



# オープンソースカンファレンス 2020 Hokkaido

## HTML5プロフェッショナル認定試験 レベル1解説セミナー

～ 実務経験者も見落としがちなポイント解説します ～

2020年6月27日(土)@OSC 2020 Hokkaido

F( ) R K 株式会社フォーク  
齊藤 涼



# 自己紹介



## 齊藤 涼

株式会社フォーク 札幌ランチ所属  
フロントエンドエンジニア

2010

デザイン(紙媒体)の仕事をする

2013

Web業界に転職

2017

株式会社フォーク入社  
Webページ制作などの案件に携わりながら、  
HTML5プロフェッショナル認定試験レベル1取得

2020

レベル2取得に向けて勉強中



# 自己紹介

## 株式会社フォーク

@渋谷、札幌

(HTML5プロフェッショナル アカデミック認定校)

### HTML5プロフェッショナル認定試験の合格者

Lv1: **41**名

Lv2: **13**名

※2019年6月時点

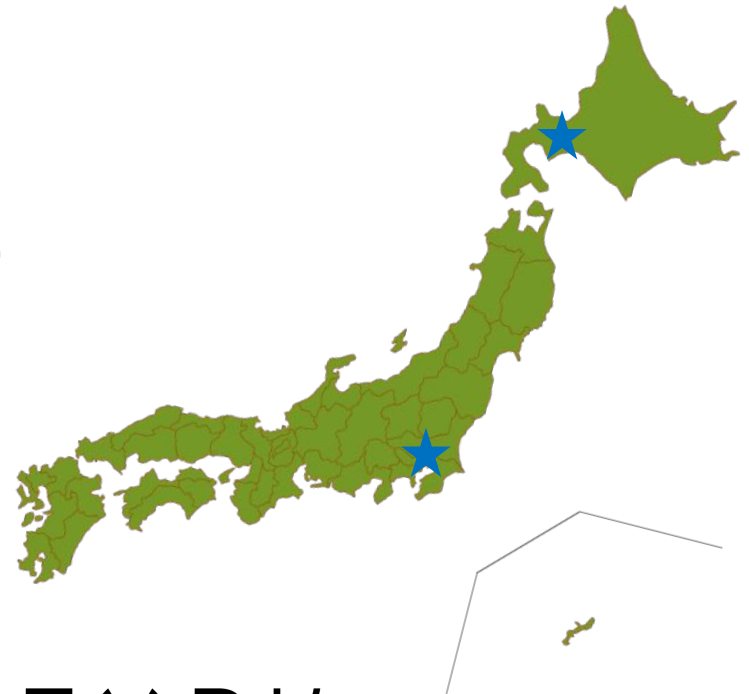


Lv1: **51**名

Lv2: **14**名

※2020年6月時点

私が所属する札幌ブランチでは、  
8名がレベル1に合格しています



# F( )RK

<https://www.fork.co.jp/>

<https://4009.jp/>



# 本日解説する内容

## ★ 試験概要、学習方法、対策の紹介

- 試験概要
- 学習方法
- 試験のポイント

## ★ 実務経験者でも見落とすポイント

- 実務と比較しての落とし穴

## ★ 資格を取得して活用できたこと

- 実務での活用
- ステップアップ



# 試験概要



# HTML5プロフェッショナル認定資格とは

**特定非営利活動法人エルピーアイジャパン(LPI-Japan)が、  
HTML5、CSS3、JavaScriptなど  
最新のマークアップに関する技術力と知識を、  
公平かつ厳正に、中立的な立場で認定する認定制度**



# 2つのレベル

今回は  
こちら!



