HTML5プロフェッショナル認定試験
試験対策講座

アシアル株式会社

HTML5プロフェッショナル認定試験

HTML5とは

- 2014年10月正式勧告
- マルチデバイス・マルチメディア対応
- リッチクライアント・アプリケーションのプラットフォーム
- 広義ではCSS3やJavaScriptによる3Dグラフィック、WebSocket、デバイスアクセス、クライアントストレージ等も含む

HTML5プロフェッショナル認定試験とは

- 特定非営利活動法人LPI-Japanが実施する、HTML5および周辺技術の知識レベルを測る認定制度です。

- 試験の難易度を示す2種類のレベルがあり、段階的に受験します。
  - Level1

    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、作成できるレベル

  - Level2

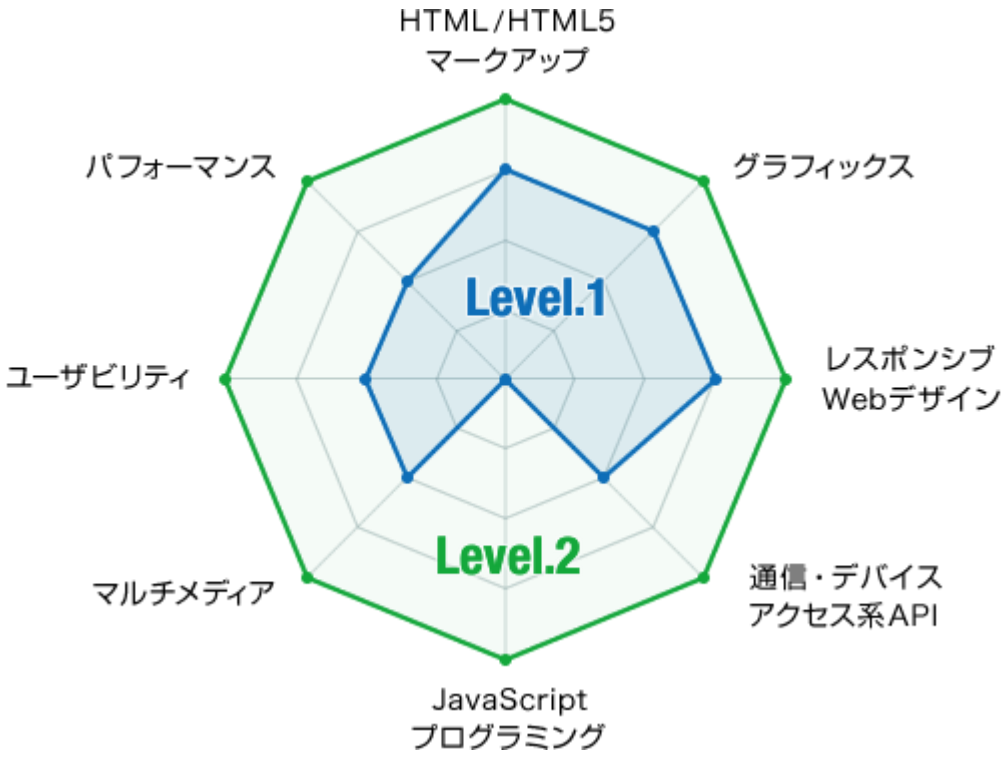
    システム間連携や最新のマルチメディア術に対応したWebアプリケーションや動的Webコンテンツの開発・設計ができるレベル

受験について

- 試験方式はコンピュータベーストテスト（CBT）です。試験配信会社の「ピアソンVUE」を通して受験します。
  - 全国のテストセンターで通年受験可能
  - 合否結果はその場でわかる
  - 試験の詳細は以下の通り

問題数	約65問
試験時間	90分
合格ライン	約7割
回答方式	殆どが選択式（複数回答あり） 記述式も１問程度
受験料	¥15,000（税抜）

Level１の試験範囲



※ Level.1 ではJavaScriptプログラミングの問題は出ません。

Level 1 の試験範囲

カテゴリ	重要度	項目	重要度
Webの基礎知識	23	HTTP, HTTPSプロトコル	8
		HTMLの書式	9
		Web関連技術の概要	6
CSS	18	スタイルシートの基本	7
		CSSデザイン	9
		カスケード	2
要素	23	要素と属性の意味	10
		メディア要素	6
		インタラクティブ要素	7
レスポンシブWebデザイン	12	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	20	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
合計	96		

出題率は目安であり、実際の試験では変動します。

Level 1 の試験攻略法

カテゴリ	重要度	攻略法
Webの基礎知識	23	『基礎知識』は知っていれば取りやすい。技術的な内容が非エンジニアにはハードルが高いかもしれないが実際に試せば何とかなる。
CSS	18	CSSの反映結果を想像できるようにしておく必要があるので『暗記』と『実験』の両方をバランスよく行う。
要素	23	『暗記系』が多いので毎日繰り返しテキストを読む。一夜漬けだとシンドイが期間を掛ければ取れる。
レスポンシブWebデザイン	12	『実験』しておけばある程度取れる。
APIの基礎知識	20	『基礎知識』なので、やはり知っていれば取りやすい。

Level 1 の試験対策：基本戦略

- 未経験者の場合
  - 範囲が広くて選択式（複数回答）なので、まずは色々なHTMLタグやCSSプロパティを試すところからスタート
- デザイナー系の場合
  - 『知識系』の問題は技術ネタが多い。配点がそこそこあるので調べたり実際に動かしておき点を取れるようにする。
- エンジニア系の場合
  - 『知識系』は有利。HTMLやCSSで点を落とさないように繰り返しの学習を行う。

