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8月

株式会社クリーク・アンド・リバー認定講師

高井 歩

本日の内容

試験の概要と勉強方法

1.5 APIの基礎知識 (知識問題・記述問題)

1.5.1 マルチメディア・グラフィックス系API概要

1.5.2 デバイスアクセス系API概要

1.5.3 オフライン・ストレージ系API概要

1.5.4 通信系API概要

HTML5プロフェッショナル認定資格とは

- 次世代のWebプロフェッショナルのスキルの向上に貢献するために、 HTML5を活用したWebページやWebアプリケーションなどのデザイン、設計、構築に関する体系だった知識とスキルを備えたHTML5のプロフェッショナルを中立的な立場で公平かつ厳正に認定する資格制度です。

- Webデザイナー、Webプログラマー、スマートフォンアプリ開発者、サーバーサイドエンジニアなどの、Web開発プロジェクトやWebサービスに関わるあらゆるプロフェッショナルが対象です。

- 多くの企業が推進する次世代Web言語の認定資格として、HTML5のプロフェッショナルのスキルの向上に役立ちます。また、企業内や研修機関での『技術力を担保する客観的基準』としても活用できす。

二つのレベル

HTML5 Level.1

Level.1

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

対象

- Webデザイナー
- グラフィックデザイナー
- フロントエンドプログラマー
- HTMLコーダー
- Webディレクター
- Webシステム開発者
- スマートフォンアプリ開発者
- サーバーサイドエンジニア

HTML5 Level.2

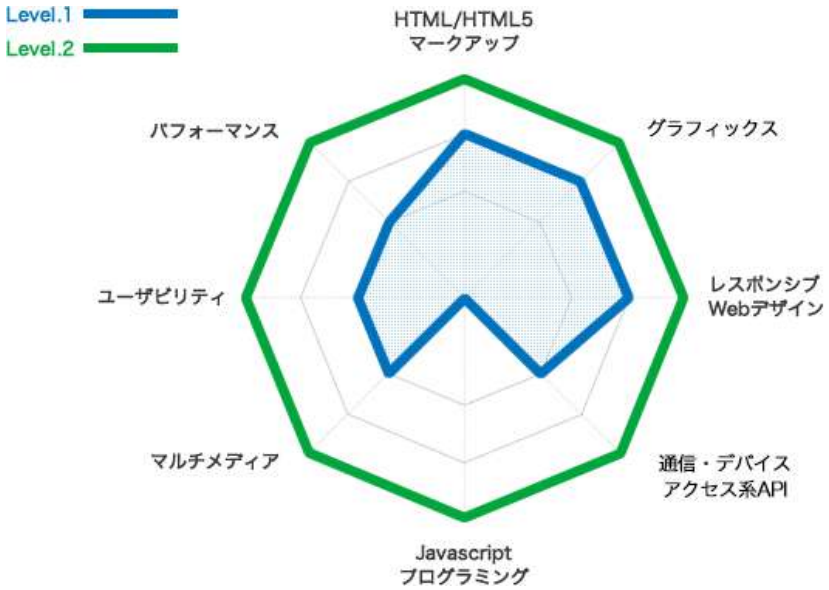
Level.2

最新のAPIを駆使したWebアプリケーションを設計・開発できる。

対象

- Webデザイナー
- フロントエンドプログラマー
- HTMLコーダー
- Webディレクター
- Webシステム開発者
- スマートフォンアプリ開発者
- サーバーサイドエンジニア

