件のツイート

FGOクエスト  
14,702件のツイート

#乃々を探せ  
6,471件のツイート

#ひらがな推し  
11,565件のツイート

#乃木坂工事中  
31,321件のツイート

ナゾブル  
13,551件のツイート

キタユメ  
27,722件のツイート

## Twitter API

審査通過が必要ですが、通過すれば  
Twitterのつぶやき情報を扱うことができます。

# APIの事例

livedoor 天気情報 livedoorが提供するお天気情報

ヘルプ | livedoor

登録はお済ですか？

ドメイン取るならお名前.c...

> dentoripea.com > 利尻ヘアカラー使い方.com  
> 低血圧の妊婦さんに良い情報.com

ドメイン取るなら 公式登録サービス  
**お名前.com**  
http://www.onamae.com  
by GMO

天気予報 災害情報

Weather Hacks

天気予報 スポット天気 Weather Hacks

トップ > Weather Hacks > お天気Webサービス仕様

## Weather Hacks

ウェザーハックス

### お天気Webサービス仕様

お天気Webサービス (Livedoor Weather Web Service / LWWS) は、現在全国14  
さっての天気予報・予想気温と都道府県の天気概況情報を提供しています。

### リクエストパラメータ

JSONデータをリクエストする際のベースとなるURLは以下になります。

<http://weather.livedoor.com/forecast/webservice/json/v1>

このURLに下の表のパラメータを加え、実際にリクエストします。

パラメータ名	説明
city	地域別に定義されたID番号を表します。 リクエストする地域とidの対応は <a href="#">全国の地点定義表 (RSS)</a> 内の 「1次細分区 (cityタグ)」のidをご参照下さい。 (例・佐賀県 伊万里=410020)

### (例) 「福岡県・久留米の天気」を取得する場合

下記URLにアクセスしてJSONデータを取得します。

基本URL + 久留米のID (400040)

<http://weather.livedoor.com/forecast/webservice/json/v1?city=400040>

# お天気API

天気情報を扱うことができるようになります。  
審査もいらないので手軽に利用できます。

> dentoripea.com

ドメイン取るなら 公式登録サービス  
**お名前.com**  
http://www.onamae.com  
by GMO

# APIを使ってどんなものが作れる？

Google Maps API × ぐるなびAPI・・・近所のおすすめ飲食店を探せるアプリ

LINE Messaging API× お天気API・・・今日の天気を返してくれるLINE Bot

Twitter API× Google Maps API・・・特定場所のつぶやきを知るアプリ

# 今回のアプリの仕様

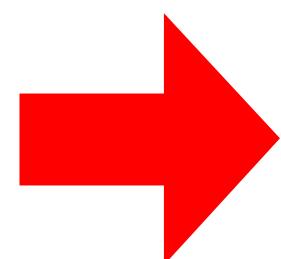


一覧が出てきて、曲が流れるので、  
流れた曲がどれになるのかを選ぶ！  
選んだ答えによって、正解不正解が表示される

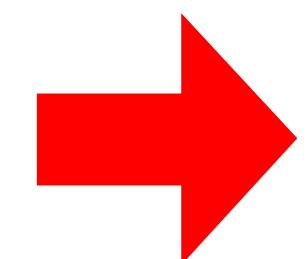
# quiz.htmlでのアプリの作成手順

## ■スタートボタンをクリックしたら

iTunes APIから  
100曲分データを取得



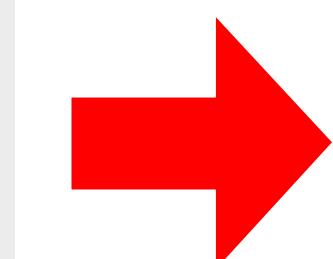
データをシャッフルして  
ランダムな3曲を取得



3曲の情報をブラウザに表示し  
クイズの正解を  
3曲の中からランダムに選択し  
正解の音声を鳴らす

## ■表示された曲の中からどれかを選んだら

選択した情報を確認



正解判断をして、  
○か×かを音付きで示す

# jQueryの色々な命令を体験しよう

**html('○○')** ・・・ 指定したHTMLの中に、○○を入れる。

(○○はHTMLタグ・テキスト・画像など)

