

# HTMLとCSS:

## 応用知識まで分かる基礎講座

このプレゼンテーションでは、初心者から経験者までを対象に、HTMLとCSSの基本から応用知識まで、実践的で分かりやすい説明を行います。



# HTMLとCSSの基礎知識

1

## HTMLとは

HTMLは、ウェブページを作成するためのマークアップ言語です。

3

## マークアップ言語とスタイルシート言語の違い

マークアップ言語は、文書の構造を定義し、スタイルシート言語は、それらの文書のデザインを整えます。

2

## CSSとは

CSSは、ウェブページのスタイルを定義するためのスタイルシート言語です。

4

## ウェブページの基本構造

HTML (HyperText Markup Language) を使用して定義されます。

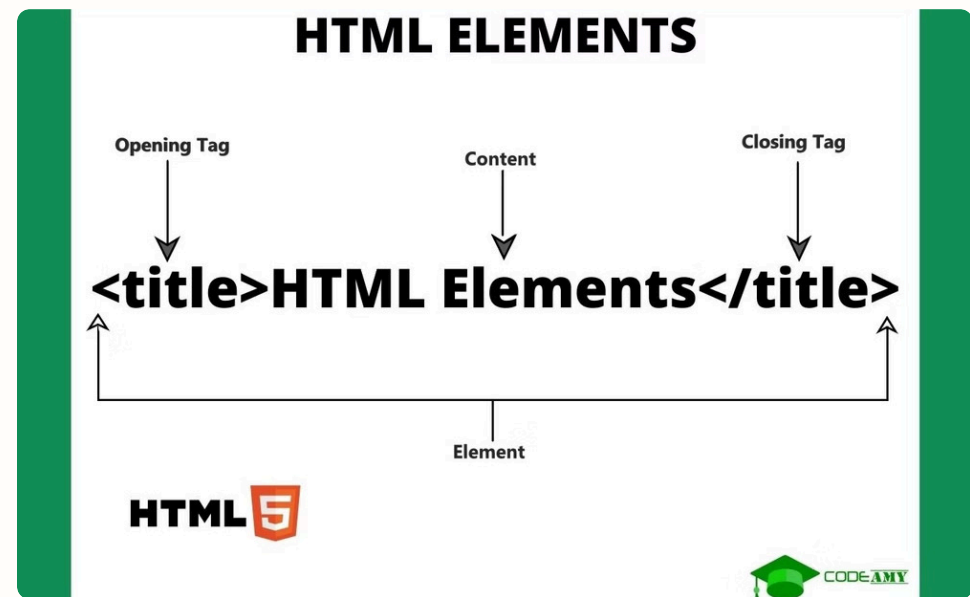
# HTMLの概要

Tag	Description
<html> ... </html>	Declares the Web page to be written in HTML
<head> ... </head>	Delimits the page's head
<title> ... </title>	Defines the title (not displayed on the page)
<body> ... </body>	Delimits the page's body
<h n> ... </h n>	Delimits a level n heading
<b> ... </b>	Set ... in boldface
<i> ... </i>	Set ... in italics
<center> ... </center>	Center ... on the page horizontally
<ul> ... </ul>	Brackets an unordered (bulleted) list
<ol> ... </ol>	Brackets a numbered list
<li> ... </li>	Brackets an item in an ordered or numbered list
 	Forces a line break here
<p>	Starts a paragraph
<hr>	Inserts a horizontal rule
	Displays an image here
<a href="..."> ... </a>	Defines a hyperlink

## タグ

タグは、HTML文書内の要素を定義します。

ex:<h1>,<p>,<h4> etc...



## 要素

要素は、開始タグと終了タグの間に存在するHTMLコードです。

ex:<h1>テキスト</h1> etc...

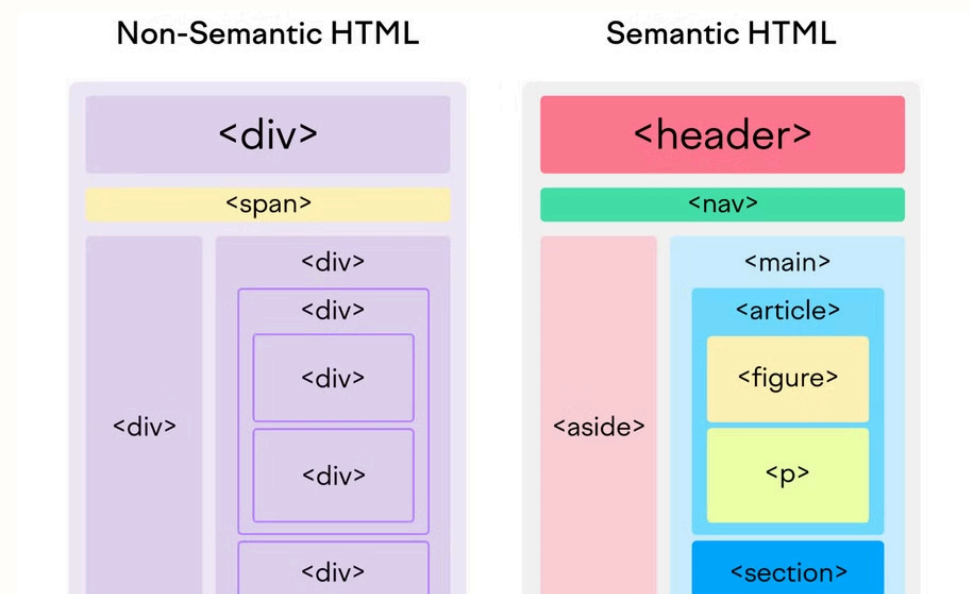


## Internal DOCTYPE declaration

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE Note [
  <!ELEMENT Note (To, From, Course*)>
  ...
]>
<Note>
  ...
</Note>
```

## ドキュメントタイプ宣言

ドキュメントタイプ宣言は、HTML文書がどのバージョンのHTMLに準拠しているかを示します。



## セマンティックHTML

セマンティックHTMLは、ウェブページに意味を付けるためのHTMLタグを使用することです。

# CSSの概要

## CSSセレクタ

CSSセレクタは、特定のHTML要素にスタイルを適用するために使用します。

## カスケーディング

カスケーディングは、異なるスタイルシートが同じHTML要素に適用された場合にスタイルが競合する方法を解決するルールです。

## スタイルの継承

スタイルの継承は、親のHTML要素に定義されたスタイルが子要素にも適用される方法です。

## ボックスモデル

ボックスモデルは、HTML要素がどのようにスタイル化されるかを定義します。



# CSSのセレクタとプロパティ