レベル1とレベル2の資格体系



HTML/HTML5マークアップ

HTML5に関するタグの用途、構造の組み立て方に関する技術

グラフィックス

JavascriptやCSSなどを用いて、動的にグラフィックスを生成したりアニメーションを実現したりする技術

レスポンシブWebデザイン

一つのソースで、スマートフォンなどの様々なデバイスの画面サイズに対応させるための技術

通信・デバイスアクセス系API

JavaScriptからクラウドと通信をして情報の送受信を行ったり、センサーなどのデバイスにアクセスしたりする技術

Javascriptプログラミング

Javascriptを使って、動的なWebコンテンツを作成する技術

マルチメディア

３D・動画・音声ファイルなどのマルチメディアコンテンツの表示・再生に関する技術

ユーザビリティ

JavaScriptやCSSなどを用いて、デザイン仕様に沿った見やすい表示や操作しやすいコンテンツを作成するための技術

パフォーマンス

ストレージや並列処理を使ってコンテンツを効率良く高速に動作させたり、オフラインでも動作する仕組みを作るための技術

レベル1とレベル2の資格体系

ベーシックレベル
HTML5プロフェッショナル向け

所要時間：90分
試験問題数：約60問
受験料：￥15,000 （税別）
認定条件： HTML5 レベル1試験に合格すること
認定の有意性の期限：5年間

認定名： HTML5 Level1 (Markup Professional)
試験名： HTML5 Level1 Exam

この資格の認定者は、下記のスキルと知識を持つWebプロフェッショナルであることを証明できます。

- **HTML5を使ってWebコンテンツを作成することができる。**
- **ユーザー体験を考慮したWEBコンテンツを設計・作成することができる。**
- **スマートフォンや組み込み機器など、ブラウザが利用可能な様々なデバイスに対応したコンテンツを制作できる。**
- **HTML5で何ができるか、どういった技術を使うべきかの広範囲の基礎知識を有する。**

アドバンストレベル
HTML5プロフェッショナル向け

所要時間：90分
試験問題数：40〜45問
受験料：￥15,000(税別)
認定条件：HTML5 レベル2試験に合格し、かつ有意な HTML5レベル1認定を保有していること。
認定の有意性の期限：5年間

認定名： HTML5 Level2 (Application Development Professional)
試験名： HTML5 Level2 Exam

この資格の認定者は、下記のスキルと知識を持つWebプロフェッショナルであることを証明できます。

- **動的に動作させて高いユーザビリティを実現するリッチユーザインターフェイスアプリケーションを作成することができる。**
- **マルチデバイスに対応し高パフォーマンスで動作する動的コンテンツを作成することができる。**
- **システム間連携を行いリアルタイムな情報を提供するアプリケーションを作成することがでる。**
- **スマートフォンなどでネイティブアプリに近い機能を組み込んだ先端のWebアプリケーションに　近い機能を組み込んだ先端のWebアプリケーションを開発することができる。**
- **APIのセキュリティモデルを理解したうえで開発することができる。**

レベル1の出題構成(1)

主題	項目	重要度
Webの基礎知識	HTTP,HTTPSプロトコル	8
	HTMLの書式	9
	Web関連技術の概要	6
CSS	スタイルシートの基本	7
	CSSデザイン	9
	カスケード(優先順位)	2
要素	要素と属性の意味(セマンティクス)	10
	メディア要素	6
	インタラクティブ要素	7

レベル1の出題構成(2)

主題	項目	重要度
レスポンシブWebデザイン	マルチデバイス対応ページの作成	4
	メディアクエリ	5
	スマートフォンサイト最適化	3
APIの基礎知識	**マルチメディア・グラフィックス系API概要**	**5**
	デバイスアクセス系API概要	**4**
	オフライン・ストレージ系API概要	**8**
	通信系API概要	**3**

学習方法

- 参考書
- サンプル問題
- 出題範囲を確認
  - 説明できない用語が無いようにする
  - HTML5では、似た用途で複数の機能、規格があるので整理する
- 自分でサンプルを作って確かめる。
  - Webブラウザ毎に動作が異なることがあるので注意
  - Webサーバの有無で動作が異なることがあるので注意

1.5.1 マルチメディア・グラフィックス系API概要

- **メディア関連要素**
  - video要素 , audio要素
- **ストリーミング**
  - HLS , MPEG-Dash , Media Source Extensions
- **DRM**
  - Encrypted Media Extends
- **ビットマップグラフィックス**
  - Canvas
- **ベクターグラフィックス**
  - SVG

メディア関連要素

- HTML5になって、Flashなどを使わずに映像や音声を再生できるようになった。
- 再生/停止などをJavaScriptから操作できる
- コントローラの表示の有無などを要素の属性で指定可能
- Webブラウザによって対応フォーマットが異なる

video要素

ソースファイルがひとつだけの場合

```
<video controls="controls" src="video/video.m4v"></video>
```

controls属性 再生ボタンなどの表示形式を指定

src属性 再生する映像ファイルを指定

Webブラウザによって再生可能なファイルが異なる場合
source要素を使用して、複数のファイルを列挙する。
上から順番にチェックして再生できるファイルを再生する。

```
<video controls="controls">
   <source src="video/video.webm" type="video/webm">
   <source src="video/audio.m4v" type="video/mp4">
</video>
```

audio要素

ソースファイルがひとつだけの場合

```
<audio controls="controls" src="audio/audio.mp3"></audio>
```

controls属性 再生ボタンなどの表示形式を指定

src属性 再生する映像ファイルを指定

Webブラウザによって再生可能なファイルが異なる場合
source要素を使用して、複数のファイルを列挙する。
上から順番にチェックして再生できるファイルを再生する。

```
<audio controls="controls">
   <source src="audio/audio.mp3" type="audio/mpeg">
   <source src="audio/audio.aac" type="audio/aac">
</audio>
```