学習リソース

- 出題範囲
  - http://html5exam.jp/outline/objectives_lv1_v2.html
- サンプル問題と解説
  - http://html5exam.jp/measures/sample.html
- HTML5学習に役立つ参考資料
  - http://html5exam.jp/contact/download.html
- 対策テキスト
  - 『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 Ver2.0対応版』
    - ✓ 受験バウチャーとセットで15%オフのキャンペーン実施中
      - – http://html5-campaign201709.peatix.com/

Webの基礎知識-HTTP, HTTPSプロトコル

出題範囲

説明（望まれるスキル）

- HTTPのコンテンツ作成や、サイト全体の設計を行うために必要なHTTPおよびHTTPSプロトコルに関する知識を有している。
- また、ブラウザでアクセスした時に返ってくるエラーコードの意味を理解できて、問題を解決するヒントとする事ができる。

主要な知識範囲

- ブラウザとWebサーバ間でやりとりされる通信内容や手順
- HTTPリクエストにおけるメソッド種類と違い
- リクエストURIの仕様について書式や利用可能文字
- Webサーバが返すレスポンスのヘッダ項目
- Webサーバが返すレスポンスのステータスコード
- HTTPプロトコルに規定されている認証方式

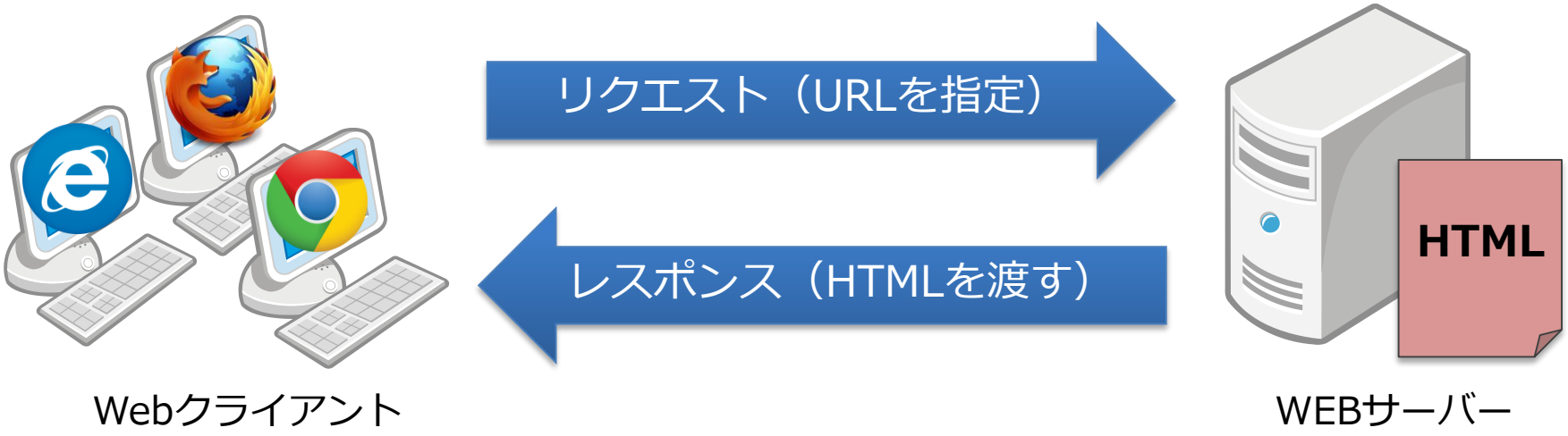
重要な技術要素

- HTTP, HTTPS, SSL/TLS
- リクエストメソッド（GET, POST, HEAD, PUT, DELETEなど）
- URI, URL
- ステータスコード, リダイレクト
- HTTP Header Fields（Accept, Authorization, Cache-Control, Content-Language, Expiresなど）
- Basic認証、Digest認証
- HTTP cookie

Webの仕組み

HTTP

- HTTPとは、Webクライアント（ブラウザ）とWebサーバー間でコンテンツを送受信するための通信方法を規定したもの
- WebクライアントからWebサーバーに対しての要求を「リクエスト」といい、それに対してWebサーバーから応答を返すことを「レスポンス」という



HTTPに関する頻出問題

- メソッドの種類（リクエスト）
  - GET/POST/HEAD/PUT/DELETE　など
- ステータスコードの種類（レスポンス）
  - 1xx　　情報
  - 2xx　　成功
  - 3xx　　転送
  - 4xx　　クライアントエラー
  - 5xx　　サーバーエラー
- ヘッダの種類（リクエスト・レスポンス）
  - User-Agent/Referer/Content-Type　など

HTTPヘッダーの例

- GETリクエスト

```
GET / HTTP/1.1
Host: s3.asial.co.jp
Connection: keep-alive
Cache-Control: no-cache
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64)
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp
Accept-Encoding: gzip, deflate
Accept-Language: ja,en-US;q=0.8,en;q=0.6
```

- レスポンス

```
HTTP/1.1 200 OK
Date: Mon, 02 Oct 2017 12:17:37 GMT
Last-Modified: Mon, 02 Oct 2017 12:17:35 GMT
Accept-Ranges: bytes
Content-Length: 51
Connection: close
Content-Type: text/html; charset=UTF-8
```


HTTPヘッダーの例

- POSTリクエスト

```
POST /html5/post.php HTTP/1.1
Host: s3.asial.co.jp
Connection: keep-alive
Content-Length: 57
Origin: http://s3.asial.co.jp

sei=%E5%B2%A1%E6%9C%AC&mei=%E9%9B%84%E6%A8%B9&name=submit
```