```
2519 </div>
2520 </div></div>
2521 </div>
2522 <div class='clearboth'></div>
2523 <div class='blog-name container'>
2524 <div class='container section' id='header' name='ヘッダー'>
  <div class='widget Header' data-version='2' id='Header1'>
2525 <div class='header-widget'>
2526 <div>
2527 <h2 blog-title='2'><a href='https://www.nagahitoyuki.com/'>
  些細な日常</a></h2>
2528 </div>
2529 <p>
2530 いつも興味や関心を取り上げよう<br>自分らしさこそ自然のままに
2531 </p>
2532 </div>
2533 </div></div>
2534 </div>
2535 </div>
2536 </header>
2537 </div>
```

## タグの選択

CSSでHTMLのタグの名前を使用して、特定のHTMLタグ全体にスタイルを適用できます。HTMLの選択したタグをCSSではセレクターと呼ぶ。



## IDセレクタ

IDセレクタは、HTML要素のID属性を使用して、特定のHTML要素にスタイルを適用できます。

CSSでは「#id名」から始める!!



## クラスセレクタ

クラスセレクタは、HTML要素のクラス属性を使用して、特定のHTML要素にスタイルを適用できます。CSSでは「.クラス名」から始める!!



## 背景プロパティ

背景プロパティは、ウェブページの背景を設定します。

文字の色、大きさ、太さetc...などの変更処理をプロパティと呼ぶ。



# 基礎固め

1

## HTML (エイチティーエムエル、HyperText Markup Language)

Webページを構成するために開発されたコンピュータ言語



株式会社ロジックスサービス

「HTML」ってなに？初心者向けに解説！ | 株式会社ロジッ...

HTMLとは、ホームページをブラウザに表示するためのプログラミング言語。HTMLは、WEBページを作るための基礎知識です。

2

## ショートカットキー mac(win)

コピー commnd (control)+ C

貼り付け commnd (control)+ V

前に戻る commnd(control) + Z

コメント commnd(control) + /

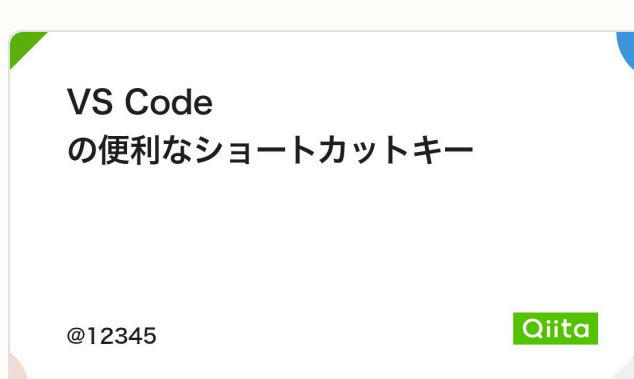
デベロッパーツール commnd(control) + option(alt)+I

複製 Shift+ option(alt)+下矢印キー

全選択 commnd(control) + A

VSコードインデントの整え Shift+ option(alt)+F

詳しくは



Qिता

VS Code の便利なショートカットキー - Qिता

Visual Studio Codeを自分が使用する際に便利なショートカットキーを、忘れないようにまとめておく。Windows版。V1.57で再確認済...

3

## 画像の挿入

imgタグを使用して画像をウェブページに挿入し、適切な属性を設定して画像を表示します。

4

## フォームの作成

formタグを使用してウェブフォームを作成し、inputタグを使用して異なるタイプのフォームの要素を作成します。

5

## アイコン画像の貼り付け方



新宿のWeb制作会社Btiesが教える！ホームページ制作のすべて

【2018年版】ホームページのタブ画像(favicon/touch-icon...

ホームページを見ていると、タブにアイコン画像が表示されていますよね。今回は、このアイコン画像の作り方と設定方法を2018年版と...

6

## HTML色々なタグのご紹介

下の表を参考に学んでください。

タグ	説明
<!-...-->	コメントを定義します
<!DOCTYPE>	ドキュメントの種類を定義します
<a>	ハイパーリンクを定義します
<abbr>	略語または頭字語を定義します
<acronym>	HTML5ではサポートされていません。代わりに <abbr> を使用します。頭字語を定義します
<address>	ドキュメントの著者/所有者の連絡先情報を定義します
<applet>	HTML5ではサポートされていません。代わりに <embed> または <object> を使用します。埋め込みアプレットを定義します
<area>	画像マップ内の領域を定義します
<article>	記事を定義します
<aside>	ページのコンテンツ以外のコンテンツを定義します
<audio>	埋め込み音声コンテンツを定義します
<b>	太字のテキストを定義します
<base>	ドキュメント内の相対URLのベースURL/ターゲットを指定します
<basefont>	HTML5ではサポートされていません。代わりにCSSを使用します。テキストのデフォルトの色、サイズ、フォントを指定します
<bd>	外部のテキストと異なる方向でフォーマットされる可能性のある一部のテキストを隔離します
<bdo>	現在のテキストの方向を上書きします
<big>	HTML5ではサポートされていません。代わりにCSSを使用します。大きなテキストを定義します
<blockquote>	別のソースから引用されたセクションを定義します
<body>	ドキュメントの本文を定義します
 	1つの改行を定義します
<button>	クリック可能なボタンを定義します
<canvas>	スクリプトを介して（通常JavaScript経由で）グラフィックスを描画するために使用されます
<caption>	テーブルのキャプションを定義します
<center>	HTML5ではサポートされていません。代わりにCSSを使用します。テキストを中央寄せに定義します
<cite>	作品のタイトルを定義します
<code>	コンピューターコードの一部を定義します
<col>	テーブル内の各列のプロパティを指定します
<colgroup>	テーブル内の1つ以上の列をフォーマットするためのグループを指定します
<data>	指定されたコンテンツの機械読み取り可能な翻訳を追加します
<datalist>	入力コントロール用の事前定義オプションのリストを指定します
<dd>	説明リスト内の用語の説明/値を定義します
<del>	ドキュメントから削除されたテキストを定義します
<details>	ユーザーが表示または非表示にできる追加の詳細情報を定義します
<dfn>	コンテンツ内で定義される用語を指定します
<dialog>	ダイアログボックスまたはウィンドウを定義します
<dir>	HTML5ではサポート
<hr>	直線的な線を書き出す
<strong>	太字で表示される
<em>	テキストが強調される
<mark>	囲ったものをマーカーを引ける

# テーブル作成方法1

## ステップ1:

`<table>` 要素の追加 まず、`<table>` 要素をHTML文書内に追加します この要素はテーブル全体を定義します。

```
<table> <!-- テーブルの内容はここに追加します --> </table>
```

## ステップ2:

`<tr>` 要素で行を追加. テーブル内の各行を作成するには、`<tr>` (table row) 要素を使用します。 `<tr>` 要素はテーブル内の行を表します。

```
<table>
```

```
<tr> <!-- 1つ目の行の内容をここに追加します --> </tr>
```

```
<tr> <!-- 2つ目の行の内容をここに追加します --> </tr>
```

```
</table>
```

# テーブル作成方法2

## ステップ3:

`<th>` または `<td>` 要素でセルを追加 各行内でセルを追加するには、`<th>` (table header) 要素または`<td>` (table data) 要素を使用します。 `<th>` 要素は見出しセルを定義し、`<td>` 要素はデータセルを定義します。

ボーダーやセル間の間隔を指定するために `border` 属性を使用できます。

```
<table border="1">
```

```
<tr>
```

```
<th>見出しセル1</th>
```

```
<th>見出しセル2</th>
```

```
</tr>
```

```
<tr>
```

```
<td>データセル1</td>
```

```
<td>データセル2</td>
```

```
</tr>
```

```
</table>
```

# テーブル作成方法3

おまけ: `colspan` 属性は、横方向にセルを結合するために使用されます。`rowspan` 属性は、縦方向にセルを結合するために使用されます。

具体的には、1つのセルが複数の列を横断することを意味します。この属性は `<th>` または `<td>` 要素に適用できます。

```
<td rowspan="2">データセル1と2</td>
```

```
<th colspan="2">見出しセル2と3</th>
```

`<caption>` タグはHTMLのテーブル内において、テーブルに対するキャプションやタイトルを定義するためのタグです。テーブルの最初に配置され、テーブル自体の説明やタイトルを提供します。

```
<table border="1">
```

```
<caption>このテーブルは商品の在庫情報を示しています</caption>
```

```
<tr> <th>商品名</th> <th>価格</th> <th>在庫数</th> </tr> <tr> <td>商品A</td> <td>1000円
```

```
</td> <td>50個</td> </tr> <tr> <td>商品B</td> <td>800円</td> <td>30個</td> </tr>
```

```
</table>
```



# iframeについて1

## Googlemap 表示方法:

- 1.Google map にログインする
- 2.埋め込みたい位置を検索
- 3.共有をタップして「地図を埋め込み」をタップする
- 4.「埋め込みコード HTMLをコピー」をタップする
- 5.HTML編集画面にコピーした埋め込みコードを貼り付ける

## Instagram 表示方法:

- 1.Web版Instagramにログインする
- 2.埋め込みたい投稿を選ぶ
- 3.投稿画面右上の「…」をタップして「埋め込み」をタップする
- 4.「埋め込みコードをコピー」をタップする
- 5.HTML編集画面にコピーした埋め込みコードを貼り付ける

# iframeについて2

## YouTube動画の埋め込みコードの取得:

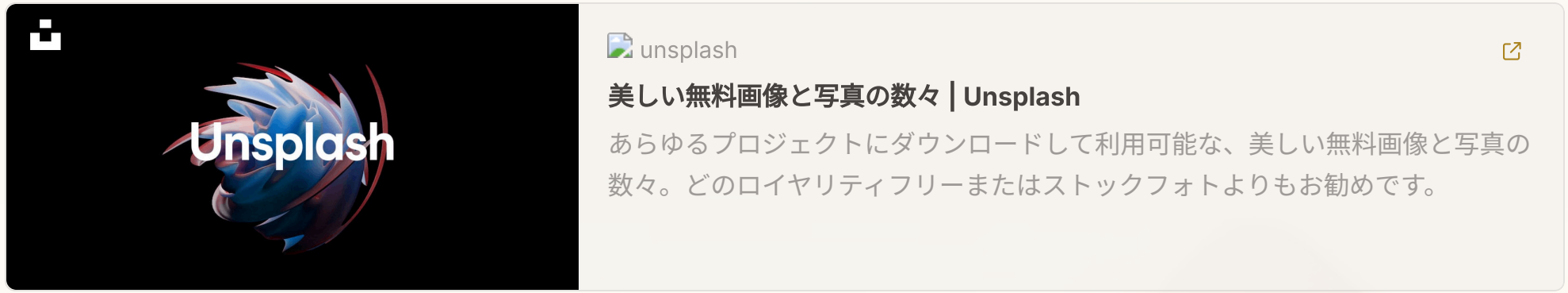
- 1.YouTubeのウェブサイトですぐ埋め込みたい動画を開きます。
- 2.動画プレーヤーの下にある「共有」ボタンをクリックします。
- 3.「埋め込む」オプションを選択します。
- 4.サイズやオプションを選択し、埋め込みコードが生成されます。必要な設定を選択してコードをコピーします。