ストリーミング

- ファイル全体をダウンロードしてからではなく、ダウンロードできた部分を順に再生していく方式をストリーミング再生と言う

- これまでは全てダウンロードしてから再生するか、Flash,Silverlightなどの拡張機能を使ってストリーミング再生を行なう必要があった

HLS

- **HTTPでのストリーミング方法はいくつかの会社が独自に開発したが互換性が無い**
  - Appleの作った**HLS (HTTP Live Streaming)**
  - Microsoftの作ったSS (Smooth Streaming)
  - Adobeの作ったHDS (HTTP Dynamic Streaming)
- **HLSはAppleが開発し、IETFで仕様策定されている**

MPEG-DASH

- **動画配信プロトコルの国際標準規格として策定された DASH(Dynamic Adaptive Streaming over HTTP)**
  - 対応ブラウザが少ないため現状で実用は難しい
  - Microsoft, Netflix, Google, Ericsson, Samsung, Adobe などが参加し、ISOで仕様策定

Media Source Extensions

- videoやaudio要素のsrc属性に、URLの替わりにJavaScriptのMediaSourceオブジェクトを使用
- MSEを使うと全部のファイルが届かなくても再生を開始可能
  (src属性にURLを使うとダウンロード完了してからの再生になる)
- 動画ファイル(WebM,MP4)や音声ファイルをストリーミング再生できるように保存形式を設定(セグメント化)する

DRM

- DRM(Digital Rights Management)は映像や音声に関する著作権保護のしくみ
- DRM付きのコンテンツを正規ユーザーになりすまして再生できないようにする
- データを暗号化して正規の鍵が無いと復号できないようにする

Encrypted Media Extensions

- HTML5でDRM(著作権管理)付きのコンテンツを再生するためにメディア要素を拡張する機能
- ライセンス管理などの仕組みは含まないため、ライセンス管理はサーバ側で行なう必要がある
- Microsoftによるサンプル(IE,Edgeのみ対応)
  - https://testdrive-archive.azurewebsites.net/HTML5/eme/

ビットマップグラフィックス

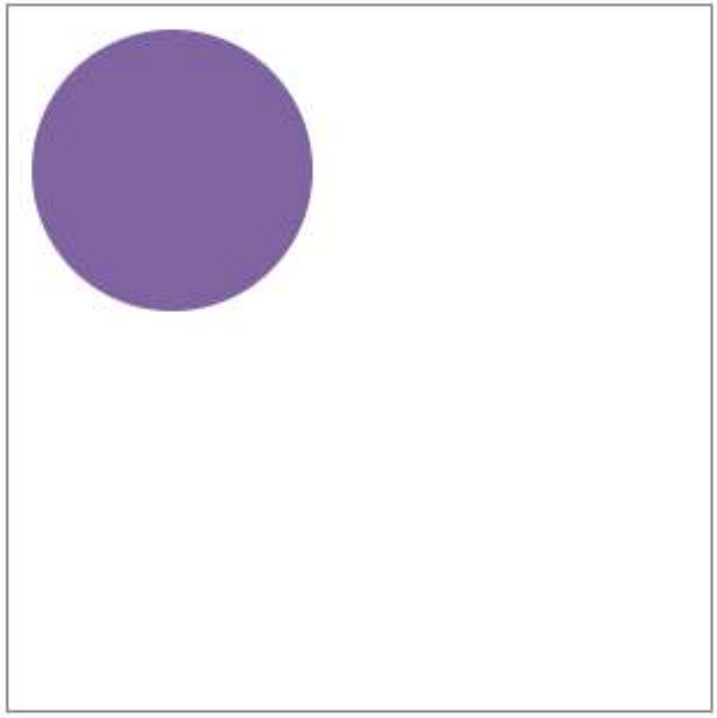
- **色の点のあつまりとして画像を扱う形式をビットマップという**

- **HTML5では、ビットマップ画像をimg要素などで表示するだけでは無く、作成や加工などもWebページ上で実現できる**
  (HTML5より前は、Flashなどを使用して実現)

Canvas

- ビットマップ画像を編集するにはcanvas要素をJavaScriptでコントロールする
- ピクセル単位での編集、図形の描画、画像ファイルの読み込み、画像ファイルの作成などができる
- 2次元画像のほか、WebGLによる3D表現にも使用される