POSTで送信され値。
HTTPヘッダでは日本語を扱えないため
「URLエンコード」されます

重要なステータスコード

- 重要なステータスコード
  - 200 OK
    - ✓ リクエストに成功した場合に返すコード
  - 301 Moved Permanently
    - ✓ リソースが恒久的に移動した場合に返すコード
      - 移動先のURLをHTTPヘッダで受け取れる
  - 304 Not Modified
    - ✓ 前回のリクエストから更新されていない場合に返すコード
      - HTTPヘッダのみ返される
  - 404 Not Found
    - ✓ リソースが見つからない場合に返すコード
  - 500 Internal Server Error
    - ✓ サーバ側でエラーが発生して正常なリソースを返せない場合に返すコード

Location ヘッダーについて

- Location ヘッダーについて
  - HTTPレスポンスで利用されるヘッダです
  - 別のページへの「リダイレクト」に利用されます
    - ✓ 3xx系のステータスコードとセットで利用
- Demoページ
  - http://s3.asial.co.jp/html5/location.php
    - ✓ サーバー側からLocationヘッダーを送出しています
- サーバー側のプログラム

```
<?php
header("Location:https://www.asial.co.jp/");
?>
```

※「HTTPヘッダインジェクション」は、サーバ側のHTTPヘッダー送出処理を改ざんする攻撃です。サーバー側のプログラムに不備がある場合に実現可能です。

Location ヘッダーについて

- サーバー側のプログラム例

```
<?php
header("Location:https://www.asial.co.jp/");
?>
```

※「HTTPヘッダインジェクション」という攻撃があります。サーバ側のHTTPヘッダー送出処理を改ざんする攻撃です。サーバー側のプログラムに不備がある場合に実現可能です。

- HTTPレスポンス

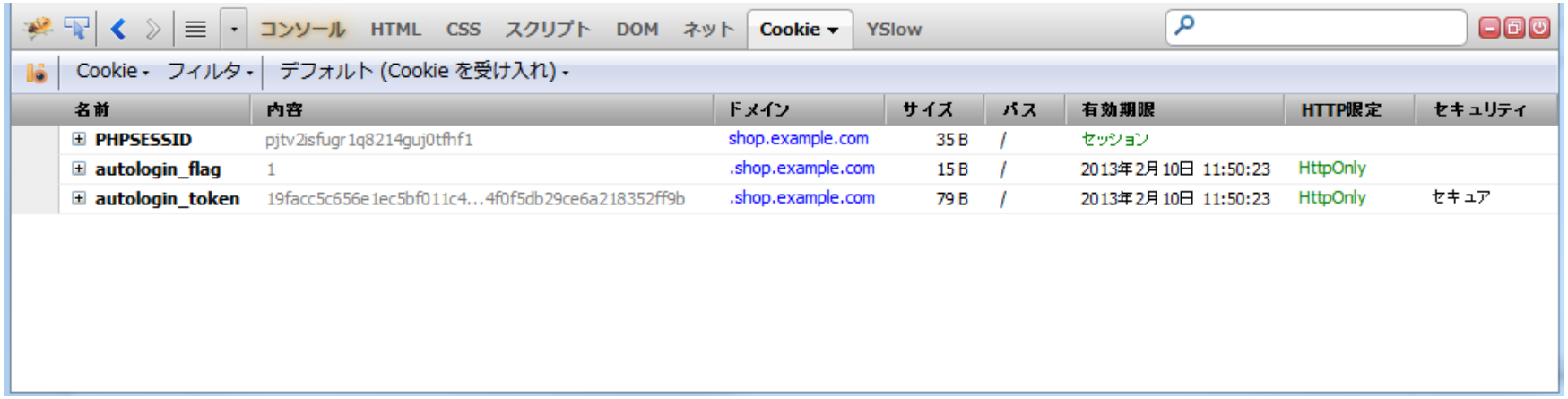
```
HTTP/1.1 302 Found
Date: Mon, 02 Oct 2017 12:49:34 GMT
Location: https://www.asial.co.jp/
Content-Length: 0
Connection: close
Content-Type: text/html; charset=UTF-8
```


ブラウザにデータを保存する方法

- クッキー
  - 昔から(HTTP 1.0)ある仕組み
  - キーバリュー型
  - 容量制限が厳しい
  - 一度セットするとブラウザからサーバーに毎回送信してしまう
  - 特に制限を掛けなければhttpとhttpsで共有される
- Web Storage
  - Cookieよりも大容量
  - 毎回送信してしまうこともない
  - 『オリジン』という単位で独立して管理（ドメイン・ポート・プロトコル）
- Indexed Database
  - インデックスを張ったりトランザクションをかけることができるストレージ

クッキーの詳細

- クッキーとは？
  - クッキーとは、サーバー側からWebブラウザに値を保存するための仕組み
    - ✓ HTTPヘッダーでやりとりされています
  - クッキーをセットされたブラウザは、同じドメインへアクセスすると自動的に値を返送する
  - ブラウザにセットされた値アhブラウザの開発ツールで確認することができる



Webの基礎知識-HTMLの書式

出題範囲

説明（望まれるスキル）

- 正しくブラウザにコンテンツを表示させるために、HTMLの仕様に沿った書式でHTMLコードを記述する事ができる

主要な知識範囲

- HTMLバージョン情報を含む文書型宣言に関する記述方法
- 要件に合わせた言語コードと、文字コード（符号化方式）の指定に関する記述方法
- HTMLで使用可能な文字参照に関する記述方法
- 必要に応じて、ヘッダ内に外部リソースを指定するリンクに関する記述方法
- 必要に応じて、ヘッダ内にメタ情報に関する記述方法

重要な技術要素

- 文書型宣言
- ISO-2022-JP, Shift_JIS, EUC-JP, UTF-8
- 文字実体参照
- <html>,<title>,<link>,<meta>