```
<iframe width="560" height="315" src="https://www.youtube.com/embed/VIDEO_ID" frameborder="0" allowfullscreen></iframe>
```

- `<iframe>` 要素内の `width` および `height` 属性で動画プレーヤーの幅と高さを設定できます。
- `frameborder="0"` を指定して、プレーヤーの周りの枠を非表示にできます。
- `allowfullscreen` 属性を指定すると、動画プレーヤーで全画面表示を許可します。



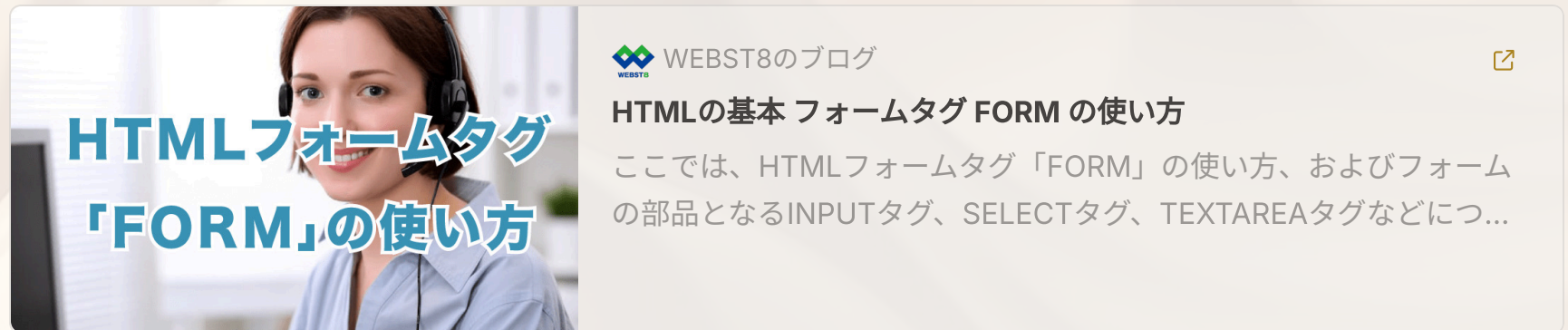
# 画像取得サイトの紹介



1

## HTMLフォームの作成において覚えるべきタグや考え方

主にお問い合わせフォームで使用します。



2

## 基本的なフォーム構造

- `<form>`タグは、入力フォーム全体を囲むために使用されます。
- `action`属性は、フォームが送信されたときにデータを処理するサーバーのURLを指定します。
- `method`属性は、フォームデータを送信するためのHTTPメソッド（通常はGETまたはPOST）を定義します。

詳しくは <https://qiita.com/12345/items/64f4372fbca041e949d0>

3

## テキスト入力:

- `<input type="text">`は、ユーザーからの一行のテキスト入力を受け取ります。
- `<textarea>`は、より長いテキストや改行を含むテキスト入力に使われます。colsとrowsで横幅の調整できます。「cols属性は横幅を半角文字数で設定します。rows属性は高さを半角文字数で設定します。」

4

## ラベルの使用:

- `<label>`タグは、フォームコントロールにテキストラベルを付けるために使用されます。
- `for`属性は、ラベルが関連付けられる入力要素のIDを指定します。これにより、アクセシビリティが向上し、ユーザビリティが改善されます。

5

## 入力フィールドのプレースホルダ: etc...

**Placeholder属性** (プレースホルダー) を設定する事で入力例や説明を表示することができます。

文字を打ち始めると内容は非表示になる。

**Required属性**(リクワイアード属性)を設定する事で入力必要項目とすることができます。

**Readonly属性** (リードオンリー) を設定する事でフォームでの入力情報として表示しつつ書き換え禁止にすることができます。

**Disabled属性**(ディーセブル)を設定する事で入力したものを無効にできる。

6

## ボタン:

- `<button>`タグは、クリック可能なボタンを作成します。
- `<input type="submit">`は、フォームを送信するためのボタンを生成します。

7

## 事前に値をセットする:

- `value`属性は、テキストボックスやラジオボタンなどの入力要素に初期値を設定します。
- `checked`属性と`selected`属性は、それぞれチェックボックスやラジオボタン、オプションが選択されている状態を示すために使用されます。

8

## 特化した入力タイプ:

- `<input type="password">`は、パスワード入力用で、入力された文字を隠します。
- `<input type="number">`は、数値の入力を受け付けます。
- `<input type="date">`は、日付入力用のフィールドを提供します。

9

## おまけ:

`<fieldset>`: これはフォーム内で複数の関連するフォーム要素を囲むために使用されます。例えば、複数の選択肢からいくつかを選択するチェックボックス群をグループ化するときなどに便利です。

`<legend>`: `<fieldset>`要素の最初の子として配置されるべき要素で、その`<fieldset>`でグループ化されたフォーム要素のグループ名を提供します。この例では、「対象の言語」というテキストがグループの名前として機能します。



# CSS基本スタイリング編+α知識

1

## CSS (カスケーディングスタイルシート、Cascading Style Sheets)

Webページのデザインを整える言語



株式会社SAMURAI

CSSとは? できることや書き方を初心者向けにわかりやすく...

この記事では「CSSとは? できることや書き方を初心者向けにわかりやすく解説」といった内容について、誰でも理解できるように解説...

2

## text-align:centerとmargin:0 autoの違い



Web鍛 (うえぶたん)

どっちが正解? 「margin: 0 auto;」と「text-align: cente...

みなさま初めまして! 実習生のNです。職業訓練校に通い、Webデザインについて学び始めてもうすぐ4ヶ月。私と同じように、自分が...

簡単に言うと

text-align:centerはテキストを真ん中に...

margin:0 autoは要素を真ん中に...

3

## font-family:

HTML文書内のテキストに使用するフォント、またはフォントファミリーを指定するためにCSSで使用されます。このプロパティを使うことで、テキストの見た目を変更し、読みやすさやデザインの向上を図ることができます。

fromkato.com

font-family : フォントの種類 - CSSプロパティ - Web開発 - fromkato.com

「フォントの種類」を設定するCSSプロパティ。

- **特定のフォント:** "Helvetica Neue", "Times New Roman" など、具体的なフォント名を指定します。フォント名が複数の単語で構成されている場合は、引用符で囲む必要があります。
- **総称ファミリー:** serif, sans-serif, monospace, cursive, fantasy など、フォントのスタイルを一般的に指定します。これは、特定のフォントが利用できなかった場合のフォールバックとして機能します。

## フォントの選択

フォントを選択する際には、読みやすさ、デザインの一貫性、ウェブページの目的、そしてブランディングを考慮する必要があります。また、全てのユーザーが同じフォントをインストールしているわけではないため、フォントスタックには複数のフォントを指定し、どの環境でも適切なフォールバックが行われるようにすることが重要です。

## フォントの可用性

ウェブフォントサービス (例: Google Fonts) を使用されることが多いです。

Google Fonts

Google Fonts

Browse Fonts - Google Fonts

Making the web more beautiful, fast, and open through great typography

4

## list-style-type:

None リストマーカーを非表示

Disc 塗りつぶされた円のマーカー

Circle 白抜きの円のマーカー

Square 塗りつぶされた四角のマーカー

Decimal 10進数 (1, 2, 3...) のマーカー

decimal-leading-zero ゼロから始まる10進数 (01, 02, 03...) のマーカー

5

## text-decoration:

```
<div style="text-decoration:none">線無し</div>
<div style="text-decoration:underline">下線</div>
<div style="text-decoration:overline">上線</div>
<div style="text-decoration:line-through">打ち消し線</div>
<div style="text-decoration:underline solid">下線・実線</div>
<div style="text-decoration:underline double">下線・二重線</div>
<div style="text-decoration:underline dotted">下線・点線</div>
<div style="text-decoration:underline dashed">下線・破線</div>
<div style="text-decoration:underline wavy">下線・波線</div>
<div style="text-decoration:underline red">下線・赤</div>
<div style="text-decoration:blink">ブリンク</div>
```

線無し  
下線  
上線  
打ち消し線  
下線・実線  
下線・二重線  
下線・点線  
下線・破線  
下線・波線  
下線・赤  
ブリンク



# CSS応用スタイリング編+α知識

1

## インライン要素、ブロック要素、インラインブロック要素:

```
<!DOCTYPE html>
<html> <head>
<style>
.block { display: block; background-color: lightblue; }
.inline { display: inline; background-color: lightgreen; }
.inline-block { display: inline-block; background-color: lightcoral; }
</style>
</head> <body>
<div class="block">ブロック要素</div>
<span class="inline">インライン要素</span>
<span class="inline-block">インラインブロック要素</span>
</body> </html>
```

こちらにコードをVSコードで見ると違いがわかると思います。

2

## ブロック要素:

- **定義** ブロック要素は、新しい行から始まり、横幅いっぱい広がります。
- これは、要素が独自の「ブロック」を形成すると考えることができます。
- **特徴**
  - 新しい行から始まる。
  - 横幅はデフォルトで親要素の幅に合わせて広がる。
  - 高さは内容に合わせて自動調整される。
  - `width` や `height` でサイズを指定できる。
- **例**: `<div>`, `<p>`, `<h1>` など。

3

## インライン要素:

- **定義** インライン要素は、テキストの流れの中にそのまま挿入されます。テキストと同じ行に表示され、前後の要素との間に改行は入りません。
- **特徴**
  - 同じ行上に表示され、他のインライン要素と並びます。
  - 幅と高さを直接指定できない。
  - テキストの一部のように扱われる。
- **例**: `<span>`, `<a>`, `<strong>` など。

4

## インラインブロック要素:

- **定義** インラインブロック要素は、インライン要素のように同じ行に表示されますが、ブロック要素のように幅と高さを指定できます。
- **特徴**
  - インライン要素のように、他の要素と同じ行に配置されます。
  - ブロック要素のように、幅と高さの指定が可能。
  - 内部ではブロック要素のように振る舞うが、外部ではインライン要素のように扱われる。
- **例**: 通常はCSSで `display: inline-block;` を設定した要素。

5

## positionプロパティ:

- **目的**: `position` プロパティは、ウェブページ上の要素の配置を決定します。
- **主な値**
  - `static`
  - `relative`
  - `absolute`
  - `fixed`
  - `sticky`

6

## static:

- **説明**: これは`position`のデフォルト値です。要素は通常の文書の流れに従って配置されます。
- **特徴**
  - `top`, `right`, `bottom`, `left`, `z-index` プロパティは影響しません。
  - 要素は自然なページの流れに従って配置されます。

7

## relative:

- **説明**: `relative`は要素を通常的位置から相対的に移動させます。
- **特徴**
  - 要素の元の位置は文書の流れに影響を与えません。
  - `top`, `right`, `bottom`, `left` プロパティで移動距離を指定できます。
  - 他の要素には影響を与えません。

8

## absolute:

- **説明**: `absolute`は要素を文書の流れから完全に切り除き、最も近い位置指定された祖先要素に対して相対的に配置します。
- **特徴**
  - 要素は文書の流れから取り除かれ、他の要素の配置に影響しません。
  - 位置指定されていない場合は、`html`要素に対して相対的に配置されます。
  - `top`, `right`, `bottom`, `left`, `z-index` で位置を制御します。

9

## fixed:

- **説明**: `fixed`は要素をビューポートに対して固定します。スクロールしても位置は変わりません。
- **特徴**
  - ページスクロールに関係なく、常に同じ位置に表示されます。
  - `top`, `right`, `bottom`, `left`, `z-index` で位置を制御します。

10

## sticky:

- **説明**: `sticky`は特定のスクロール位置で固定されるようになるまで、要素を`relative`のように振る舞わせ、その後`fixed`のように動作します。
- **特徴**
  - スクロール位置に基づいて、`relative`と`fixed`の間で切り替わります。
  - `top`, `right`, `bottom`, `left` を使って、固定位置の開始点を設定できます。

11

## 実際の確認コード:

```
<!DOCTYPE html>
<html>
<head>
<style>
.static { position: static; background-color: lightblue; }
.relative { position: relative; top: 20px; left: 20px; background-color: lightgreen; }
.absolute { position: absolute; top: 40px; left: 40px; background-color: lightpink; }
.fixed { position: fixed; top: 60px; left: 60px; background-color: lightgrey; }
.sticky { position: sticky; top: 0; background-color: lightcoral; }
</style>
</head>
<body>
<div class="static">Static</div>
<div class="relative">Relative</div>
<div class="absolute">Absolute</div>
<div class="fixed">Fixed</div>
<div style="height: 500px;">スクロールしてStickyを確認</div>
<div class="sticky">Sticky</div>
<div style="height: 1000px;">スペース</div>
</body>
</html>
```

12

## ビデオの貼り付け方:

HTMLの`<video>`タグは、ウェブページに動画を埋め込むために使用される要素です。

このタグに関する詳細な説明を以下に示します。

ex:

```
<video src="movie.mp4" controls> お使いのブラウザはvideoタグをサポートしていません。 </video>
```

ex:

```
<video src="/videos/matsuri.mp4" width="500" height="300" autoplay muted loop></video>
```

## 主要な属性

1. **src**: 動画ファイルのURLを指定します。相対パスまたは絶対パスを使用できます。
2. **controls**: この属性が存在すると、再生、一時停止、音量調整などのコントロールが表示されます。
3. **autoplay**: この属性が指定されていると、ページ読み込み時に動画が自動的に再生されます（一部のブラウザでは、ユーザーの操作がないと自動再生されない場合があります）。
4. **loop**: この属性が指定されていると、動画が終わると自動的に再び開始します。
5. **muted**: この属性が指定されていると、動画の音声がミュートされます。`autoplay`と組み合わせる際によく使用されます。
6. **preload**: この属性は、ページ読み込み時に動画ファイルをどの程度事前に読み込むかを指定します。値として`none`（読み込まない）、`metadata`（メタデータのみ読み込む）、`auto`（全て読み込む）があります。
7. **width / height**: 動画の表示サイズを指定します。

13

## サポートされるビデオ形式:

HTML5では、主に以下のビデオ形式がサポートされていますが、

ブラウザによって対応状況が異なります。

- **MP4 (MPEG-4 Part 14) with H.264 video codec and AAC audio codec**
- **WebM with VP8 or VP9 video codec and Vorbis or Opus audio codec**
- **Ogg with Theora video codec and Vorbis audio codec**

14

## ブラウザによって対応状況を整える:

```
<video controls>
<source src="movie.mp4" type="video/mp4">
<source src="movie.webm" type="video/webm"> お使いのブラウザはvideoタグをサポートしていません。
</video>
```

\*すべてのブラウザがすべてのビデオ形式をサポートしているわけではありませんので、複数のフォーマットを提供することが重要！

\*モバイルブラウザや一部のデスクトップブラウザでは、音声付きでの自動再生が制限されていることがあります。（この理由は僕は駅などパブリックな場で急な音が流れる動画を防ぐためブラウザにより制限されていてgoogle Chromなどもこのような理由だと思います）

\*大きな動画ファイルはページの読み込み時間に影響を与え、ユーザーのデータ使用量に影響を与えることがあるので程々に笑



著作権フリーな動画の素材が取得できるサイトをご紹介します。ここから取得して下さい

videoAC



videoAC



### 無料の動画素材サイト | 動画AC

動画ACは無料で高品質な動画素材を提供しています。フリー動画素材は、ウェブサイトやSNS広告などで使えるHD・4Kをご用意しております。個人、商用...



# CSSセレクター編+α知識

CSSはHTMLに記述された指定の範囲または要素に対して装飾を施します。そして、**CSSによるデザイン指定をどのHTML要素に適用させるかを指定するのに用いられるのが「CSSセレクタ」**です。

1

## \* (すべての要素を指定):

「\* (アスタリスク)」と記述することで、すべての要素に装飾が適用されます。

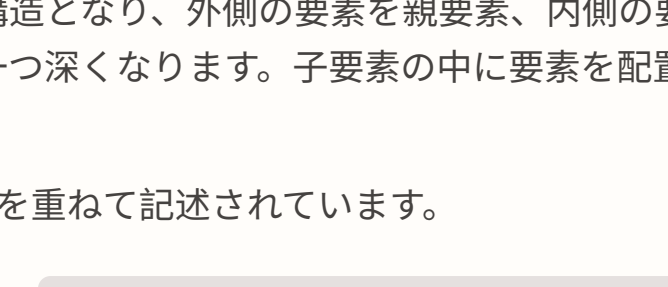
```
* {
  margin:0;
}
```

2

## HTMLの構造

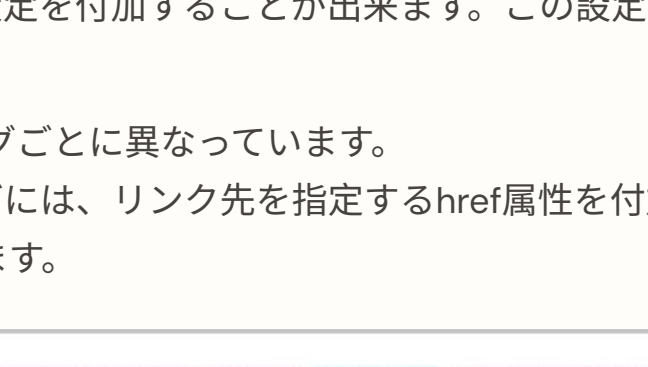
CSSを理解する前にまずHTMLがどのような仕組みで記述されているかを理解しておく必要があります。

HTMLはタグと呼ばれる記号で構成されています。タグは一般的に**開始タグ**と**終了タグ**があり、記述をタグで囲むことにより文章構造を定義しています。この、**タグで囲われた塊のことを「要素」と**いいます。



さらに、要素をタグで囲むことも出来ます。この場合、二つの要素は階層構造となり、外側の要素を親要素、内側の要素を子要素といいます。子要素は親要素よりも階層が一つ深くなります。子要素の中に要素を配置すれば、さらに一階層深い「孫要素」となります。

このようにHTMLは要素の階層を重ねて記述されています。



## タグの属性

HTMLタグはその種類に応じて設定を付加することが出来ます。この設定のことを「**属性**」、設定の内容のことを「**属性値**」といいます。

付加することの出来る属性はタグごとに異なっています。例えばリンクを定義する<a>タグには、リンク先を指定するhref属性を付加し、属性値としてリンク先のURLなどを設定することが出来ます。

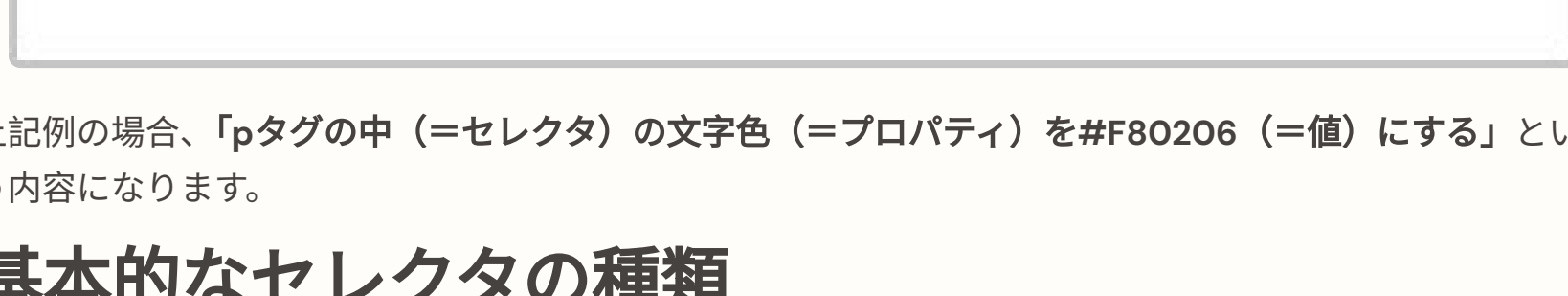
```
<a href="https://www.asobou.co.jp/" target="_blank">株式会社アーティス</a>
属性 属性値                                属性 属性値
```

CSSの適用対象として活用されるid名やclass名も、id属性の属性値、class属性の属性値となります。CSSはこれら要素、階層、属性、属性値を指定して装飾を施します。

## CSSの基本文法

CSSは、「セレクタ」「プロパティ」「値」の3つで構成されます。CSSはこれら要素、階層、属性、属性値を指定して装飾を施します。

それでは、一般的によく使われるCSSセレクタを紹介します。



上記の場合、「pタグの中 (=セレクタ) の文字色 (=プロパティ) を#F80206 (=値) にする」という内容になります。

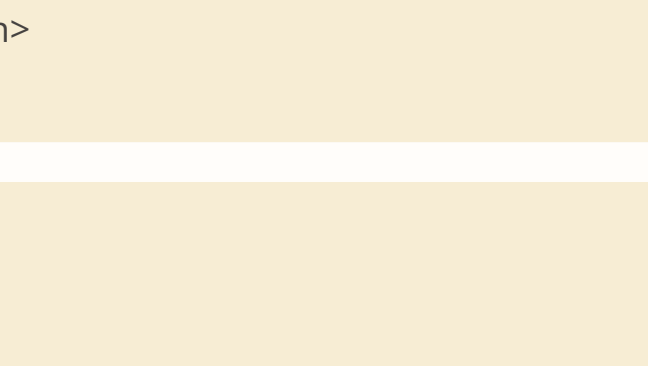
## 基本的なセレクタの種類

スタイルを適用したい要素を指定する際に用いられる「CSSセレクタ」ですが、CSSセレクタには幅広い種類があり、様々な指定方法があります。

それでは、一般的によく使われるCSSセレクタを紹介します。

### 1. 要素の指定

HTMLタグを指定することで、装飾を施す範囲を指定します。



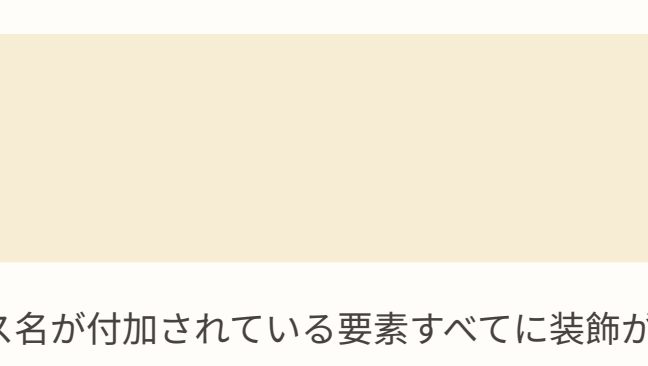
```
<p>pタグの内容</p>
<span>spanタグの内容</span>
```