参考: Canvasの使用例



```
<canvas id="cv" width="300" height="300"></canvas>
<script>
  // canvas要素とコンテキストの取得
  var cv = document.getElementById('cv');
  var ctx = cv.getContext('2d');
  // 色指定
 ctx.fillStyle = 'rgb(128,100,162)';
  // 図形の描画(円)
  ctx.beginPath(); // パスの開始
  ctx.arc(70, 70, 60, 0, Math.PI*2, false); // 円弧
  ctx.fill(); // 塗り潰し
</script>
```

ベクターグラフィックス

- 色の点で構成されるビットマップと違い、構成する図形の座標などで表わされる画像形式

- 拡大してもピクセルのジャギが見えにくい。

SVG

- SVGという形式のベクターグラフィックスを使用できる
- SVGはXMLの書式で記述し、HTML内に直接SVGを記述することも、img要素のsrcで読み込むこともできる
- CSSのbackground-imageに指定することも可能
- JavaScriptからSVG画像を構成する要素を操作できる

参考:SVGの使用例

```
<svg version="1.1" xmlns="http://www.w3.org/2000/svg" width="200" height="200" >
  <circle cx="95" cy="95" r="90" opacity="1"
              fill-opacity="0" stroke="#1017f1" stroke-width="3"/>
  <rect x="50" y="50" width="20" height="50" fill="#1d24f2"/>
  <rect x="120" y="50" width="20" height="50" fill="#1d24f2"/>
  <path d="M140 130C130 180 60 180 50 130"
              fill-opacity="0" stroke="#1017f1" stroke-width="7" />
</svg>
```

問題1

以下の関連する用語の組合せとして正しいものを選択しなさい。

A. ビットマップ: Canvas 、ベクターグラフィックス:SVG

B. Apple: MPEG-DASH 、Microsoft:HLS

C. 著作権保護: MSE 、ストリーミング:DRM

D. 映像: video要素、音声: sound要素

1.5.2 デバイスアクセス系API概要

- **位置情報**
  - Geolocation API
- **加速度センサー / ジャイロ**
  - DeviceOrientation Event , DeviceMotionEvent
- **入力デバイス**
  - Touch Events , Pointer Events

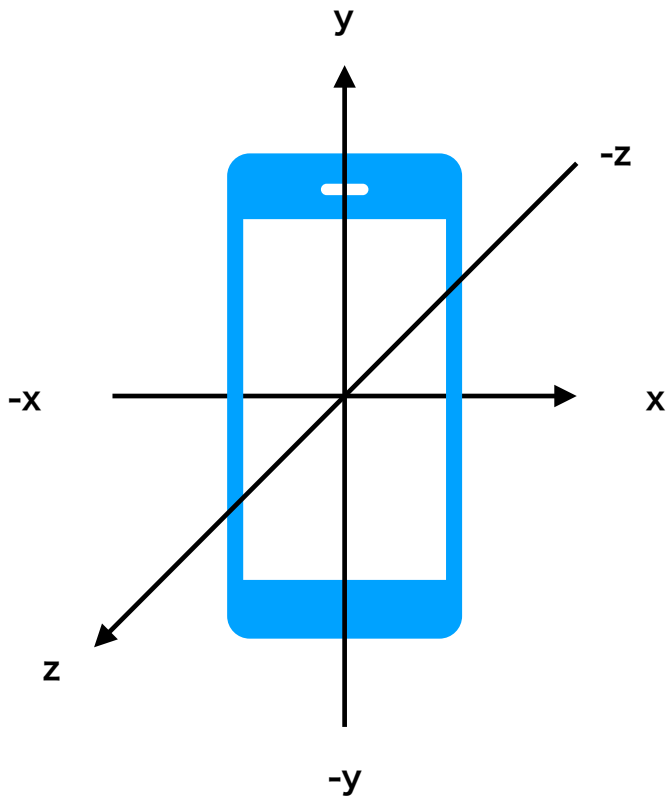
位置情報(Geolocation API)

- GPS(Global Positioning System)やWi-Fiなどを使用して現在位置を取得することができる
- 現在地の緯度、経度や移動速度を取得できる
- プライバシー保護の観点から位置情報の取得にはユーザの承認が必要。