**fadeIn(○○)** ・・・ 非表示になっているHTMLタグを、ふわっと表示させる。

(○○は時間の設定。1秒=1000で入力)

**fadeOut(○○)** ・・・ 表示されているHTMLタグを、ふわっと消す。

(○○は時間の設定。1秒=1000で入力)

**hide()** ・・・ 表示されているHTMLタグを、ぱぱっと消す

# Practice!! jQueryを使った復習

ボタンを押したらモーダルが削除されるようにしてみよう！

大事なのは

どのHTMLに対して

`$('')`

何をしてあげるのか？

`.show() .hide() .fadeOut()`など



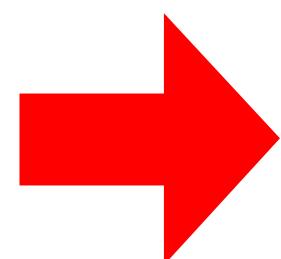
# 曲当てクイズアプリを作ってみよう

- 文法・基本的なルールを覚えよう -

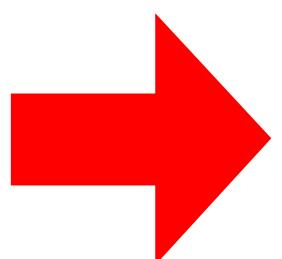
# アプリの作成手順

- スタートボタンをクリックしたら

iTunes APIから  
50曲分データを取得



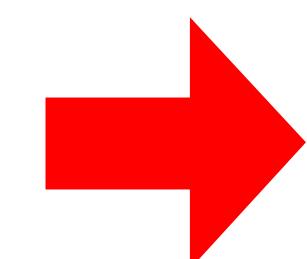
ランダムな4曲を取得



4曲の情報をブラウザに表示し  
クイズの正解を  
4曲の中からランダムに選択し  
正解の音声を鳴らす

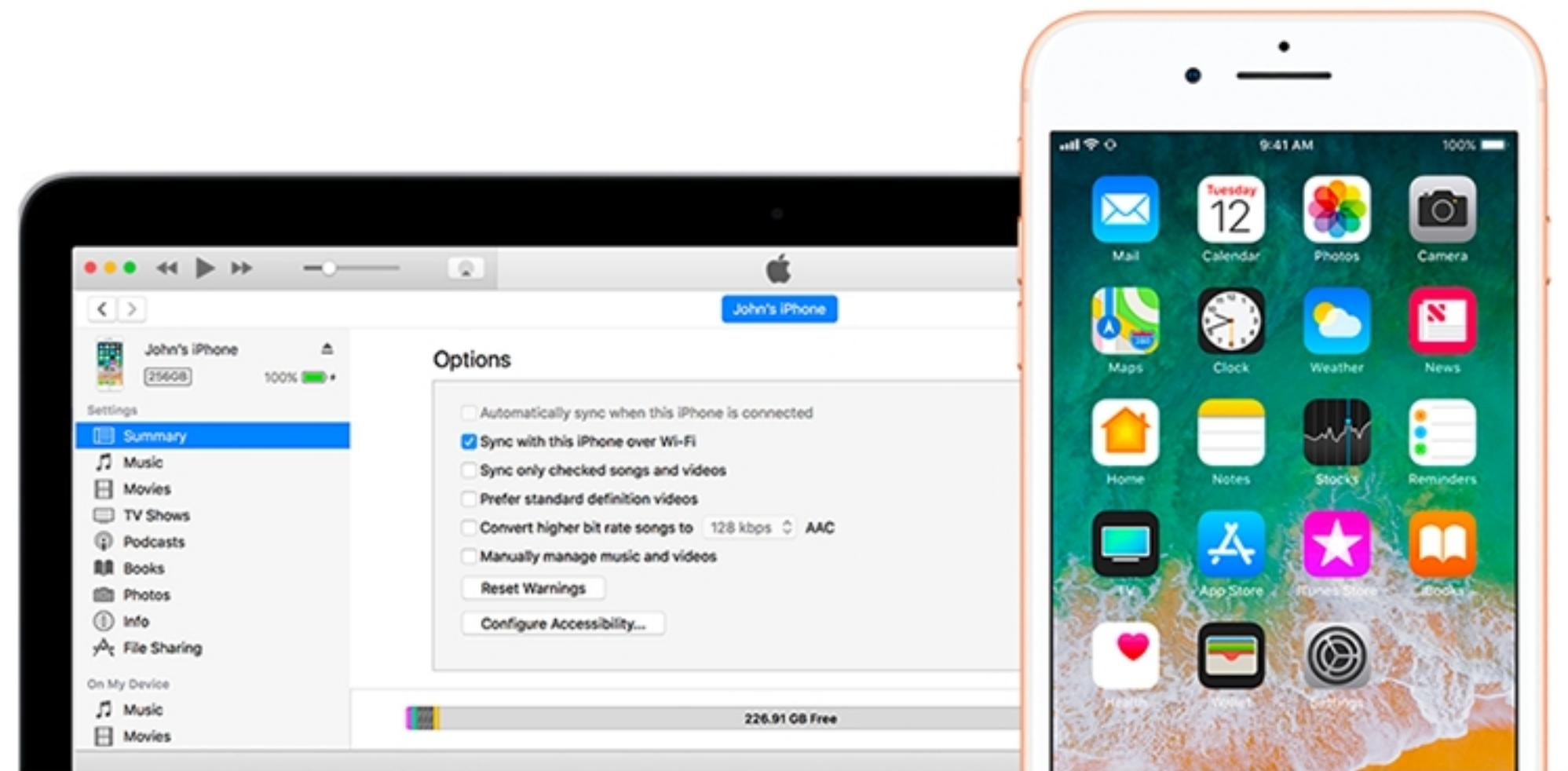
- 表示された曲の中からどれかを選んだら

選択した情報を確認



正解判断をして、  
○か×かを音付きで示す

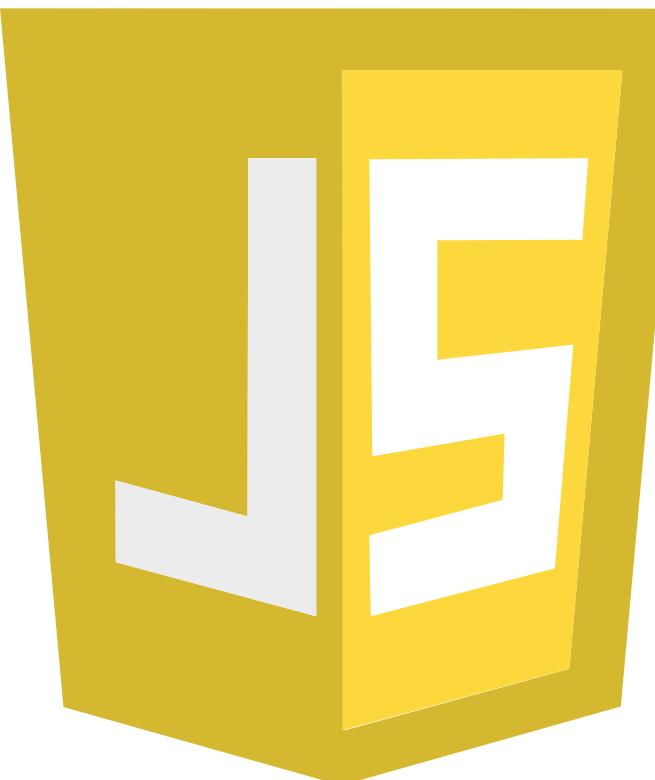
# 今回使用するAPI - iTunes Search API



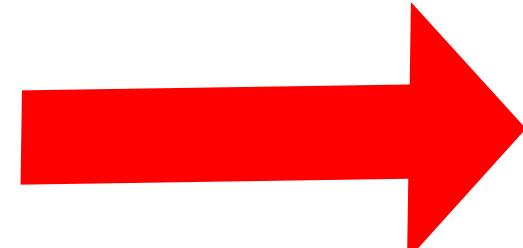
Appleが提供するiTunes内にあるデータを  
扱うことができるAPIです。