HTMLの書式

例

```
<!DOCTYPE html>
<html lang="ja">
    <head>
        <meta charset="UTF-8">
        <title>HTML5入門</title>
    </head>
    <body>
        <h1>HTML5とは</h1>
        <p>HTMLはWeb上で文章を公開するための技術です。HTML5のバージョンでは動画やインタラクティブなコンテンツもブラウザだけで表現でき、またアプリケーションプラットフォームとしての機能も多く備えています。</p>
    </body>
</html>
```

- 省略できる属性が増え、シンプルなシンタックスに
- セマンティックWeb対応（見た目ではなく、意味を重要視する）

HTML5文書の作成

DOCTYPE宣言

```
<!DOCTYPE html>
```

文字エンコーディング指定

- 文書の先頭にBOMを付加するか、以下のどちらかを<head>内に指定

```
<meta charset="UTF-8">
```

```
<meta http-equiv="Content-Type" content="text/html;charset=UTF-8">
```


Webの基礎知識-Web関連技術の概要

出題範囲

説明（望まれるスキル）

- 動的なWebコンテンツを作成するプロジェクトにおいて、どのような技術や対策を行っているのかを理解し、プロジェクト内で円滑にコミュニケーションできるために必要な知識を有している。
- Webコンテンツへのアクセスを伸ばす方法として、一般的に利用されているものについて説明する事ができる。

主要な知識範囲

- Webコンテンツを作成する際に使うスクリプト言語や画像ファイル、規格の概要
- Webに関する、セキュリティ脅威に関する技術の概要
- 要件に応じて、HTMLコンテンツ作成の際に理解が必要となるWeb関連技術の概要

重要な技術要素

- セッション
- Ajax
- インタレース, 画像ファイルフォーマット(BMP, PNG, JPEG, GIFなど)
- MVCアーキテクチャ
- Base64
- Data URI スキーム
- SQLインジェクション, XSS, CSRF, ディレクトリ・トラバーサル, HTTP ヘッダ・インジェクション
- DOM
- マイクロデータ, カスタムデータ属性

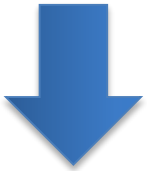
SQLインジェクション

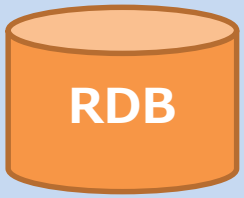
- SQLインジェクションとは？
  - データベースサーバに発行する問い合わせに、悪意のあるSQL（データの不正取得、削除など）を挿入する攻撃

SQLインジェクション

- 正常な問合せ

①一般ユーザーの操作
「ID」を表示
http://shop.example.com/item.php?id=5



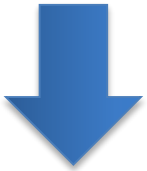


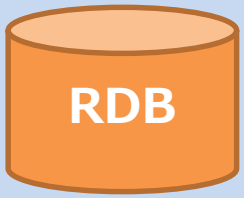
②データベース側での処理
SELECT * FROM items WHERE id = '5';

SQLインジェクション

- 不正な問合せ

①攻撃者の操作
「ID」を表示
http://shop.example.com/item.php?id=' OR 1 = 1;--





②データベース側での処理
SELECT * FROM items WHERE id = '' OR 1 = 1;--';

idが空文字か1==1（実質検索条件なし）、--以降は無効化

全データが抽出されてしまう！

SQLインジェクション

- 対策方法①
  - 特殊文字をエスケープする
    - 開発言語が用意しているエスケープ関数などを利用する

- 対策方法②
  - あらかじめSQLを用意しておき、可変の部分だけを置き換える「プリペアードステートメント」という仕組みを利用する

XSS

- XSSとは？
  - 掲示板などの、ユーザーの登録した文字列が公開される場所にスクリプトを埋め込む攻撃
  - 主な被害
    - 正規ユーザーのクッキー盗難
    - フィッシングサイトへの誘導
    - サイト改ざん
  - 正式名称
    - Cross Site Scripting

XSS

- XSSの例

掲示板

②スクリプトを含んだ文章が掲載される

```
<script>
    location.href = "http://crack.example.com"
</script>
```