```
p {
  color: #F80206;
}
```

上記のようにHTMLとCSSの記述をしたとき、セレクタで指定してあるpタグにのみ装飾が適用されます。spanタグには適用されません。

### 2. .class (クラス名の指定)

「.(ドット) クラス名」と記述することで、指定のクラスに装飾が適用されます。



```
<p>pタグの内容</p>
<p class="example">クラス名exampleのpタグ内容</p>
<span>spanタグの内容</span><br>
<span class="example">クラス名exampleのspanタグ内容</span>
```

```
.example {
  color: #F80206;
}
```

クラス名を指定するとそのクラス名が付加されている要素すべてに装飾が適用されます。要素に続けてクラス名を指定することで、そのクラス名が付加された要素に装飾が適用されます。

pタグの内容  
クラス名exampleのpタグ内容  
spanタグの内容  
クラス名exampleのspanタグ内容

```
p.example {
  color: #F80206;
}
```

exampleクラスが付加されているpタグ要素にのみ、装飾が適用されます。

### 3. #id (ID名の指定)

「# (シャープ) ID名」と記述することで、指定のIDに装飾が適用されます。

pタグの内容  
ID名exampleのpタグ内容  
spanタグの内容  
ID名example2のspanタグ内容

```
<p>pタグの内容</p>
<p id="example">ID名exampleのpタグ内容</p>
<span>spanタグの内容</span><br>
<span id="example2">ID名example2のspanタグ内容</span>
```

```
p#example {
  color: #F80206;
}
```

クラス名の指定と同じく、要素に続けて指定することで、そのID名が付加された要素に装飾が適用されます。

### 4. A B (子孫セレクタの指定)

セレクタの次に半角スペースを入れセレクタを指定することで、指定の親要素内のすべての子要素に装飾が適用されます。

pタグの内容  
pタグの子要素のspanタグの内容  
spanタグの内容

```
<p>pタグの内容</p>
<p><span>pタグの子要素のspanタグの内容</span></p>
<span>spanタグの内容</span>
```

```
p span {
  color: #F80206;
}
```

pタグに囲われたspanタグにのみ、装飾が適用されます。

クラス名を指定することも可能です。

pタグの内容  
pタグの子要素のspanタグの内容  
クラス名exampleのspanタグの内容  
spanタグの内容

```
<p>pタグの内容</p>
<p><span>pタグの子要素のspanタグの内容</span></p>
<p><span class="example">pタグの子要素のクラス名exampleのspanタグの内容</span></p>
<span>spanタグの内容</span>
```

```
p span.example {
  color: #F80206;
}
```

上記の例の場合は、pタグで囲われていて、かつexampleのクラス名が指定されているspanタグ要素に装飾が適用されます。

また、装飾の範囲は孫要素など、階層が深くなってもすべてに適用されます。



```
<div>divタグの内容</div>
<div>
  <span>divタグの子要素のspanタグの内容</span>
  <div><p><span>divタグの孫要素のspanタグの内容</span></p></div>
</div>
<span>spanタグの内容</span>
```

```
div span {
  color: #F80206;
}
```

divタグで囲われているすべてのspanタグに装飾が適用されます。

### 5. A > B (子セレクタの指定)

セレクタの次に「>」を入れセレクタを指定することで、指定の親要素内の一階層下の子要素に装飾が適用されます。

```
div > span {
  color: #F80206;
}
```

### 6. A > B (子セレクタの指定)

セレクタの次に「>」を入れセレクタを指定することで、指定の親要素内の一階層下の子要素に装飾が適用されます。

divタグの内容  
divタグの子要素のspanタグの内容  
divタグの孫要素のspanタグの内容  
spanタグの内容

```
<div>divタグの内容</div>
<div>
  <span>divタグの子要素のspanタグの内容</span>
  <div><p><span>divタグの孫要素のspanタグの内容</span></p></div>
</div>
<span>spanタグの内容</span>
```

```
div > span {
  color: #F80206;
}
```

先ほどと同じHTMLの記述であっても、divタグの一階層下にあるspanタグにのみ装飾が適用されます。

### 7. A + B (隣接セレクタの指定)

セレクタの次に「+」を入れセレクタを指定することで、指定の要素に隣接した要素に装飾が適用されます。

pタグの内容  
divタグの内容  
divタグに隣接したpタグの内容  
pタグに隣接したpタグの内容

```
<p>pタグの内容</p>
<div>divタグの内容</div>
<p>divタグに隣接したpタグの内容</p>
<p>pタグに隣接したpタグの内容</p>
```

```
div + p {
  color: #F80206;
}
```

divタグの直後にあるpタグに装飾が適用されます。

### 8. A ~ B (要素の後ろにある同じ階層のセレクタの指定)

セレクタの次に「~」を入れセレクタを指定することで、指定の要素の後ろにある同じ階層の要素に装飾が適用されます。

divタグより前にあるpタグの内容  
divタグの内容  
divタグより後にあるpタグの内容  
divタグより後にあるpタグの内容  
divタグの子要素のpタグの内容  
divタグより後にあるpタグの内容

```
<p>divタグより前にあるpタグの内容</p>
<div>divタグの内容</div>
<p>divタグより後にあるpタグの内容</p>
<p>divタグより後にあるpタグの内容</p>
<div><p>divタグの子要素のpタグの内容</p></div>
<p>divタグより後にあるpタグの内容</p>
```

```
div ~ p {
  color: #F80206;
}
```

divタグの以降にあるすべてのpタグに装飾が適用されます。ただし階層が違うと適用されません。

### 9. A, B (複数のセレクタの指定)

セレクタの次に「, (カンマ)」を入れセレクタを指定することで、複数の要素に同じ装飾が適用されます。

divタグの内容  
pタグの内容  
spanタグの内容