今回は、iTunes Musicが提供する音楽データの  
うち、試聴データを扱ってアプリを作ります！

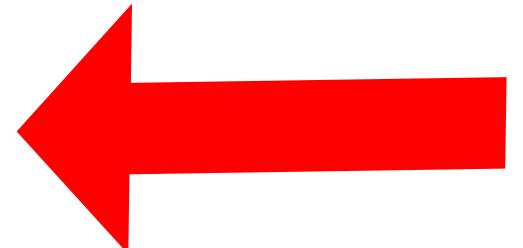
# JavaScriptでiTunes Search APIを扱う



iTunesさんお願ひ！  
試聴データを使わせて～！



通信が発生！



OK!いいよ！  
決まりごとに従ってデータを扱ってね！

# JavaScriptを使って通信をする - ajax -



- JavaScriptを使って、通信を行うことができる。
- ページの更新をしなくてもページの一部の情報を書き換えすることができる。

# ajaxプログラミングの仕組み



```
$getJSON('url',function(){  
});
```

• ajax通信をする



通信先のURLを入力  
(APIのデータが  
ある場所)



通信してデータとて  
きたら実行したい  
プログラムを書く

# ajaxの構文

JS

```
$.getJSON(  
  'https://itunes.apple.com/search',  
function(data){  
});
```

通信先のURLを指定する部分です。



# ajaxの構文

JS

```
$.getJSON(  
  'https://itunes.apple.com/search',  
function(data){  
});
```

通信が成功した時に  
プログラミングで何をしたいのか  
記述する部分です。



# iTunes Search APIの決まりごと

JS

```
const params = {  
    lang: 'ja_JP',  
    entry: 'music',  
    media: 'music',  
    country: 'JP',  
    term: $('#names').val(),  
    limit: 100  
};  
  
$.getJSON(  
    'https://itunes.apple.com/search',  
    'params',  
    function(data){  
        //...  
    }) ;
```