## HTML5 レベル1

マルチデバイスに対応した静的なWebコンテンツを HTML5を使ってデザイン・作成できる。



## HTML5 レベル2

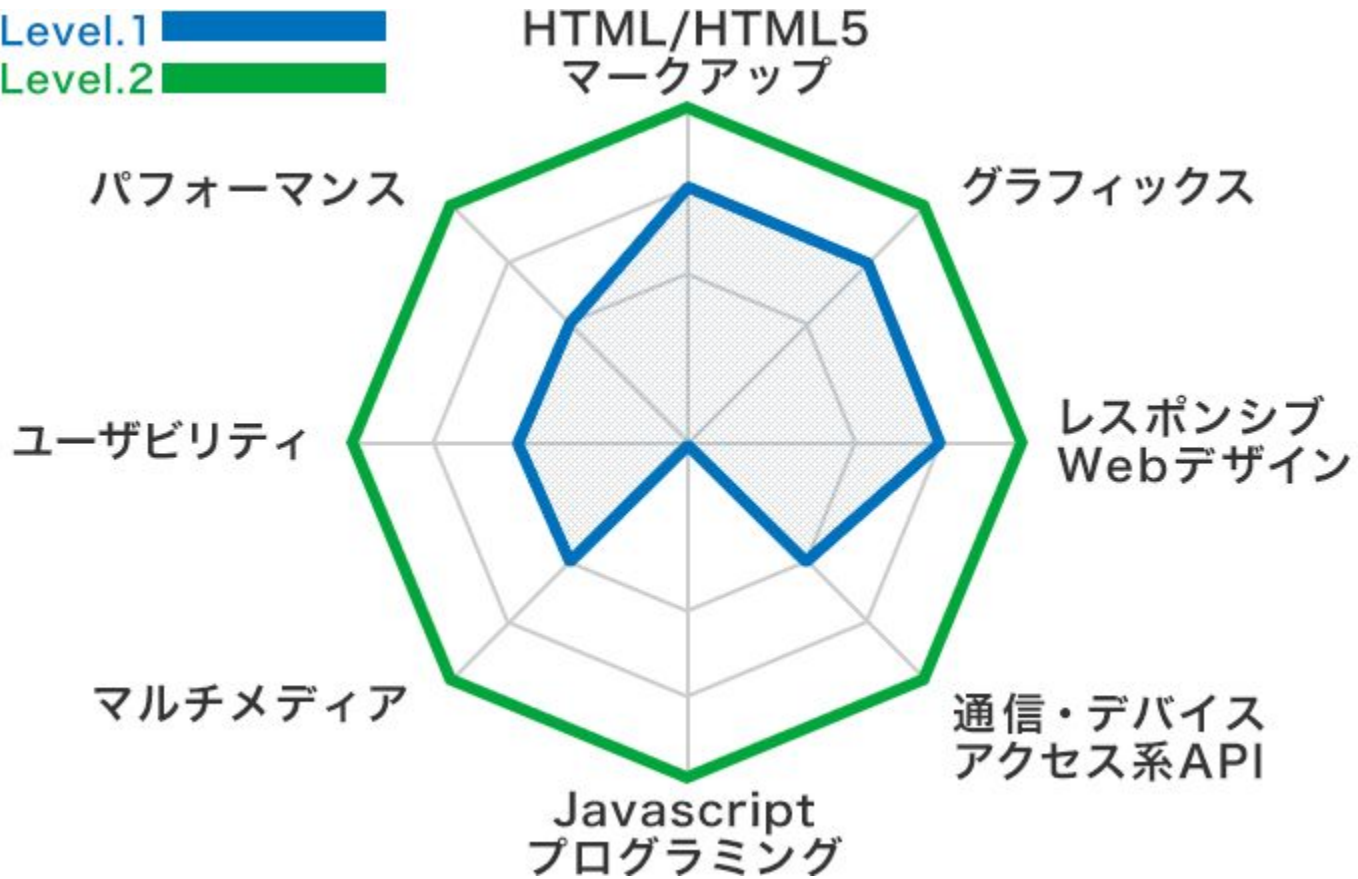
システム間連携や最新のマルチメディア技術に対応した Webアプリケーションや動的Webコンテンツの開発・設計ができる。





# 2つのレベル

Level.1   
Level.2 





# 試験の概要

試験実施方式	コンピュータベーステスト(CBT) ※ほとんどの問題が選択式
合否結果	試験終了と同時に
所要時間	90分(機密保持契約とアンケートの時間を含む)
問題数	約60問
合格ライン	約7割