```
<div>divタグの内容</div>
<p>pタグの内容</p>
<span>spanタグの内容</span>
```

```
div, span {
  color: #F80206;
}
```

divタグとspanタグに装飾が適用されます。

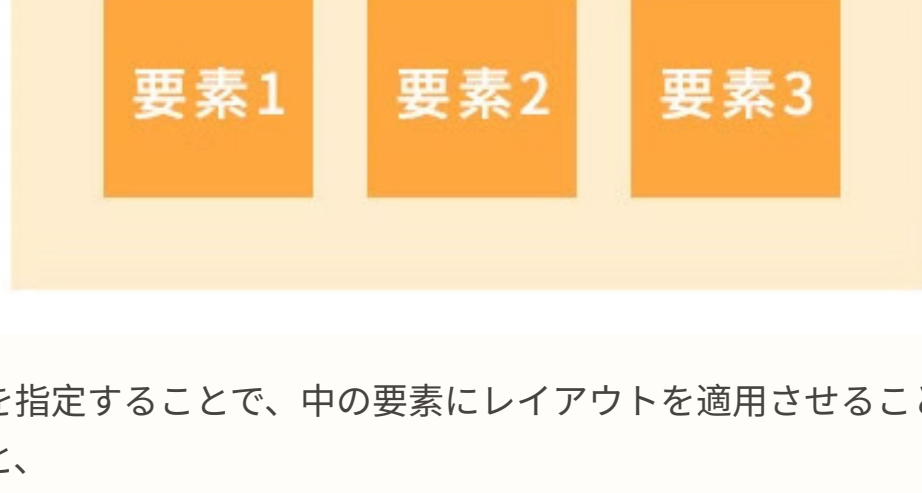


# CSSフレックス編+α知識

## フレックスボックスの構造、要素とコンテナ

フレックスボックスはレイアウトを適用させたい子要素の「要素」と、それらをラップする親要素の「コンテナ」から構成されます。

### コンテナ



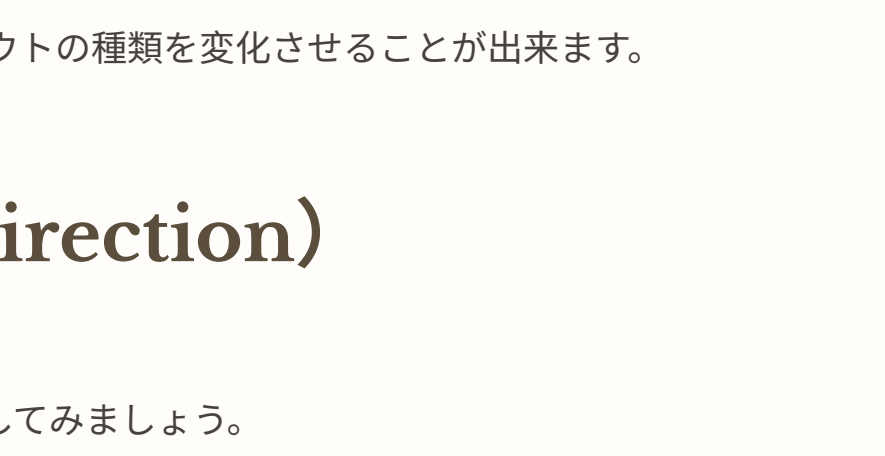
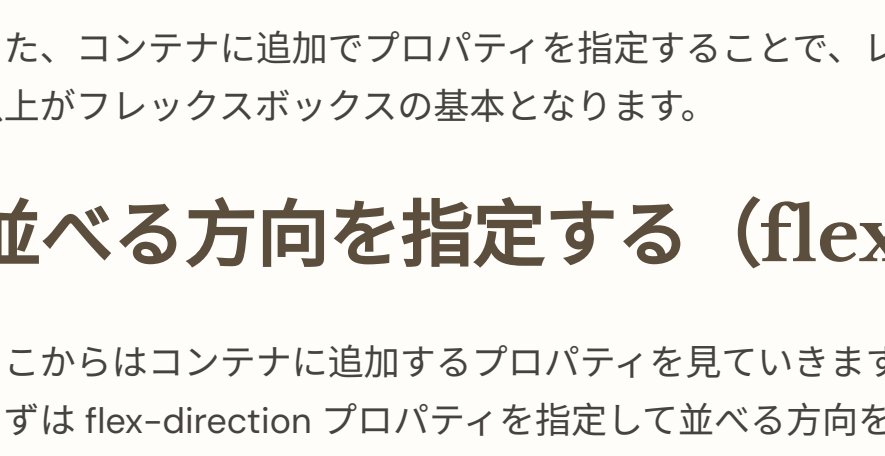
コンテナに対して `display:flex` を指定することで、中の要素にレイアウトを適用させることが可能となります。下記のようにHTMLを記述すると、

```
<div class="container">
  <div class="item">
    要素1
  </div>
  <div class="item">
    要素2
  </div>
  <div class="item">
    要素3
  </div>
</div>
```

コンテナはクラス名 `container` でその中のクラス名 `item` が要素となります。この場合は `container` に `display:flex` を指定することで、`item` のレイアウトを変更することが出来ます。

```
.container {
  display: flex;
}
```

このように、要素をラップしているコンテナに対し、`display: flex;` とCSSを一行追加するだけで直下の子要素を横並びにすることが出来ます。



また、コンテナに追加でプロパティを指定することで、レイアウトの種類を変化させることが出来ます。以上がフレックスボックスの基本となります。

## 並べる方向を指定する (flex-direction)

ここからはコンテナに追加するプロパティを見ていきます。まずは `flex-direction` プロパティを指定して並べる方向を指定してみましょう。CSSの記述は下記になります。

```
.container {
  display: flex;
  flex-direction: row-reverse;
}
```

使用できる値は次の4つになります。

row (初期値)	左から右に横並びで子要素を配置
row-reverse	右から左に横並びで子要素を配置
column	上から下に縦並びに子要素を配置
column-reverse	下から上に縦並びに子要素を配置

それぞれの表示結果は下記ようになります。



このように `flex-direction` プロパティを追加すると、デフォルトでは左から右へ水平方向に並べるレイアウトを変更することが出来ます。

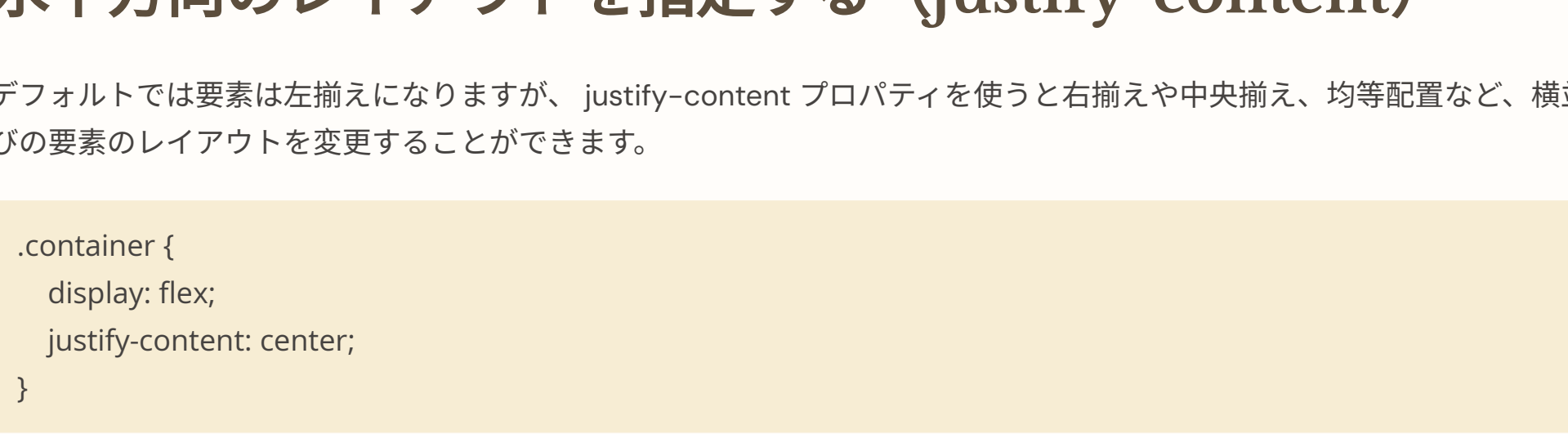
## 折り返しを指定する (flex-wrap)

子要素の横幅の合計が親要素の横幅を超えると、デフォルトでは子要素の横幅が縮んでいきます。このような場合に子要素を折り返すかどうかを指定できるのが `flex-wrap` プロパティです。