iTunes Search APIに対して  
「こんなデータがほしい！」と  
お願いごとをしている部分です、



# val()



**Song Intro Quiz!!!**

どのアーティストの曲でクイズをしましょうか！？入力してください！

アーティスト名を入力 例) 乃木坂46

クイズ開始

`$('●●●').val()`

目印をつけたHTML（**入力欄**）に設定された  
情報を取得することができる

# iTunes Search APIから取得したデータは

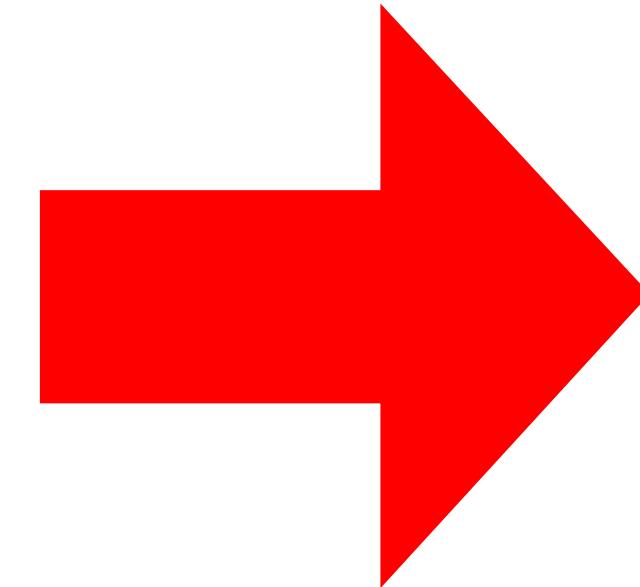
```
▼ {resultCount: 10, results: Array(10)} ⓘ  
  resultCount: 10  
  ▼ results: Array(10)  
    ► 0: {wrapperType: "track", kind: "song", artistId: 292706922, collectionId: 681020617, trackId: 681020904, ...}  
    ► 1: {wrapperType: "track", kind: "song", artistId: 292706922, collectionId: 385919647, trackId: 385919648, ...}  
    ► 2: {wrapperType: "track", kind: "song", artistId: 292706922, collectionId: 454109455, trackId: 454109457, ...}  
    ► 3: {wrapperType: "track", kind: "song", artistId: 292706922, collectionId: 435459289, trackId: 435459290, ...}  
    ► 4: {wrapperType: "track", kind: "song", artistId: 292706922, collectionId: 372367666, trackId: 372367684, ...}  
    ► 5: {wrapperType: "track", kind: "song", artistId: 292706922, collectionId: 397971665, trackId: 397971666, ...}  
    ► 6: {wrapperType: "track", kind: "song", artistId: 292706922, collectionId: 1062460735, trackId: 1062460737, ...}  
    ► 7: {wrapperType: "track", kind: "song", artistId: 292706922, collectionId: 569799129, trackId: 569799301, ...}  
    ► 8: {wrapperType: "track", kind: "song", artistId: 292706922, collectionId: 472169665, trackId: 472169666, ...}  
    ► 9: {wrapperType: "track", kind: "song", artistId: 292706922, collectionId: 525361128, trackId: 525361169, ...}  
    length: 10  
    ► __proto__: Array(0)  
  ► __proto__: Object
```

iTunes Search APIからデータ取得成功すると、「オブジェクト」と呼ばれるデータの固まりが返ってきます。返ってきたデータを使ってプログラミングで色々なことをしてみましょう！



# 例えば1曲目の情報を表示したいとき

```
$.getJSON(  
    'https://itunes.apple.com/search',  
    'params',  
    function [data] {  
    } );
```



Appleから送り返された情報が  
入っている

```
▼ {resultCount: 10, results: Array(10)} i data  
  resultCount: 10  
  ▼ results: Array(10)  
    ► 0: {wrapperType: "track", kind: "song", artistId: 292706922, collectionId: 681020617, trackId: 681020904, ...}  
    ► 1: {wrapperType: "track", kind: "song", artistId: 292706922, collectionId: 385919647, trackId: 385919648, ...}  
    ► 2: {wrapperType: "track", kind: "song", artistId: 292706922, collectionId: 454109455, trackId: 454109457, ...}  
    ► 3: {wrapperType: "track", kind: "song", artistId: 292706922, collectionId: 435459289, trackId: 435459290, ...}  
    ► 4: {wrapperType: "track", kind: "song", artistId: 292706922, collectionId: 372367666, trackId: 372367684, ...}  
    ► 5: {wrapperType: "track", kind: "song", artistId: 292706922, collectionId: 397971665, trackId: 397971666, ...}  
    ► 6: {wrapperType: "track", kind: "song", artistId: 292706922, collectionId: 1062460735, trackId: 1062460737, ...}  
    ► 7: {wrapperType: "track", kind: "song", artistId: 292706922, collectionId: 569799129, trackId: 569799301, ...}  
    ► 8: {wrapperType: "track", kind: "song", artistId: 292706922, collectionId: 472169665, trackId: 472169666, ...}  
    ► 9: {wrapperType: "track", kind: "song", artistId: 292706922, collectionId: 525361128, trackId: 525361169, ...}  
  length: 10
```