# 学習方法



# 学習方法

## 公式サイトでの学習方法を参考にする

セミナーの紹介や、メールマガジン、サンプル問題があるので活用する

<https://html5exam.jp/measures/>

## 教本を使う

認定教材を読み込む

サンプル問題があったら、全問正解するまで挑戦する

## 実際にコードを書く

[CodePen](#)などの、即座に確認できるものを使えば捗ります

## わからないところは検索

HTML5の仕様などが掲載されているサイトがあるので、教本などと合わせて理解を深める



# 試験のポイント



# 出題範囲

**1.1 Webの基礎知識**

**1.2 CSS**

**1.3 要素**

**1.4 レスポンシブWebデザイン**

**1.5 APIの基礎知識**



# 1.1 Webの基礎知識

## 1.1 Webの基礎知識

### 1.1.1 HTTP, HTTPSプロトコル

### 1.1.2 HTMLの書式

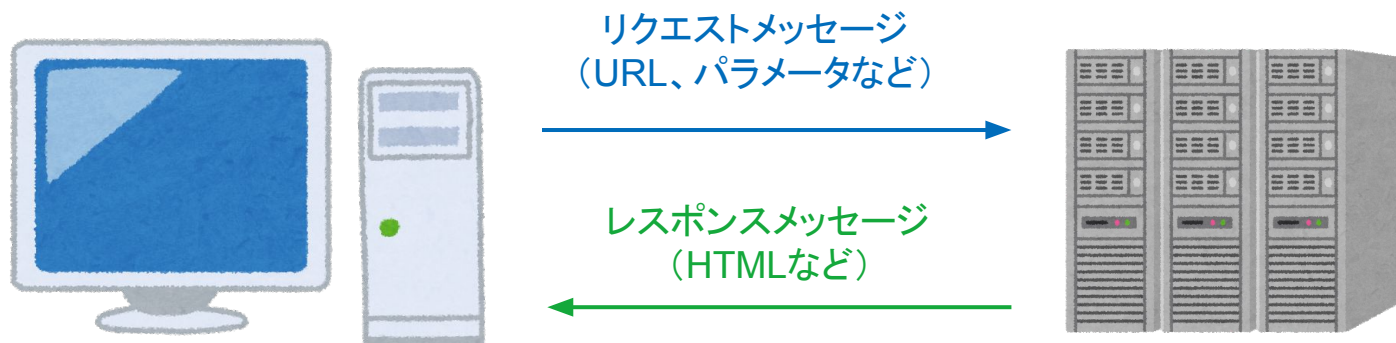
### 1.1.3 Web関連技術の概要

## HTTP通信とは

HTTP通信とは、ブラウザとサーバー間で、HTML文書などのテキストメッセージを受け渡すためのプロトコルです。

ブラウザ側から、**リクエストメッセージ**を送信し、サーバーから**レスポンスメッセージ**を受信します。

これにより、ブラウザにWebサイトの表示を行うことができます。







# 1.1 Webの基礎知識

## HTMLの構造

```
<!DOCTYPE html>
<html>
  <head>
    <title></title>
  </head>
  <body>
    <header></header>
    <main></main>
    <footer></footer>
  </body>
</html>
```

先頭にDOCTYPE宣言をし、そのあとにhtmlタグを設置し、各中身を設置します。  
DOCTYPE宣言は、大文字で書いても小文字で書いてもOKです。



# 1.2 CSS

## 1.2 CSS

### 1.2.1 スタイルシートの基本

### 1.2.2 CSSデザイン

### 1.2.3 カスケード(優先順位)

## カスケード(CSSの優先順位) - 1

CSSの優先順位は右の図のようになります。

基本的には、HTMLで読み込んだCSSが適用されますが、!importantを付けた場合は、強制的に優先度があがります。



## カスケード(CSSの優先順位) - 2

セレクタによる優先順位の設定

1桁目 → 要素

2桁目 → クラスセレクタ、属性セレクタ、擬似クラス

3桁目 → IDセレクタ

その個数によって、桁ごとに1が足されることになり、数値が高いものが優先されます。

## カスケード(CSSの優先順位) - 3

セレクタによる優先順位の設定

例)

body div#wrapper	102
div#wrapper	101
#wrapper	100
p.subtitle	011
.subtitle	010
p	001