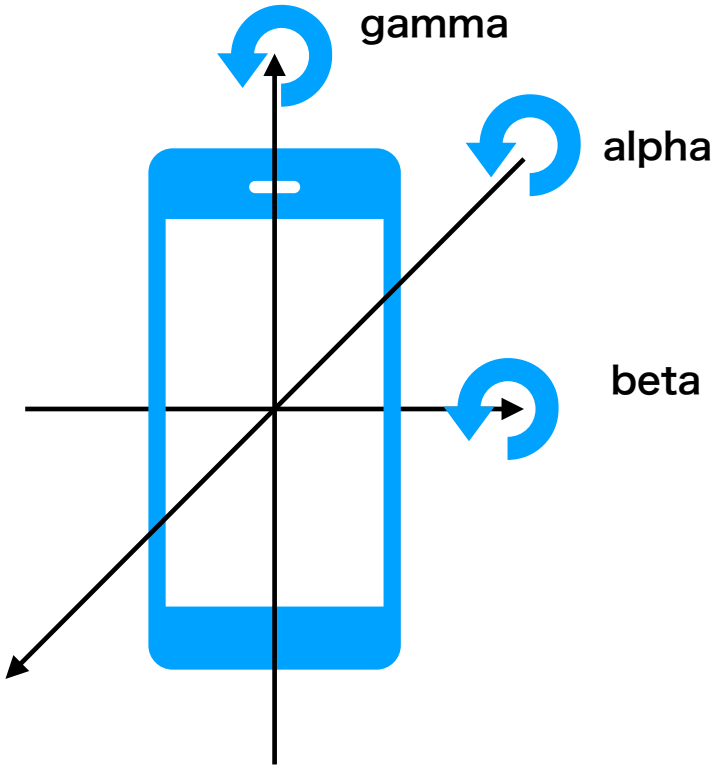
GeoLocation API

- **GeoLocation APIを使用すると、JavaScriptで以下の情報を取得できる**
  - **緯度** (北緯 0 ～ 90、南緯 0 ～ 90 / 赤道 0° から北極、南極 90° まで)
  - **経度** (東経 0 ～ 180、西経 0 ～ 180 / 旧グリニッジ天文台 0°から太平洋まで)
  - **緯度/経度の精度** (GPS搭載の有無などで精度が異なるため)
  - **高度と高度の精度** (m)
  - **方角**
  - **速度**
- **一度だけ取得することも、継続して取得しつづけることもできる**

加速度センサー / ジャイロ



ブラウザによって
座標系が異なるので
注意



加速度センサー

スマートフォンを動かした速度を取得

ジャイロ

スマートフォンを回転させた角度を取得

Device Orientation Event

- 加速度センサーやジャイロを使って Device Orientation Event で デバイスの傾きを取得できる
  - Alpha,Beta,Gamma,(iOSでは方角も)を取得できる
- 端末の位置や向きが変化したときにイベント発生
- 端末の動きを知るなら、Device Motion Eventsを使用する
  - 加速度や回転速度を取得

入力デバイス

- キーボードやマウスなどに対応するイベントはDOM3 Events(UI Events)で扱う
- スマートフォン向けに、マウスクリックではなくタッチによる操作を受けつける必要性が出てきた
  - マウスイベントではピンチなどのマルチタッチに対応できない

Touch Events

- スマートフォンやタブレットなどではTouch Eventsを使う
- マウスとタッチでイベントを使い分ける
  - マウスは onclick 、onmousemove など
  - タッチは touchstart , touchmove など
- タッチ操作にはホバー動作が無いのでUI設計で注意する

Pointer Events

- マウス、タッチ、ペン操作を統合的に扱うPointer Events
- DOM3 Events,Touch Eventsで扱える情報を含む上にペンの傾きを取得することもできる
- 一部のWebブラウザ(Safari,iOS系)では対応していないので注意

問題2

以下の選択肢のうち内容の正しいものを選択しなさい。

A.Geolocation APIではGPSの電波が届かない場所では位置が取得できない

B.Pointer Eventsを使うとペンの傾き情報が取得できる

C.DeviceOrientation Eventsでは加速度を取得する

D.UI EventsとTouch Eventsはマルチタッチに対応する

1.5.3 オフライン・ストレージ系API概要

- **データストレージの仕組み**
  - WebStorage , Indexed Database API
- **オフラインアプリケーション**
  - Application Cache , Service Workers
- **バックグラウンド処理**
  - Web Workers , Service Workers

データストレージの仕組み

- **HTML5より前は、ユーザーデータを保存するためにはサーバが必要**(もしくはCookieに4KB程度までのデータ保存)
- **HTML5では、Webブラウザにデータを保存するデータストレージ機能が追加され、クライアント毎に異なる設定などを保存できるようになった**
- **単機能な Web Strage と高機能なIndexed Database API**

Web Strage

- KVS(キーワードとデータを紐付ける)形式で、Webブラウザにデータを保存する
- ドメインに紐付いているため、他のドメインのWeb Storageは操作できない
- ローカルストレージとセッションストレージの2種類がある

Indexed Database API

- 高機能なデータベースとして使用できるIndexed Database
- トランザクションやインデックス機能が使えるため、一般的なデータベースとしての利用ができる
- オフライン状態でもデータにアクセスできる