1つの曲の中にもたくさん情報が！



```
data.results[0].trackName
```

たとえば、1曲目の曲名を表示したい場合は  
四角で囲んでる部分を数珠繋ぎのようにして  
上記のように書いてあげることで実現できます！

※プログラミングは0から数え始めるので、  
1曲目 = 0と表現します

# Practice!! iTunes Search APIでデータ取得

iTunes Search API 経由で取得した曲データの中から  
さまざまな情報をconsole.log() を使って表示してみましょう！

- ・1曲目だけでなく他の曲情報を表示できるようにしてみましょう！
- ・曲名だけでなく色々な情報を表示できるようにしてみましょう！



# Practice !! 曲情報を画面に表示



`attr()`、`eq()`、そして今まで使った`html()`を使って、曲情報を4つ分画面に表示しましょう！

用意してあるHTMLを使っていただければOKです！

# 同じようなことを4回書くの面倒臭い・・・

```
const songImage = data.results[0].artworkUrl100;  
const songName = data.results[0].trackName;  
$(".song-card__image").eq(0).attr("src", songImage);  
$(".song-card__name").eq(0).html(songName);
```

•  
•  
•

4回続く



面倒だ・・・

```
const songImage = data.results[3].artworkUrl100;  
const songName = data.results[3].trackName;  
$(".song-card__image").eq(3).attr("src", songImage);  
$(".song-card__name").eq(3).html(songName);
```

# for文で繰り返し

JS

繰り返したい回数の設定

繰り返し行いたいプログラムをここに記載

for文は、特定のプログラムを繰り返し繰り返し行いたい時に  
使うと便利な文法です！！！

(丸括弧の中で設定された回数分だけ赤枠部分のプログラムを繰り返し実行する)



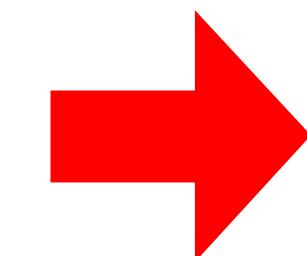
# 曲当てクイズアプリを作ってみよう

- 繰り返し処理を覚えよう -

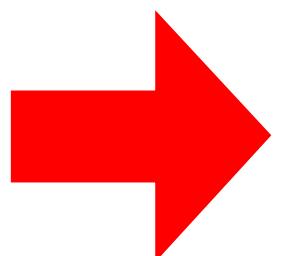
# アプリの作成手順

- スタートボタンをクリックしたら

iTunes APIから  
100曲分データを取得



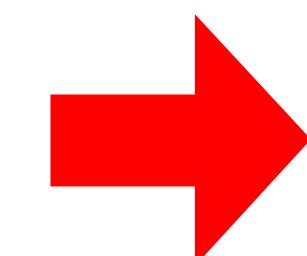
ランダムな4曲を取得



4曲の情報をブラウザに表示し  
クイズの正解を  
4曲の中からランダムに選択し  
正解の音声を鳴らす

- 表示された曲の中からどれかを選んだら

選択した情報を確認



正解判断をして、  
○か×かを音付きで示す

# 4曲の情報を画面に表示する



```
<div class="song-card">  
  <div class="song-card__item">  
      
    <p class="song-card__name">文字文字</p>  
  </div>
```

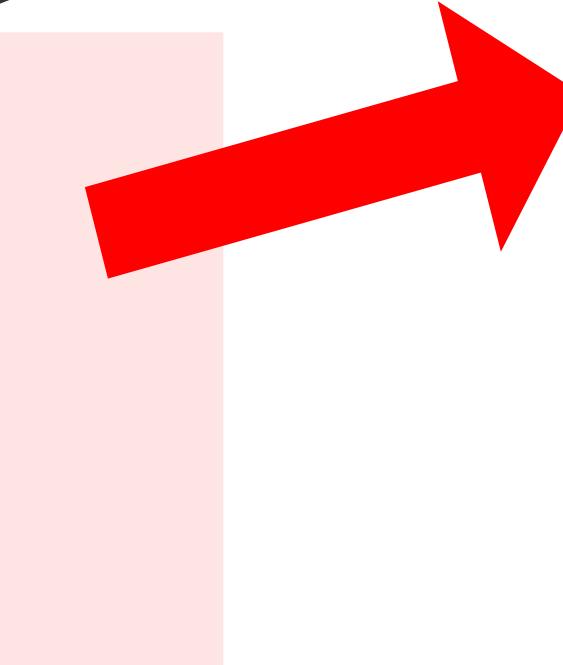
・・・これが4曲分並んでる



4曲の情報を画面に表示してみましょう！  
すでにHTML部分は書いてあるので、あとは  
jQueryを使って書き換えればOK!!!