優先度が設定され、数値が高ければ優先されます。



## ボックスの構造 - 1

HTMLの要素は、ボックスと呼ばれる四角い領域で表示されているのがほとんどです。

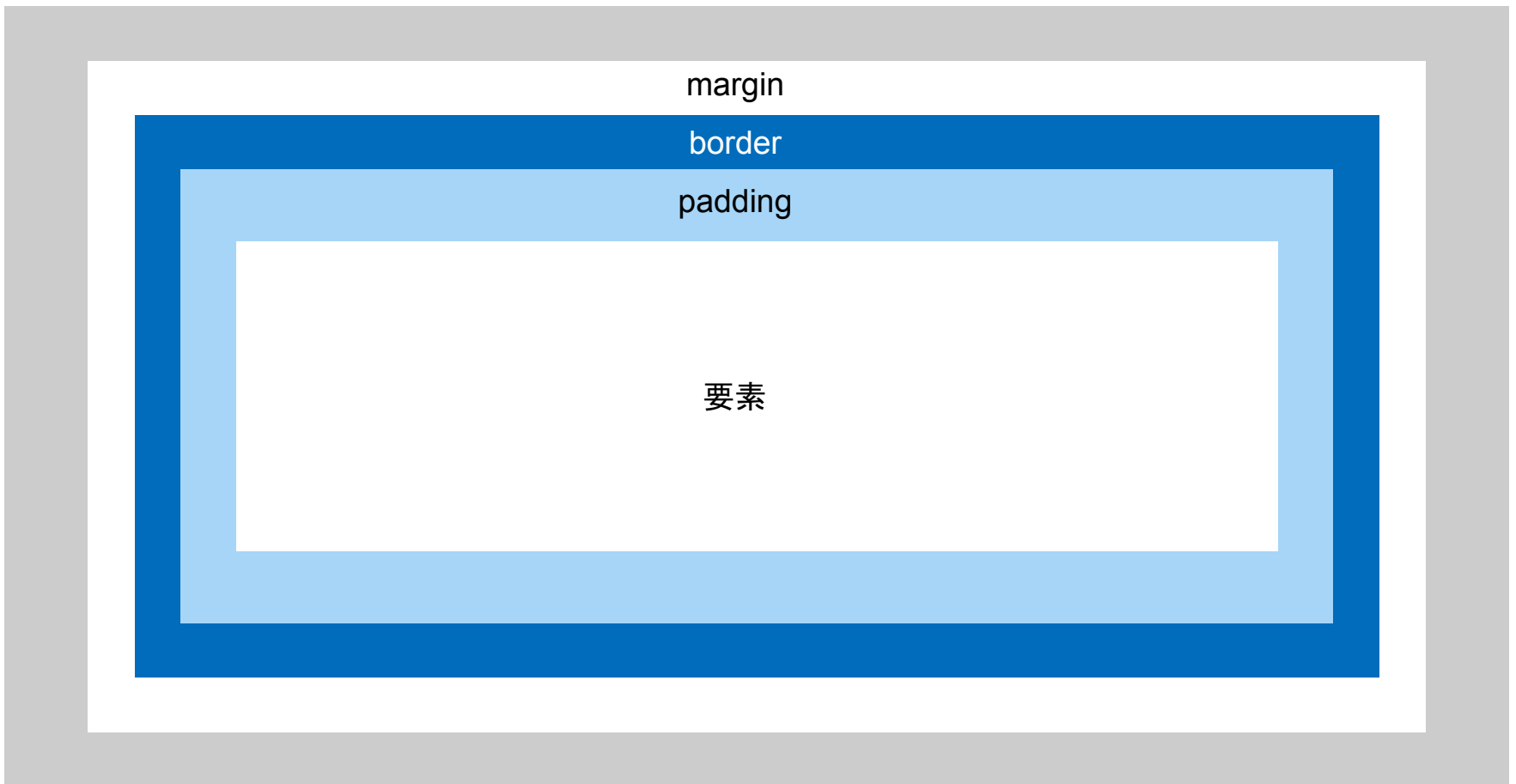
ボックスの余白をあけるCSSは、marginとpadding、borderになります。