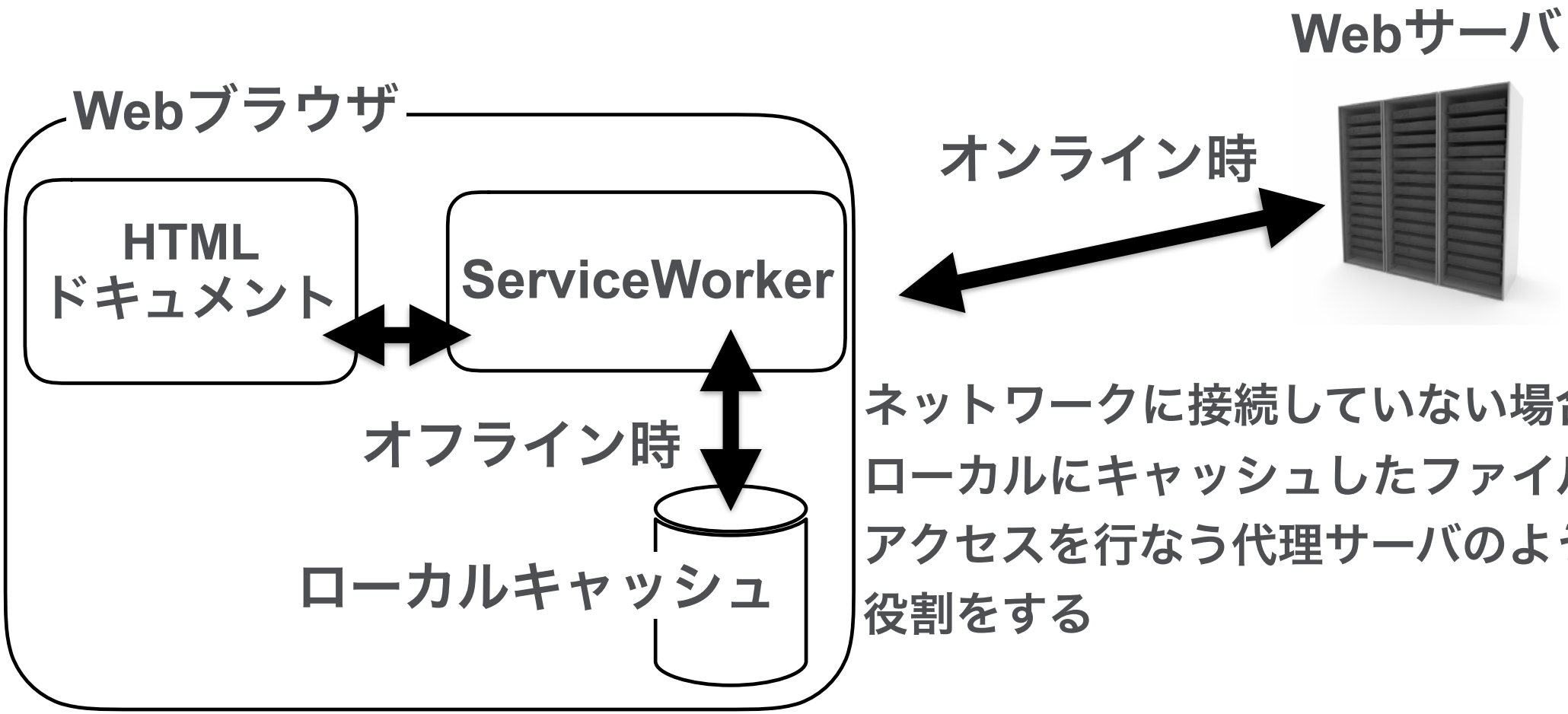
オフラインアプリケーション

- ネットワークに接続していない場合には、Webブラウザに保存していたデータを使ってWebページを表示できるようにする技術
- ネイティブなスマートフォンアプリのようなものをWeb技術を使って作成するときにも使用できる
- Application CacheからService Workersに移行中

