



APIを実際に使ってみよう！

HTML5プロフェッショナル認定試験 レベル1 ポイント解説無料セミナー

2019年1月
NTTテクノクロス株式会社
鈴木雅貴(SUZUKI Masataka)



自己紹介

- 鈴木雅貴（すずきまさたか）
 - NTTテクノクロス株式会社
 - レベル1,2 認定取得
- レベル1試験対策本執筆
- 弊社技術ブログでVR記事連載



HTML5プロフェッショナル認定試験 レベル1 対策テキスト&問題集

著者の1人です



本日は話すこと

- **試験範囲の確認**
- **JavaScript API概要**
- **各種APIを試してみよう**
 - **マルチメディアグラフィックスAPI (Video, Canvas)**
 - **オフラインストレージAPI (Web Storage)**

試験範囲の確認

- JavaScript APIは対象だが、JavaScriptのプログラミングは対象外
- APIで何ができるかが問われる
- 今回はAPIを使うプログラミングの解説なので対象外
- だが、実際に使うことで、APIでできることがイメージしやすくなる（と想着ていますので、その助けになるような内容としています）

JavaScript API概要

何をするものかを理解しよう



JavaScriptとは

- Webページの構成物（要素やスタイル）を操作できるプログラミング言語
 - 主にWebブラウザ上で動作する
- ECMAScriptとして仕様化されている実装の総称をJavaScriptと呼ぶ
- ES5、ES6、ES2018などはECMAScriptの略+バージョン番号



よくあるJavaScriptの用途

- フォーム入力内容チェック
- グラフ表示
- ボタンクリックで新しい要素追加・削除
- 再読込なしでページ内容更新
- アニメーション表示

- **script要素の内容として記述する**

```
<script>  
  // JavaScriptのコードを記述  
</script>
```

- **script要素のsrc属性で外部スクリプトファイルを指定して読み込む**

```
<!-- JavaScriptのコードを記述したscript.jsを読み込む -->  
<script src="js/script.js"></script>
```



- Application Programming Interfaceの略
- アプリケーションを作るにあたり、対象のさまざまな機能を利用するが、それらを楽に利用できるように用意されたもの

- Webブラウザの要素やスタイルの操作をJavaScriptから簡潔に利用できるよう、Webブラウザ上に用意されたもの
- 現在は、要素やスタイルの操作にとどまらず、マルチメディアコンテンツの操作、デバイスへのアクセス、通信、オフライン時のストレージ操作といった機能およびAPIがWebブラウザに実装されている
 - ので、特に準備なく使うことができる

- CodePenを使用
 - Webブラウザ上でHTML/CSS/JavaScriptを書いて確かめることができる
 - 左メニューの[Create]→[Pen]で新規作成
 - あとは書くだけで結果が表示されていく
 - 勉強の際のちょっとした確認に便利

Video

video要素を操作

- video要素で動画を再生可能
- そのまま再生するだけなら、ブラウザの組み込みプレイヤーから操作する
- JavaScript APIを使えば、組み込みプレイヤー以外から制御可能
- 動画の再生状況(どこまで再生したかなど)も取得可能

- <https://codepen.io/suzukima/pen/qmzRby>
- [再生]ボタンで再生、[停止]ボタンで停止

```
<video width="100%" controls src="動画ファイルURL"></video>
<div class="controller">
  <button type="button" class="btn btn-play">再生</button>
  <button type="button" class="btn btn-pause">停止</button>
</div>
```

- "var 変数名"で変数宣言
 - let や const もあるがここではあえてvarを使用
- 変数として宣言したものは、あとから利用可能

```
// 変数videoを宣言し、document.querySelector()の結果を入れている
var video = document.querySelector("video");

// 略

playBtn.addEventListener("click", function () {
  // 先程宣言した変数videoを利用
  video.play();
});
```

- 制御する要素を取得するAPIを使用
- "document.querySelector(セレクタ)"でセレクタを使った要素取得が可能

```
// セレクタ"video"で要素を取得し、変数videoに代入
var video = document.querySelector("video");

// 略

playBtn.addEventListener("click", function () {
  // 先程宣言した変数videoを利用
  video.play();
});
```

- イベント制御のAPIを使用
- “取得した要素.addEventListener()”で、取得した要素に指定したイベントが発生した際、実行する内容を指定

```
// playBtn(取得した要素を入れた変数)がクリックされたら、function()を実行
playBtn.addEventListener("click", function () {
  // 動画を再生
  video.play();
});
```

イベント名	概要
click	クリックされたとき
mouseenter	マウスカーソルが要素に重なったとき
mouseleave	マウスカーソルが要素から離れたとき
focus	要素がフォーカスされたとき
keydown	キーが押されたとき
keyup	キーが離されたとき
change	input等の値が変更されたとき
DOMContentLoaded	HTMLの読み込みと解析が終わったとき
load	画像等の読み込みが完全に終わったとき

- 取得したvideo要素を使い、APIで制御

```
// 再生ボタンクリック時の処理
playBtn.addEventListener("click", function () {
    // play()で再生
    video.play();
});

// 停止ボタンクリック時の処理
pauseBtn.addEventListener("click", function () {
    // pause()で停止
    video.pause();
});
```



video要素の制御について

- 以下が参考になります
- [動画と音声のコンテンツ | MDN](#)
- [HTMLMediaElement | MDN](#)