**margin** → borderの**外側**の余白

**padding** → borderの**内側**の余白

**border** → ボックスの境界線（太さが幅に影響します）

## ボックスの構造 - 2



## ボックスの構造 - 3

marginとpaddingは、認識の違いで表示に大きな影響を与えることがあります。

例)

```
width: 240px;
```

```
padding: 0 10px;
```

実際の表示では260pxになります。

paddingではなくmarginを使うと、240pxのままとなります。





# 1.3 要素

## 1.3 要素

### 1.3.1 要素と属性の意味(セマンティクス)

### 1.3.2 メディア要素

### 1.3.3 インタラクティブ要素

### セクショニングコンテンツ - 1

セクションとは章・節・項のような文章のまとまった範囲のことを指します。

**article**、**section**、**nav**、**aside** の4つがセクショニングコンテンツとなります。

**article** → 章・節・項のような通常は見出しからはじまる一般的なセクション  
を示すための要素

**section** → 独立した内容となっているセクションを示すための要素

**nav** → ナビゲーション部分のセクションを示すための要素

**aside** → 補足情報などのメインコンテンツではないセクションを示すための  
要素

## セクショニングコンテンツ - 2

セクショニングコンテンツは、直接見た目に出るものではありません。

しかし、**アウトライン**というのが形成され、検索エンジンの解釈に影響がでます。

```
<body>
  <h1>見出しA</h1>
  <h2>見出しB</h2>
  <h2>見出しC</h2>
  <section>
    <h3>見出しD</h3>
  </section>
</body>
```



```
1.見出しA
   1.見出しB
   2.見出しC
   3.見出しD
```

### インタラクティブコンテンツ

インタラクティブコンテンツは、ユーザーが操作可能なコンテンツです。

**a**、**audio**、**button**、**details**、**embed**、**input** などなどが該当します。

細かいルールとして、

- ・**a**要素は、**href**属性が指定されている場合のみ

- ・**input**要素は、**type**属性が「**hidden**」以外の場合のみ

など、ユーザー側が操作できない状態の時は、含まれないので注意しましょう。



# 1.4 レスポンシブWebデザイン

## 1.4 レスポンシブWebデザイン

### 1.4.1 マルチデバイス対応ページの作成

### 1.4.2 メディアクエリ

### 1.4.3 スマートフォンサイト最適化



# 1.4 レスポンシブWebデザイン

## レスポンシブのメリット・デメリット

### メリット

- ・画面サイズでの切り替えとなるため、どのデバイスにも対応できる
- ・1つのHTMLで完結するため、リダイレクトの設定が不要
- ・URLが同じになるためSEO的に有利

### デメリット

- ・1つのHTMLにまとめるため、設計や構造を考える必要がある
- ・設計によっては、画像やCSSの読み込み量が多くなるため重たくなる可能性がある

## メディアクエリ

メディアクエリとは、デバイスの特性によってCSSを切り替えるものになります。

例)

```
<link rel="stylesheet" media="screen and (max-width: 480px)" href="style.css">
```

上記は、幅が480pxまでのデバイスに、style.cssを適用することができるようになります。

また、CSS内の記述によって分岐することも可能です。

```
@media screen and (max-width: 480px){  
    .foo{  
        margin: 100px 0;  
    }  
}
```

## Fluid Grid

ウィンドウ幅に合わせてコンテンツ幅を変更する技術となります。

幅に合わせることになるので、コンテンツ幅は、pxではなく%などの固定されない単位での設定が必要となります。

PC



スマートフォン







# 1.5 APIの基礎知識

## 1.5 APIの基礎知識

### 1.5.1 マルチメディア・グラフィックス系API概要

### 1.4.2 デバイスアクセス系API概要

### 1.4.3 オフライン・ストレージ系API概要

### 1.4.4 通信系API概要



# 1.5 APIの基礎知識

## そもそも、APIとは

wikiによると....