# attr()

```
<div class="song-card">  
  <div class="song-card__item">  
      
    <p class="song-card__name">文字文字</p>  
  </div>
```



`$('●●●').attr('属性名', '書き換えた情報')`

目印をつけたHTMLタグの■■属性を書き換える



属性・・・タグについているさまざまな情報のこと

``

src属性

class属性

# eq()

```
$(".song-card__image").eq(0).attr("src", songImage);
```

同じclass名がついているHTMLが複数ある場合、

「●番目の.song-card\_\_imageに対して」命令をするという指定の仕方ができます。



属性・・・タグについているさまざまな情報のこと

```

```

src属性

class属性

# For文を使うことでスリムになる！！

```
const songImage = data.results[0].artworkUrl100;  
const songName = data.results[0].trackName;  
$(".song-card__image").eq(0).attr("src", songImage);  
$(".song-card__name").eq(0).html(songName);  
:  
: 4回続く
```

```
const songImage = data.results[3].artworkUrl100;  
const songName = data.results[3].trackName;  
$(".song-card__image").eq(3).attr("src", songImage);  
$(".song-card__name").eq(3).html(songName);
```

```
for(let i = 0; i < 3; i++){  
    const songImage = data.results[I].artworkUrl100;  
    const songName = data.results[I].trackName;  
    $(".song-card__image").eq(i).attr("src", songImage);  
    $(".song-card__name").eq(i).html(songName);  
}
```

# 音楽を鳴らすには・・・？



```
<audio src="" id="audio"></audio>
```

音楽データを扱うには、HTMLの「**audioタグ**」を使用します。

先ほど取得した試聴データをsrcの部分に入れて、play()という命令を記述することで  
試聴音楽データを鳴らすことが可能です。 (**`$('#audio')[0].play();`**)

```
$('#audio').attr("src", songslist[answerNumber].previewUrl + '#t=0.0,2.0');
```

→id="audio"がついているHTMLのsrcの部分に、取得データを設定

# 決まりごとに従った記述例



**data.results[0].previewUrl**

上記のように記述すると、「iTunes Search API」から取得したデータの固まりの中から  
1曲目の試聴音楽データを取得することができます。

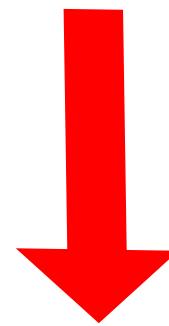
(プログラミングは0から数え始めるので、1曲目 = 0と表現します)

**data.results[randomNum].previewUrl + '#t=0.0,2.0'**

→試聴音楽データの1秒～2秒の部分のデータを取得

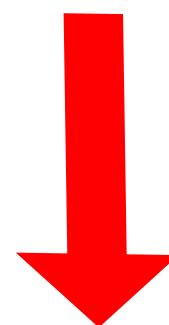
# Math.randomを再活用！

**Math.ceil( Math.random() \* 3 )**



1~3のうち、ランダムな数値データが作成される。

(1行に1つの命令 → 変数を使って数値データを保存)



数値データの結果に応じて、おみくじの結果を変える。



プログラムは基本的に

1行=1つの命令、と考えるとOK。

# 変数について

```
let test = 1;      グローバルで有効  
別変数と解釈される  
{  
    let test = 2; 波括弧内で有効  
}  
  
console.log(test);=>1
```

```
let test = 1;      グローバルで有効  
↑  
{ 同名なので上書き  
    test = 2;      let宣言ない場合、  
                    同名変数の中身を上書き  
}  
  
console.log(test);=>2
```