①スクリプトを投稿

③閲覧

攻撃者

一般ユーザー

別のサイトに誘導されてしまう！

XSS

対策方法

- HTMLの生成時に、サニタイズ（タグの構成文字等の特殊文字をHTMLエンティティに変換すること）を行う
- HTMLエンティティ一覧

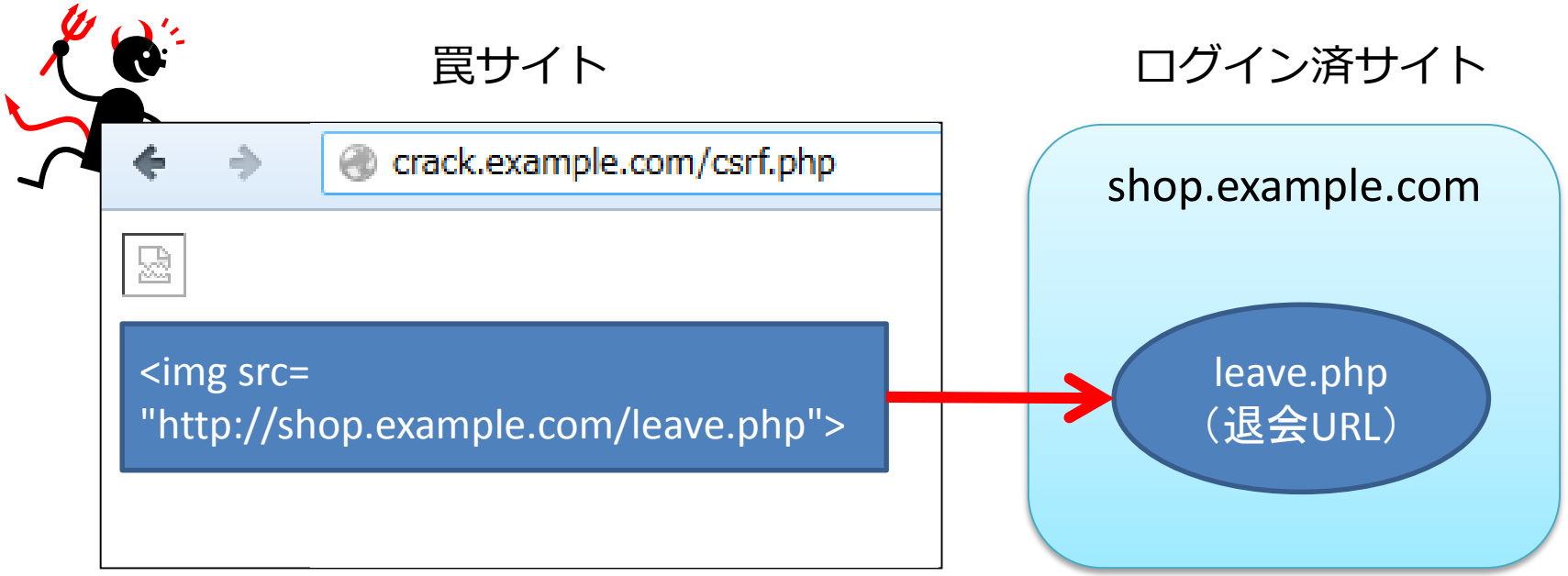
```
&   →   &amp;
<   →   &lt;
>   →   &gt;
'   →   &#39;
"   →   &quot;
```


CSRF

- CSRFとは？
  - ログイン済みユーザに対して、外部から不正操作を強要する攻撃
  - 主な被害
    - 投稿処理
    - 退会処理
    - 登録情報の変更
  - 正式名称
    - Cross Site Request Forgery

CSRF

- CSRFの仕組み
  - ログイン済みサイトに対して、img要素やJavaScriptによってリクエストを送信する



正規のユーザーとして退会処理が実行されてしまう！

CSRF

- 対策方法
  - パスワード入力によって本人の意思を確認する
  - 直前のページにトークンを埋め込み、外部サイトからのアクセスではなく正しい導線に基づいた要求であるかを判断する

- 間違った対策方法
  - リファラーをチェックする
    - リファラーは偽装が可能である

HTTPヘッダインジェクション

HTTPヘッダインジェクションとは？

- 主に改行文字でHTTPヘッダー行を追加し、任意のレスポンスを挿入する攻撃
  - 外部の入力を元にHTTPヘッダを組み立てているプログラム（パラメータで指定されたURLにリダイレクトするプログラムなど）はこの攻撃を受ける可能性がある
- 主な被害
  - 偽ページの表示
  - セッション固定攻撃（セッションの乗っ取り）
  - フィッシングサイトへのリダイレクト

HTTPヘッダインジェクション

- リダイレクタに改行文字と任意のHTTPヘッダをインジェクション

```
login.php?redirect=mypage.php%0DSet-Cookie: PHPSESSID=1234
```

例えば、古いバージョンのPHPでは、header関数が改行文字を適切にチェックしていないため、2つのHTTPヘッダがセットされてしまう

```
header('Location: '.$_GET['redirect']);
```

パラメータを埋め込み

```
header('Location: mypage.php%0DSet-Cookie: PHPSESSID=1234');
```

サーバー側からのHTTPレスポンス

```
Location:mypage.php
Set-Cookie:PHPSESSID=1234
```


HTTPヘッダインジェクション

- 対策方法
  - 特殊文字（改行コード）をエスケープする
  - HTTPヘッダインジェクション対策が行われている最新の開発言語を利用する

ディレクトリトラバーサル

- ディレクトリトラバーサルとは？
  - ../や/etc/passwdなど、パスを含むリクエストを送ることで、公開領域外のディレクトリにアクセスする攻撃
    - パラメータで指定されたファイルを開くプログラムなどはこの攻撃を受ける可能性がある

- 対策方法
  - 対象ファイルが公開領域にあるものかどうかをチェックする
  - ファイルにアクセス権を設定する
  - パスを含む文字列を受け取らない

APIの基礎知識

canvas要素

- JavaScriptでビットマップのグラフィックを描画する要素
  - 具体的な描画方法はLevel2の範囲
  - Level1ではキャンバスで描画できる図形の種類を知っておくこと
    - 四角形/線/円弧/画像

Web Storageの詳細

Web Storageとは？

- 『セッションストレージ』と『ローカルストレージ』が存在
  - ✓ セッションストレージはタブやウィンドウを閉じると消滅
- 保存できるデータは文字列のみ
  - ✓ 配列やオブジェクトを保存する場合はJSONという形式に変換します。