```
.container {
  display: flex;
  flex-wrap: wrap;
}
```

使用できる値は次の3つになります。

nowrap (初期値)	子要素を折り返ししないで、一行に並べる
wrap	子要素を折り返しさせ、上から下へ複数行で並べる
wrap-reverse	子要素を折り返させ、下から上へ複数行で並べる



折り返さずにレイアウトを崩したくない場合も、折り返して要素を変形させたくない場合も、コーディング時には両方のパターンがありますので非常によく使うプロパティです。

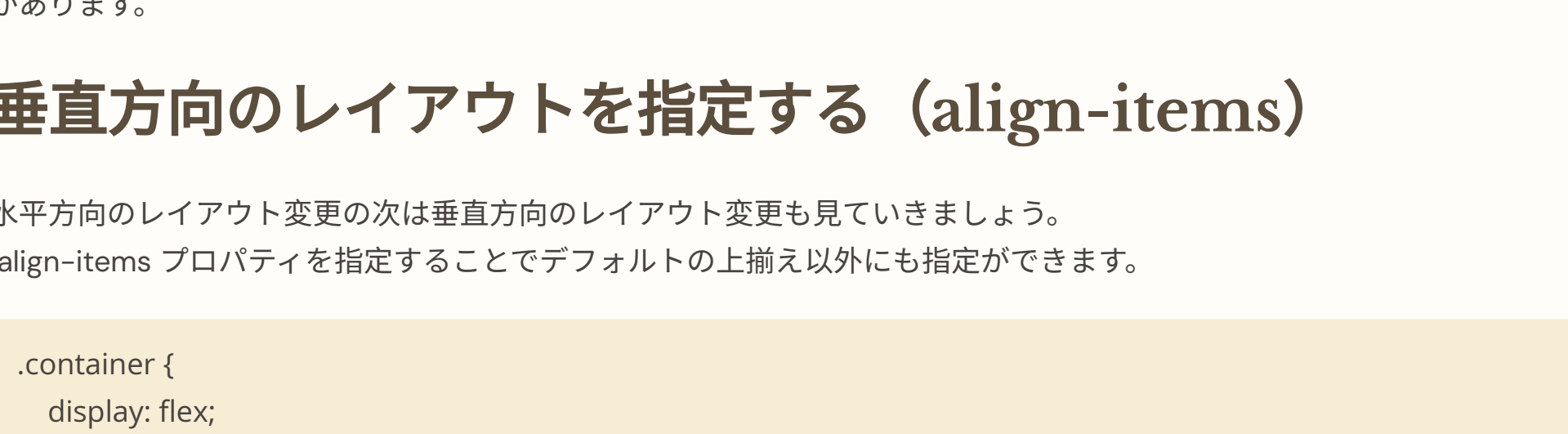
## 水平方向のレイアウトを指定する (justify-content)

デフォルトでは要素は左揃えになりますが、`justify-content` プロパティを使うと右揃えや中央揃え、均等配置など、横並びの要素のレイアウトを変更することができます。

```
.container {
  display: flex;
  justify-content: center;
}
```

使用できる値は次の5つになります。

flex-start (初期値)	子要素を左揃えで配置する
flex-end	子要素を右揃えで配置する
center	子要素を中央揃えで配置する
space-between	子要素を均等配置する。このとき左右端の要素を親要素の端に接して配置する
space-around	子要素を均等配置する。このとき左右端の要素も親要素の端から均等に離して配置する



`space-between` と `space-around` はどちらも均等配置ですが、親要素(コンテナ)の端に接するかそうでないかの違いがあります。

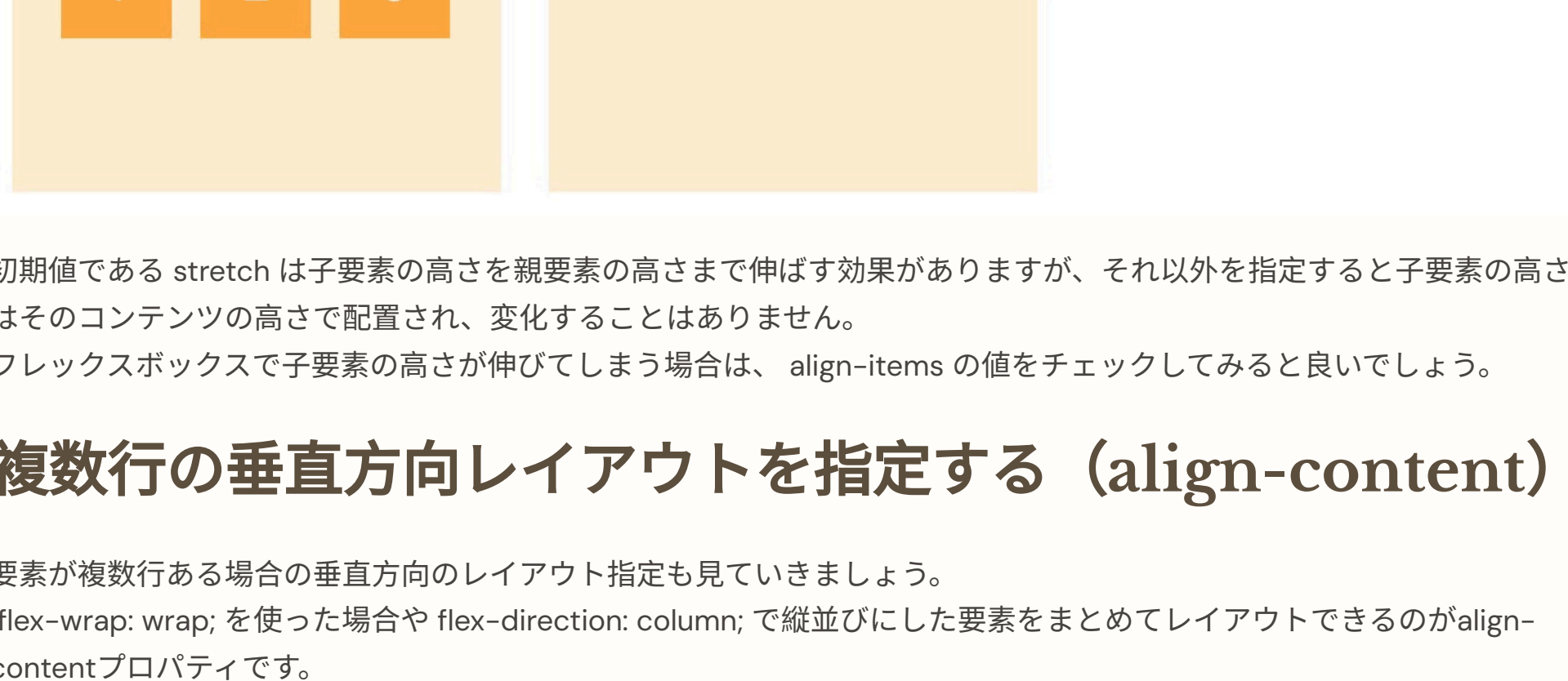
## 垂直方向のレイアウトを指定する (align-items)

水平方向のレイアウト変更の次は垂直方向のレイアウト変更も見ていきましょう。`align-items` プロパティを指定することでデフォルトの上揃え以外にも指定ができます。

```
.container {
  display: flex;
  align-items: center;
}
```

使用できる値は次の5つになります。

stretch (初期値)	子要素の中で一番高い要素に合わせて全ての子要素の高さを統一させて配置する
flex-start	親要素の上端を基準に上揃えで配置する。
flex-end	親要素の下端を基準に下揃えで配置する。
center	親要素の上下中央を基準に、中央揃えで配置する。
baseline	子要素のベースラインで揃えて配置する。



初期値である `stretch` は子要素の高さを親要素の高さまで伸ばす効果がありますが、それ以外を指定すると子要素の高さはそのコンテンツの高さで配置され、変化することはありません。

フレックスボックスで子要素の高さが伸びてしまう場合は、`align-items` の値をチェックしてみると良いでしょう。

## 複数行の垂直方向レイアウトを指定する (align-content)

要素が複数行ある場合の垂直方向のレイアウト指定も見ていきましょう。`align-content` プロパティを使うことでデフォルトの上揃え以外にも指定ができます。

```
.container {
  display: flex;
  align-content: center;
}
```

使用できる値は次の6つになります。

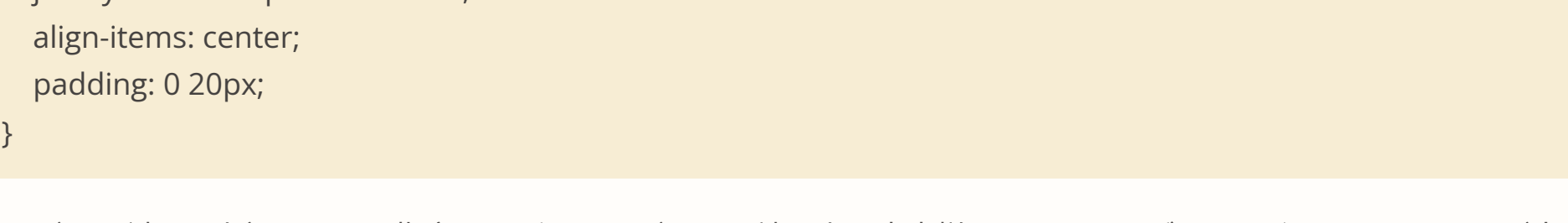
stretch (初期値)	親要素の高さに合わせて子要素の高さを統一させて配置する
flex-start	親要素の上端を基準に上揃えで配置する。このとき子要素の高さは変化しない
flex-end	親要素の下端を基準に下揃えで配置する。このとき子要素の高さは変化しない
center	親要素の上下中央を基準に、中央揃えで配置する。このとき子要素の高さは変化しない
space-between	子要素を均等配置する。このとき上下端の要素を親要素の端に接して配置する
space-around	子要素を均等配置する。このとき上下端の要素も親要素の端から均等に離して配置する



`align-items` と異なる点として、`space-between` や `space-around` といった均等配置の値があることが見て取れます。効果としては `justify-content` と同じで方向が垂直方向になったものだと思っておきましょう。

## フレックスボックスの使用例

最後にフレックスボックスの具体的使用例を紹介します。例えばよくあるヘッダーのレイアウトにおいて、左にサイトロゴ、右にGナビがある場合フレックスボックスで解決できます。

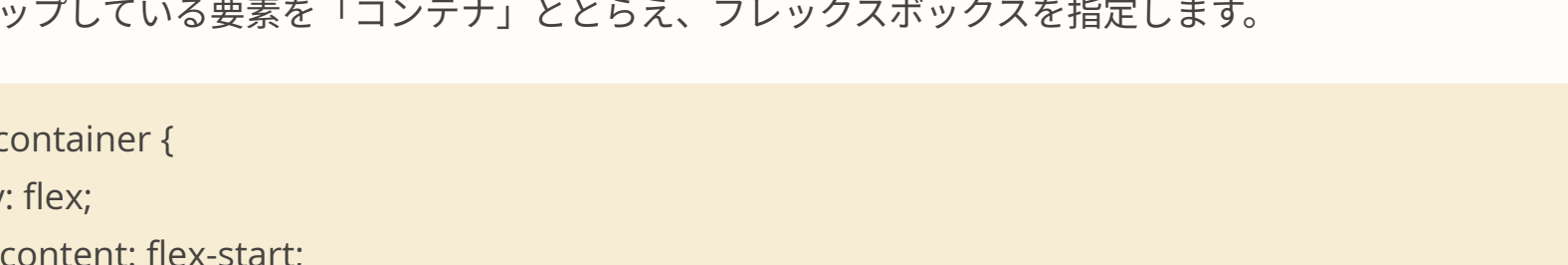


サイトロゴとGナビを「要素」ととらえ、それをラップしているヘッダーを「コンテナ」ととらえることが出来ます。