変数にはスコープがあります。  
波括弧 ({ }) の内側に書かれた変数は、  
その波括弧の外側から参照できません。  
(参照→保存されているデータの中身の確認)

波括弧 ({ }) の内側に書かれた変数で、  
外側に同名の変数がある場合は、中身を  
上書きします。

# Practice !! ランダムに音楽を鳴らす

占いの時にやった「Math.random()」を使って、  
ランダムに試聴音楽データを鳴らしてみましょう！

同時に、Math.random()を使って抽出した数字を  
変数answerNumberに保存させて、正解の番号として  
プログラムに覚えさせましょう！！！



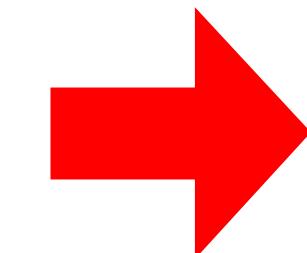
# 曲当てクイズアプリを作ってみよう

- 正解を表示しよう -

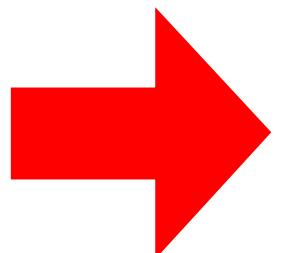
# アプリの作成手順

- スタートボタンをクリックしたら

iTunes APIから  
100曲分データを取得



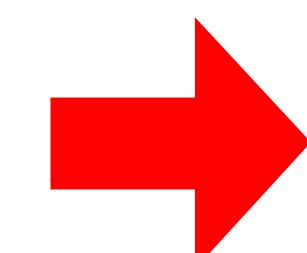
ランダムな4曲を取得



4曲の情報をブラウザに表示し  
クイズの正解を  
4曲の中からランダムに選択し  
正解の音声を鳴らす

- 表示された曲の中からどれかを選んだら

選択した情報を確認



正解判断をして、  
○か×かを音付きで示す

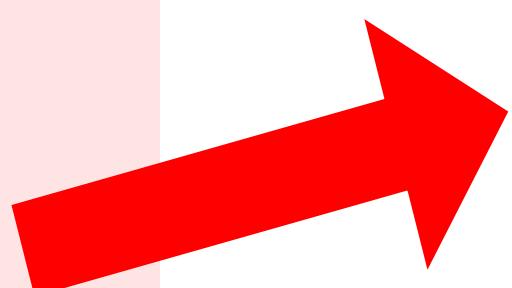
# 4曲から1つ選んで正解不正解表示



4曲表示されたものから1つだけを選択し、  
その曲が正解か不正解かを表示しましょう！

# index()

```
$(".song-card__item").on("click", function () {  
    const index = $(this).index();  
    console.log(index);  
});
```



\$(this)というの  
は「クリックしたHTML」  
のことを指します！



.song-card\_\_itemという目印が複数のHTMLについてる場合、上記のように書くと  
「何番目の.song-card\_\_item」を選択したのかを取得することができます

# if文を使って正解か不正解かで内容を変える

JS

```
if(選択した曲と正解が同じだったら){
```

```
}else{
```

```
}
```



おみくじのときと同じです！  
条件設定さえきちんとできれば、  
正解か不正解かによって違うプログラムを  
実行することができます！

# 仕上げ・条件設定



今回は、  
選んだ選択肢の順番 = 変数answerNumの数字だと  
正解という判断になります。

`answerNum == $(this).index()`

→鳴った曲 = 選んだ選択肢が同じ

# alert()



**alert("メッセージ")**

alert()は警告文形式でメッセージを表示することができる命令です。

# Practice !! 最後の仕上げ

アプリがきちんと動くかを確認してみましょう！！！

「もう余裕でできるぜ！」という人は、今回のアプリを発展させると  
どんな機能が追加できるか考えてみましょう！

- ・ 5問のクイズ形式にする。正解数を保存する。
- ・ 対戦形式にする。
- ・ 他のAPIを使ってみる。