アプリケーションプログラミングインタフェース (API、英: Application Programming Interface) とは、ソフトウェアコンポーネントが互いにやりとりするのに使用するインタフェースの仕様である

↑ よくわかりませんね。

ざっくり言うと、

誰かが作った便利な機能を、自分の製作物に取り込むことが出来るものとなります。



# 1.5 APIの基礎知識

## Web Storage

ローカル環境(ブラウザ)にデータを保存するための仕組みです。

cookieに似てはいますが、大きな違いとして、保存できる容量が違います。

cookie → 4KBytes

Web Storage → 5MBytes

sessionStorageとlocalStorageの2種類のストレージが用意されています。

sessionStorage → ウィンドウやタブを閉じるまでデータを保持

localStorage → ウィンドウやタブを閉じててもデータを保持

## XMLHttpRequest

javascriptを用いてHTTP通信を行うAPIです。

画面遷移を行わずにHTTP通信を行い、各種データを取得することができます。

↓例えば、よく見る地図アプリケーションが想像しやすいです。





# 実務経験者でも見落とすポイント

実は私、試験に**1度落ちて**おります。

「見落とすポイント」というタイトルでしたが、

私自身の見落とししたところ、反省点に寄ったお話をいたします。



失敗したけど...



再チャレンジで合格！



# 普段使うことがない範囲

この試験は、HTMLやCSSだけではなく

- HTTP, HTTPSプロトコル
- APIの基礎知識

といった、コードを書くこととは直接関連することが少ない部分があります。



では、何をしたか.....



## 教本を元に勉強範囲を拡大

### HTTP通信での例

5XX → サーバー側でのエラー  
あとは重要そうな番号だけ覚える

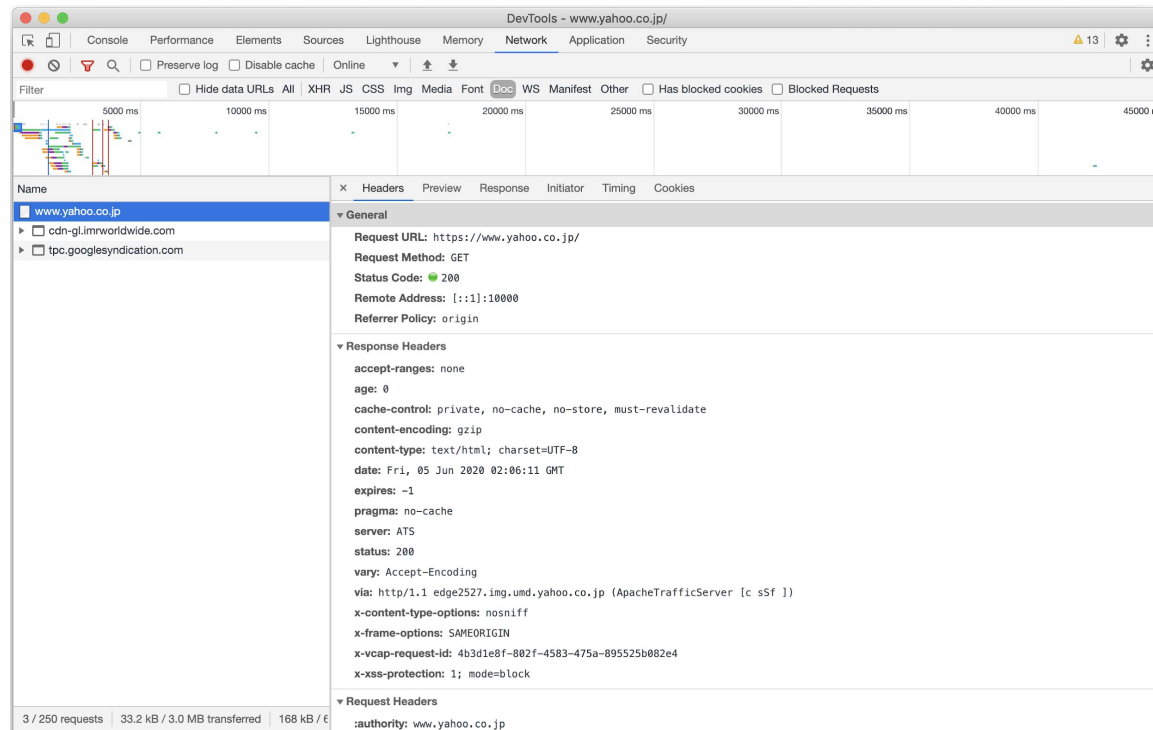


503 → サーバーはリクエストを処理する準備ができていない  
504 → サーバーが時間内にレスポンスを得られない  
505 → リクエストで使用した HTTP のバージョンをサーバーが対応していない