データの保存

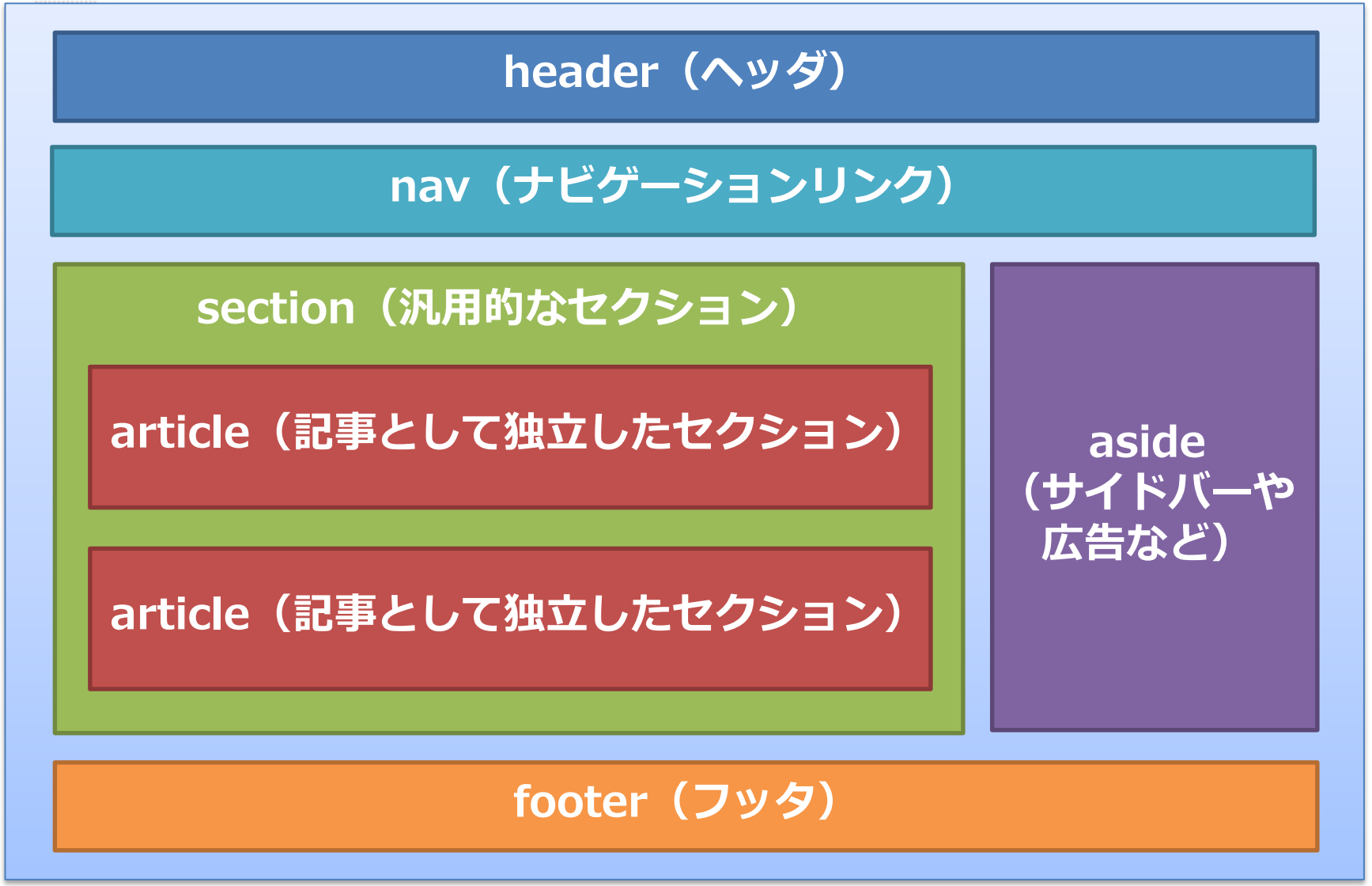
```
localStorage.setItem("キー", "保存する文字列");
```

データの取得

```
localStorage.getItem("キー");
```


要素

セクショニング要素


header（ヘッダ）
nav（ナビゲーションリンク）
section（汎用的なセクション）
article（記事として独立したセクション）
article（記事として独立したセクション）
aside
（サイドバーや
広告など）
footer（フッタ）

セクショニング要素の詳細

- article要素とsection要素
  - 記事として独立しているセクションは<article>、それ以外の汎用的なセクションは<section>
  - <article>、<section>要素はどちらが親要素となっても構わない
- header要素とfooter要素
  - <header>、<footer>は<article>、<section>の中に入れることもできる
  - <header>内には通常<h1>～<h6>を含むが、含まなくても構わない

video/audio/source要素

- 動画（video）、音声（audio）の再生
  - どちらの要素も使い方はほぼ同じ

```
<video src="video.mp4" controls autoplay loop></video>
```

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

  - クロスブラウザ対応するには<source>要素を子要素に使う

```
<audio controls autoplay loop>
  <!--ブラウザが対応しているファイルタイプに適合するファイルを使用-->
  <source src="audio.ogg" type="audio/ogg" media="all">
  <source src="audio.mp3" type="audio/mp3" media="all">
  <p>ご利用中のブラウザでは再生できません</p>
</audio>
```

再生範囲の指定（video/audio）

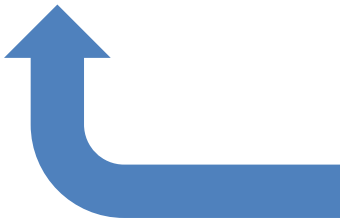
- 動画、音声の再生範囲を指定する
  - ファイル名の後ろに「#t=開始時間,終了時間」を指定すると、再生範囲を指定することができる

```
<video controls>
    <source src="video.wmv#t=20,40">
    <source src="video.mp4#t=20,40">
    <p>ご利用中のブラウザでは再生できません</p>
</video>
```

track要素

- メディアファイルに字幕をつける
  - WebVTT形式で作成したテキストトラックを、video要素などに埋め込む

```
<video src="video.mp4">
  <track src="ja.vtt" srclang="ja" label="日本語" kind="subtitles" default>
  <track src="en.vtt" srclang="en" label="英語" kind="subtitles">
</video>
```



```
WEBVTT

00:00:01.000 --> 00:00:05.000
こんにちは

00:00:06.000 --> 00:00:11.000
はじめまして
```