Web Storage

ブラウザにデータを保存し利用



Web Storage API

- ブラウザにデータを保存し、利用することができるAPI
- データはキーと値のペアで保存される
 - 例えばキーが"名前"で、値が"鈴木雅貴"
- かんたんな設定の保存などに使われる

- <https://codepen.io/suzukima/pen/QvXvZw>
- 指定した色が背景色になる
- 再読込しても背景色が保存されている



HTML

```
<input type="color" class="colorpallet">
```

- "document.getElementsByTagName(要素名)"で指定した要素を取得
- 複数取得するので、その1つ目という意味で後ろに[0]をつける

```
// body要素  
var bodyElm = document.getElementsByTagName('body')[0];
```

- 2種類あるストレージのうち、どちらを使うかを指定して宣言
 - localStorageはブラウザを閉じて開いてもデータが残っている
 - sessionStorageはブラウザを開いている間に限りデータが残っている(再読込を含む)

```
// ストレージの種類はlocalStorageに  
var storage = localStorage;
```

- "ストレージ.setItem(キー, 値)"で、ストレージに値を保存することができる
- 今回はキーをbgcolorとしている

```
// ストレージのbgcolorに値を保存  
storage.setItem('bgcolor', this.value);
```

- "指定したフォーム入力要素要素.value"で、フォームに入力された値を取得できる

```
// カラーパレットで選択されている色に変更されたら
colorPallet.addEventListener('change', function () {
  // カラーパレットで選択した色に背景色を変更
  bodyElm.style.backgroundColor = this.value;
  // カラーパレットで選択した色をストレージのbgcolorに保存
  storage.setItem('bgcolor', this.value);
});
```


- "if(条件) { 特定の処理 }"で、条件どおりのときに特定の処理が実施される

```
// ストレージにbgcolorがあるかどうかの条件文
if (storage.getItem('bgcolor')) {
  // ストレージにbgcolorがあれば、取得して背景色変更の処理
}
```

- "ストレージ.getItem(キー)"で、キーに対応する値を取得することができる

```
// ストレージからbgcolorを取得  
var color = storage.getItem('bgcolor');
```

- "取得した要素.style.プロパティ名"に値を指定することで、指定したプロパティの値を変更することができる

```
// body要素のbackground-colorを変更  
bodyElm.style.backgroundColor = color;
```

- "指定した要素.value"に値を指定することで、フォームの値を外から変更可能

```
// フォームで選択している色を変数colorに変更  
colorPallet.value = color;
```



Web Storageについて

- 以下が参考になります
 - [Web Storage API を使用する | MDN](#)

Canvas

ビットマップ画像の操作

- **canvas要素を対象に、JavaScriptで図形などを描画**
- **画像はビットマップ形式**
- **アニメーション、グラフ描画、写真編集、動画のリアルタイム加工などに使われる**



- <https://codepen.io/suzukima/pen/owxNMg>
- 線を引くだけ



HTML

```
<canvas id="canvas"></canvas>
```

- "取得したcanvas要素.getContext('2d')"で、描画可能な領域（描画コンテキスト）を作成できる

```
// canvas要素を取得  
var canvas = document.getElementById("canvas");  
// 描画コンテキストを取得  
var ctx = canvas.getContext("2d");
```

- "コンテキスト.beginPath()"でパスを初期化し、新規作成

```
// パスを初期化し、新しいパスを作成  
ctx.beginPath();
```

- “コンテキスト.moveTo(座標)”で、指定した座標へペンを移動
 - 座標はコンテキストの左上を(0,0)とする
- “コンテキスト.lineTo(座標)”で、現座標から指定した座標へ移動しつつパスを引く

```
// 指定した座標にペンを移動する  
ctx.moveTo(20, 30);  
// 指定した座標までパスを引く  
ctx.lineTo(120, 30);
```

- "コンテキスト.closePath()"で現在のパスを閉じる
- "コンテキスト.stroke()"で現パスを描画

```
// パスを閉じる  
ctx.closePath();  
// 図形を描画  
ctx.stroke();
```



Canvasについて

- 以下が参考になります
 - [canvas チュートリアル | MDN](#)

勉強方法

できれば触ってみましょう

- level1はプログラミングが対象外ですが、読んだだけではイメージがわきづらい
- CodePenのような手軽に試せる環境があるので、それを利用して少しでも試してみるのがおすすめ
 - そのほうが技術者としての力もつく
- [Mozilla Developer Network](#) は日本語も多く情報源として優秀

- 7割できればよい
- 全部覚えようとして焦るくらいなら、よく使いそうなところに絞って時間をかける
 - 時間があればその他を浅く
- 試験本番でわからないことがあっても焦らずに先へ進む
- 7割できればよい



LPI-JAPAN HTML5 Professional Certification

Open the Future with HTML5.

ご清聴ありがとうございました。

■ お問い合わせ ■

鈴木雅貴  suzuki.masataka@po.ntt-tx.co.jp