Application Cache

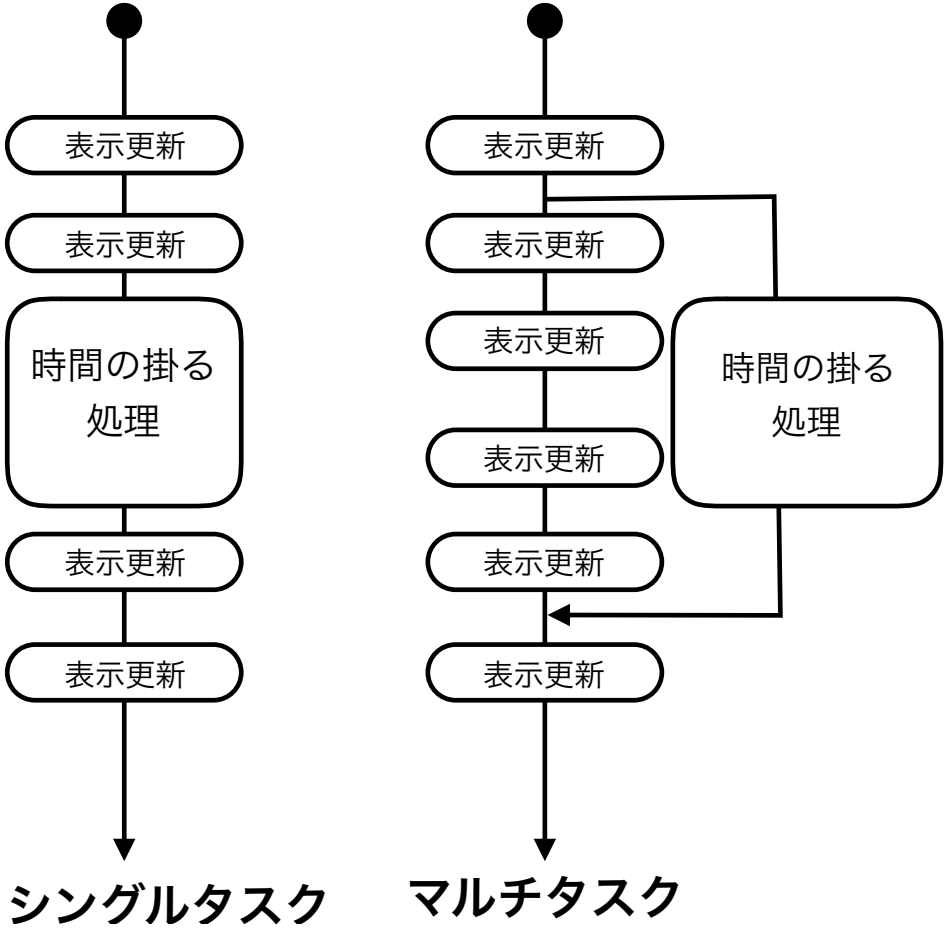
- キャッシュするファイルのリストを記述するマニフェストファイルと、HTMLへの設定が必要になる
- マニフェストファイルには以下の項目を記述
  - CACHE:　　… キャッシュするファイルを列挙
  - NETWORK: … 必ずネットワークから取得するファイル
  - FALLBACK: … 取得に失敗に際に使用されるファイル
- HTML5.1で仕様から削除(Service Workersに移行)

Service Workers



ネットワークに接続していない場合は、ローカルにキャッシュしたファイルにアクセスを行なう代理サーバのような役割をする

バックグラウンド処理



- 通常、JavaScriptでは一度にひとつの処理しか行えない(シングルタスク)ため時間の掛る処理をすると、画面表示やユーザ操作への対応が止まってしまう

- Worker(Web Worker,Service Worker)で並行処理することで複数の処理を同時に実行できる

Web Workers

- Web WorkersはJavaScriptの含まれているページが表示されている間、バックグラウンドでJavaScriptを実行できる
- Web Workersの中からはDOM(HTML要素)を操作できない

Service Workers

- Service Workersは処理の登録を行ない、ページが表示されていない状態でもバックグラウンドで実行される
- ページが表示されていない状態でもサーバからの通知を受け取ったり、更新されたコンテンツやデータをサーバから取得できる
- Service WorkerはセキュリティのためHTTPS通信でのみ動作
- Service Workersの中からはDOM(HTML要素)を操作できない

問題3

以下の選択肢のうち内容の正しいものを選択しなさい。

A. Service Workersはローカル PCにあるHTMLファイルを開いても使用できる

B. Web StorageはIndexed Databaseより高機能である

C. Application Cacheでは、キャッシュするファイルをマニフェストファイルに記述する

D. Web Worker でHTML要素を操作すると表示を高速化できる

1.5.4 通信系API概要

- **AJAX**
  - XmlHttpRequest
- **双方向データ リアルタイム通信**
  - WebSocket , WebRTC
- **サーバープッシュ**
  - Server-Sent Events