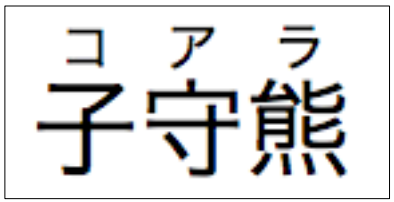

ruby要素

ルビを付与する

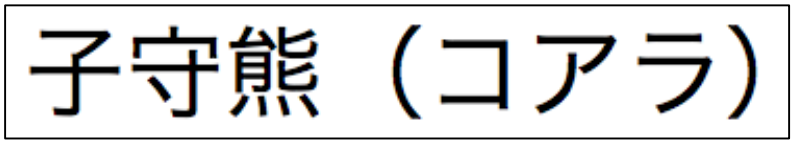
- <ruby>　対象テキストをマークアップ
- <rt>　ルビテキストを指定
- <rp>　ルビ未対応ブラウザでのみ表示

```
<ruby>子守熊<rp>(</rp><rt>コアラ</rt><rp>)</rp></ruby>
```

ルビ対応ブラウザ



ルビ未対応ブラウザ



bdi要素

- 向きが異なるテキストを埋め込む
  - アラビア文字などの、右から左に記述する自然言語をマークアップすることによって前後のテキストが入れ替わって配置される現象を回避する

```
<ul>
    <li>User Name: <bdi>james</bdi> 2020/5/28</li>
    <li>User Name: <bdi>steve</bdi> 2020/6/19</li>
    <li>User Name: <bdi>إيان</bdi> 2020/7/20</li>
</ul>
```


wbr要素

- テキストが改行されても良い位置を指定する
  - 一般的なブラウザはスペースを含まないアルファベットや記号の並びは改行せずに表示する
  - ブラウザの横幅が狭く文章を表示しきれない場合に、<wbr>が挿入されている箇所を改行する

```
section,nav,article,aside,header,footer,main,<wbr>figure,figcaption,video,audio,sourc
e,canvas,mark,time,data,ruby,rt,rp,rb,bdi,progress,meter,output,datalist,keygen
```


progress/meter要素

<progress>

- タスクの進捗状況を表す

```
<progress value="75" max="100">100%中75%まで完了</progress>
```

<meter>

- 範囲内の数、量、割合などを表す

```
<meter value="75" max="100" min="0">100人中75人が回答</meter>
```

- 見た目はどちらもほぼ同じ 

time/data要素

<time>

- 日付や時刻を表す
- datetime属性（省略時は内容テキスト）をコンピュータが認識可能な形式で指定する

```
<time datetime="2020-05-15 19:00">May 15</time>
```

<data>

- 日付以外のデータを表す
- value属性をコンピュータが認識可能な形式で指定する

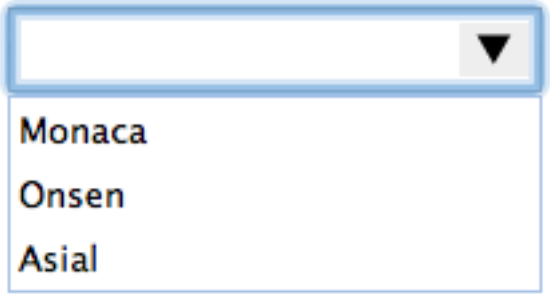
```
<data value="30">三十歳</data>
```


datalist要素

- フォームの入力候補を定義する
  - 定義した入力候補はテキストボックスなどに付与することができる

```
<datalist id="keywords">
    <option value="Monaca">
    <option value="Onsen">
    <option value="Asial">
</datalist>

<input type="text" list="keywords">
```



独自データ属性

独自データ属性

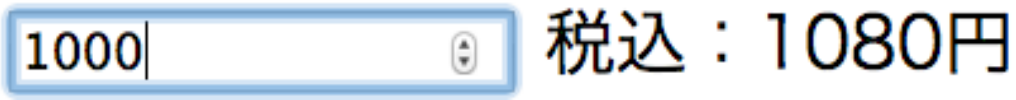
- スクリプトで利用する値を要素に保持するために、data-で始まる属性を自由に定義することができる

```
<ul>
    <li data-id="0001">商品A</li>
    <li data-id="0002">商品B</li>
    <li data-id="0003">商品C</li>
</ul>
```


output要素

- スクリプトによる計算結果などを表示する
  - ユーザーに対して表示することのみを目的とする（フォーム送信されない）

```
<form oninput="result.value = parseInt(price.value * 1.08)">
    <input type="number" name="price" value="0">
    税込:<output name="result">0</output>円
</form>
```



バリデーション属性

■ バリデーション属性

- <input>要素に付与すると、 submit時にチェックを行ってくれる
  - required　必須
  - pattern　正規表現
  - min　最小値
  - max　最大値
  - maxlength　最長文字数
- title属性値を指定するとエラーメッセージを拡張することができる

```
名前:<input type="text" required>
郵便番号:<input type="text" pattern="^[0-9]{3}-[0-9]{4}$" title="例:130-0011">
年齢:<input type="number" min="18" max="99">
ID:<input type="text" maxlength="6">
```

input type属性値

input type属性値

- urlやemailを指定した場合、フォーム送信時にURL、メールアドレスの形式として正しいかバリデーションが行われる
- rangeを指定するとレンジバーが、colorを指定するとカラーパレットが表示されるなど、UIの拡張が行われる

```
URL：<input type="url">
メールアドレス：<input type="email">
数値：<input type="number">
検索キーワード：<input type="search">
電話番号：<input type="tel">
日付：<input type="date">
範囲：<input type="range">
色：<input type="color">
```


CSS3

CSSの作成

CSSの記述方法

① <style>要素内に記述

② <link>要素でCSSファイルを読み込む

- CSSファイル内で@importを使うと別のCSSファイルを読み込むことが出来る

index.css（index.htmlのスタイルシート）

```
@import url("common.css");  /* 共通のスタイルシート */
```