というように、1つ1つ詳細に覚えるようにし、勉強の範囲を拡大しました。

## ブラウザで使うことができる検証機能を使用

添付画像は、Chromeのデベロッパーツールを使ったものになります。



APIは、教本に書いてあるようなAPIが

- ・何に使われているのか
- ・他にどんなAPIがあるのか

ということを中心に調べ直しました。

## Geolocation API

ユーザーの位置情報を扱うためのAPIです。

下記の情報を取得することが出来ます。

- 緯度
- 経度
- 高度
- 緯度・経度の精度
- 高度の精度
- 方角
- 速度

これにより、地図アプリでは、現在位置の表示や、ナビゲーションに活用されることがあります。



# 勉強範囲の拡大

実際にやってみるのもオススメです。

例えば、Geolocation APIは下記のようにJavaScriptを設定すれば、緯度と経度をHTMLに書き出せます。

```
<script>
//ユーザーの現在の位置情報を取得
navigator.geolocation.getCurrentPosition(successCallback);

/***** ユーザーの現在の位置情報を取得*****/
function successCallback(position) {
  var gl_text = "緯度:" + position.coords.latitude + "<br>";
  gl_text += "経度:" + position.coords.longitude;
  document.getElementById("id").innerHTML = gl_text;
}
</script>
```

※document.getElementById("id")のidには、HTMLに設定したidを入れてください。

## HTMLやCSSは、あまり使わないところを補強

CSSも普段使っている時は、それぞれに下記のような名称があることを意識して使うことは少ないと思います。



文字にして説明するのは難しそうなところは、補強して勉強していくことがオススメです。



# わかっているでも対策を

「わかっている」という感覚に陥りがちになるので、下記3点をしっかりやっていくことをオススメします。

- ・いつもやっていることの補強
- ・いつもやっていないことは、教本より範囲を広げて勉強
- ・目に見えないことは実践して実感



# 資格を取得して活用できたこと



- 今まで、実務の中で勉強してきたので、改めて教本などを通して勉強することによって、理解が深まり、より正確な文書構造が書けるようになった
- 今まで使っていなかったタグを使うようになった
- 使う頻度が少なかった動画や音声の設置も、苦手意識が無く使えるようになった
- CSSの書けるプロパティの数が増えた
- CSS3のアニメーションの活用幅が広がった
- APIの活用方法を意識するようになった
- 人に教える時の引き出しが増えた

など、出来ることが大きく変わりました。

違う仕事でも活用できる場面は大いにあります。

**デザイナー** → UIに配慮のあるデザインが作成できる  
コードを配慮したデザインが作成できる

**ディレクター** → 出来る出来ないの判断が早くなる  
見積もりを暫定的に出せるようになる



# ステップアップ

## 今は他言語への勉強も....

活用とは違うのかもしれませんが、HTMLが次に繋がるきっかけになってくれました。

## 未来のために

リモートでの開催となりましたが、今回の出来事が、こういったリモートのような技術や、Webでの買い物などの、ITに関連したことが、災害が起きた時に役に立つというのが、知れ渡ることになったのかなと思っています。

HTMLだけでは、直接そういう役に立つことが出来るわけではないのですが、他の技術に隣接しているものだと思います。

今後、役に立つツールを作る、または、役に立つものが世の中に出てきた時に活用できるように、ぜひ受験して資格を取得していただければと思います。



**ご清聴ありがとうございました。**



# ご質問



# 公式アカウント、URL

## HTML5プロフェッショナル認定試験のサイト

<http://www.html5exam.jp/>

### Twitter

@html5Cert

### Facebook

<https://www.facebook.com/html5exam>

過去のセミナーでの配布資料や、イベント情報などが配信されています。

## LPI-JAPAN HTML5 Professional Certification

Open the Future with **HTML5**.