AJAX

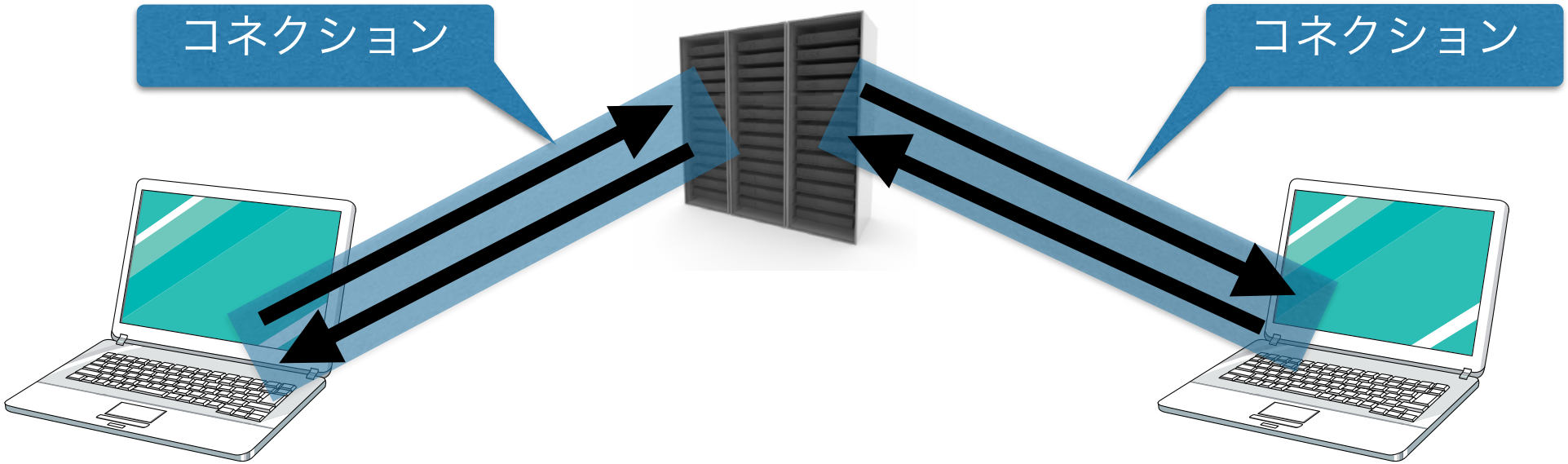
- Asynchronous JavaScript And XML
- 通常はWebページのリロードのタイミングでしかWebサーバと通信は行なえない(同期通信)
- JavaScriptを使うと任意のタイミングでWebサーバと通信を行なえる(非同期通信)。

XmlHttpRequest

- JavaScriptでXMLHttpRequestというオブジェクトを操作することで、AJAXを実現できる。
- 名前はXMLとついているが、XML以外のデータ形式も送受信できる
- クライアント(Webブラウザ)からのリクエストが無いとサーバからデータを送ることはできない

双方向データリアルタイム通信

一度接続すれば、リロードを必要としないでクライアントとサーバのどちらからでもデータの送信をおこなえる。



WebSocket API

- HTTPはリクエスト/レスポンスで接続を毎回切断するが、WebSocketは接続したまま双方向通信を行なえる
- HTTPと同じポート番号80を標準で使用するが、Webサーバではなく WebSocket用のサーバが必要
- バイナリデータを送受信できる

WebRTC

- 映像や音声のやりとりに特化した、WebRTC(Real Time Communication)
- Webブラウザ同士で通信(P2P)を行なうが、最初に相手と接続するためにシグナリングサーバを介する
- テキストデータやバイナリデータの送受信もできる

サーバープッシュ

- クライアント(Webブラウザ)からサーバに接続したあと、サーバ側から任意のタイミングでデータを送信できる
- Service Workerと組み合わせることでPush通知を実現できる
- サーバからクライアントへの単一方向通信

Server-Sent Events

- WebSocketと異なり既存のHTTPの枠組みの中で実装されている
- 接続を維持するためサーバの負荷が高くなる可能性がある
- IE,Edgeでの実装が遅れている

問題4

以下の選択肢のうち内容の正しいものを選択しなさい。

A. XmlHttpRequestではXML形式のデータしか送受信できない

B. Server-Sent Eventsでは双方向のデータ送信ができる

C. WebSocketは既存のWebサーバを使って通信が行なえる

D. WebRTCの主な目的はリアルタイムコミュニケーションである

本日の内容

試験の概要と勉強方法

1.5.1 マルチメディア・グラフィックス系API概要

1.5.2 デバイスアクセス系API概要

1.5.3 オフライン・ストレージ系API概要

1.5.4 通信系API概要

問題の答え: 1. A　2. B　3. C　4. D

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▪

株式会社 クリーク・アンド・リバー社

プロフェッショナルエデュケーションセンター[PEC]

E-mail ： pec@pr.cri.co.jp

http://www.c-place.ne.jp/

https://tlp.edulio.com/pec/seminars/