③ 要素のstyle属性で指定する

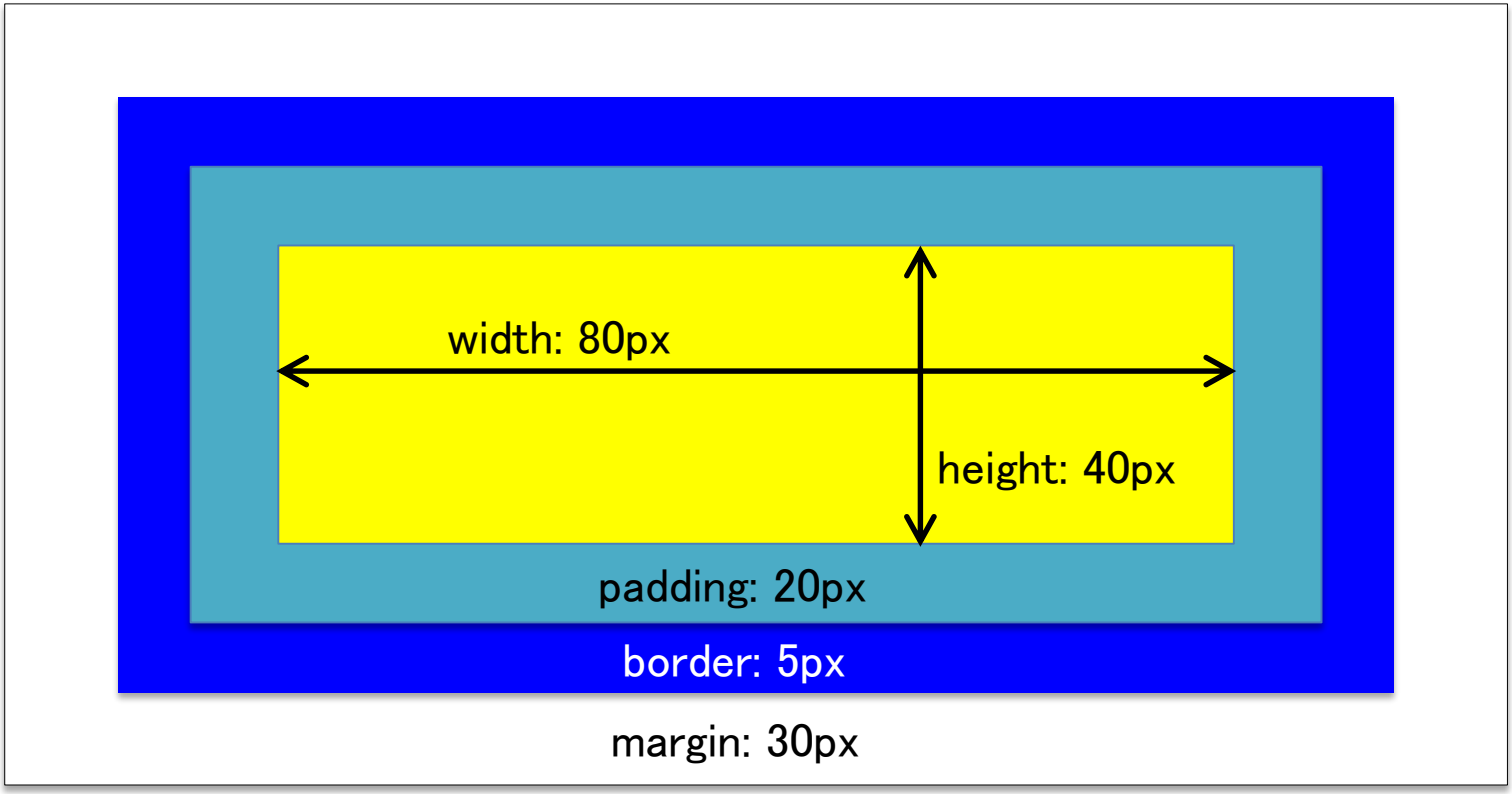
セレクタ

- セレクタ、疑似要素、疑似クラスの種類
  - idセレクタ/要素型セレクタ/クラスセレクタ/属性セレクタ
  - 子セレクタ/子孫セレクタ/隣接セレクタ/間接セレクタ
  - 疑似クラス（:first-child/:active/:link/:hover/:target/:notなど）
  - 疑似要素（::before/::after/::first-lineなど）

ボックスのサイズ

- ボックスは「コンテンツ領域（width,height）」「内側の余白（padding）」「枠線（border）」「外側の余白（margin）」によって構成される

枠線までを含んだボックス幅 ：（　　）px



ボックスのサイズ

CSS3での変更点

- box-sizingプロパティによって、width、heightの指定値をborderまで含めた値にすることが可能
  - box-sizing：content-box　　従来通り
  - box-sizing：border-box　　borderまでを幅と高さに含める

margin/paddingプロパティの指定方法

margin/paddingプロパティの値

- 値が１つ：上下左右すべて同じ値

```
margin: 10px;
```

- 値が２つ：上下 と 左右

```
margin: 10px 20px;  /* 上下10px, 左右20px */
```

- 値が３つ：上　左右　下

```
margin: 10px 20px 30px;  /* 上10px, 左右20px, 下30px */
```

- 値が４つ：上、右、下、左（時計回りと覚える）

```
margin: 10px 20px 30px 40px;  /* 上10px, 右20px, 下30px, 左40px */
```


Webフォント

Webフォントとは

- Web上に配置されたフォントを読み込む技術。ユーザーの環境に依存しないため、どのような環境でも同じフォントを表示することができる
- Webフォントの利用手順

① フォントをサーバーにアップロード

② フォントを読み込み、フォントファミリー名を定義

```
@font-face {
    font-family: myfont;
    src: url(../font/myfont.woff);
}
```

③ 定義したフォントファミリーを利用

```
p { font-family: myfont; }
```