```
#header {
  display: flex;
  justify-content: space-between;
  align-items: center;
  padding: 0 20px;
}
```

ヘッダーに対して上記のように指定すれば、ヘッダーの両端に上下中央揃えでサイトロゴとGナビを配置することが出来ます。また、ヘッダーの端に余白を設けたい場合はpaddingを指定すれば解決です。

バナーを配置する場合でもフレックスボックスが役立ちます。



バナーをラップしている要素を「コンテナ」ととらえ、フレックスボックスを指定します。

```
.banner-container {
  display: flex;
  justify-content: flex-start;
  flex-wrap: wrap;
  gap: 30px 24px;
}
```

上記のように指定すると、左揃えで折り返る配置が実装出来ます。また、`gap` を指定すれば要素の間に余白を設けることも出来ます。



# CSSメディアクエリ編 + α知識

1

## 基本構文の理解:

メディアクエリは、@media キーワードから始まり、特定のメディアタイプ（例: screen）と条件（例: max-width: 600px）を指定します。この条件が真の時、中括弧 {} 内のスタイルが適用されます。

2

## ブレイクポイントの設定:



モバイル・スマホWeb・WordPressのSEO塾.com



【2022年10月最新版】レスポンシブCSSメディアクエリ (...)

メディアクエリ・ブレイクポイントもCSS次第ではあるが、iPhoneなどの大型スマホ、さらにはiPadの大型タブレットの登場で、レスポ...

3

実際に後ほどHP制作で復習行います

# CSSアニメーション編 + $\alpha$ 知識

1

## アニメーションの定義:

アニメーションは、静止画像を高速で切り替えることで、動いているように見せる技術です。

2

## 完成デモサイト:

[MAKOTOお勧めデモサイト](#)

3

## animation-timing-function:

- **ease** 少しゆっくり始まって、少しゆっくりになって終わります。
- **ease-in** ゆっくりと始まり、その後は等速で動きます。
- **ease-out** 等速で動き、最後はゆっくりと終わります。
- **ease-in-out** ゆっくり始まって加速し、ゆっくりになって終わります。easeよりもメリハリの効いた動きです。
- **linear** 最初から最後まで等速で動きます。




 easings.net



### イーザング関数チートシート

イーザング関数はアニメーションの速度を指定して、動きをより自然にします。現実のオブジェクトは単に一定の速度で動かず、即座に...

4

 mdn web docs

 MDN Web Docs



### animation - CSS: カスケーディングスタイルシート | MDN

animation は CSS の一括指定プロパティで、スタイルの間のアニメーションを適用します。これは animation-name, animation-...



# おまけ

## HTML

```
<div class="anim-box"></div>
```

## CSS

### フェードイン:

```
.anim-box.fadein.is-animated {  
  
  animation: fadeIn 0.7s cubic-bezier(0.33, 1, 0.68, 1) forwards;}  
  
@keyframes fadeIn {  
  
  0% {  
  
    opacity: 0;  
  
  }  
  
  100% {  
  
    opacity: 1;  
  
  }  
  
}
```

### 跳ねる:

```
.anim-box.poyoyon.is-animated {  
  
  animation: poyoyon 0.5s cubic-bezier(0.12, 0, 0.39, 0) 1 forwards;}  
  
}  
  
@keyframes poyoyon {  
  
  0% {  
  
    transform: translateX(140px);  
  
    opacity: 0;  
  
  }  
  
  50% {  
  
    transform: translateX(0);  
  
  }  
  
  65% {  
  
    transform: translateX(30px);  
  
  }  
  
  100% {  
  
    transform: translateX(0);  
  
  }  
  
  20%,100% {  
  
    opacity: 1;  
  
  }  
  
}
```

### きらんと光る:

```
.anim-box.kiran {  
  
  opacity: 1;  
  
  overflow: hidden;  
  
  position: relative;  
  
  cursor: pointer;  
  
}  
  
.anim-box.kiran::before {  
  
  background-color: #fff;  
  
  content: "";  
  
  display: block;  
  
  position: absolute;  
  
  top: -100px;  
  
  left: 0;  
  
  width: 30px;  
  
  height: 100%;  
  
  opacity: 0;  
  
  transition: cubic-bezier(0.32, 0, 0.67, 0);  
  
}  
  
.anim-box.kiran:hover::before {  
  
  animation: kiran 0.5s linear;  
  
}  
  
@keyframes kiran {  
  
  0% {  
  
    transform: scale(2) rotate(45deg);  
  
    opacity: 0;  
  
  }  
  
  20% {  
  
    transform: scale(20) rotate(45deg);  
  
    opacity: 0.6;  
  
  }  
  
  40% {  
  
    transform: scale(30) rotate(45deg);  
  
    opacity: 0.4;  
  
  }  
  
  80% {  
  
    transform: scale(45) rotate(45deg);  
  
    opacity: 0.2;  
  
  }  
  
  100% {  
  
    transform: scale(50) rotate(45deg);  
  
    opacity: 0;  
  
  }  
  
}
```

# HTML & CSS 模写コーディング編 課題

ECサイト、アプリ紹介ページ、ポートフォリオを制作してみよう

実践! ポートフォリオサイトを模写しよう

実践! アプリのランディングページを模写しよう

実践! ECサイトを模写しよう

# 自力でHP制作しよう編 課題

## ヒント:

### コンテンツ幅

コンテンツの横幅は960pxで横のパディングは4%です。  
メインビジュアルだけ全幅にします。

### メインビジュアル

全幅で高さは420px固定です。

### About

テキストのみのエリアです。住所のブロックとプロフィールのブロックで分割されているのがポイントですが、特に決まりはないので自由にコーディングしてみてください。

### Works

画像を横並びに配置します。  
3つめの画像で折り返されるように、折り返し設定も忘れないようにしてください。  
スマホ表示の際は縦1列に並びます。

### News

日付とニュースタイトルを横並びにしてリストのように並べます。  
スマホ表示の際は日付とタイトルが縦に並びます。

\*納期は2週間。

\*スマホサイズとパソコンサイズの2つを制作して下さい。

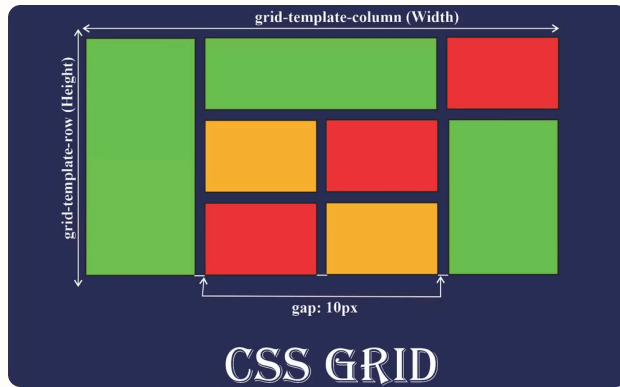
\*出来上がればgithubで共有して下さい。

## 完成後:

こちらのHP完成後もし別のHPも制作したいと思う方は連絡お待ちしております。



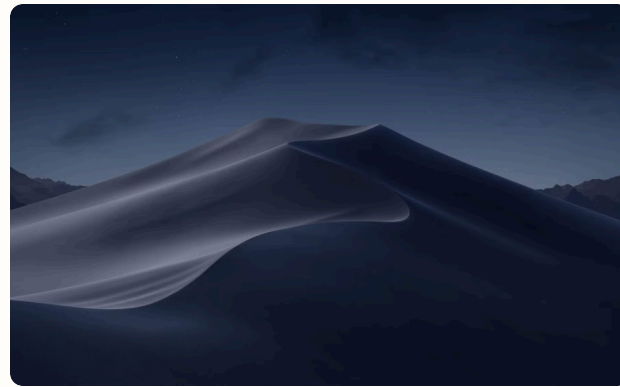
# 最新のHTMLとCSSのトレンドとベストプラクティス



## CSSグリッド

グリッドは、効率的で柔軟なレイアウトを提供するためにCSSで作成された新しい機能です。

[CSS Grid Layout を極める! \(基礎編\) - Qiita](#)



## ダークモード

ダークモードは、目の疲れを軽減し、視認性を向上させるために、HTMLとCSSで実装できるトレンドの一つです。

[Webサイトをダークモードに対応させよう](#)



## 3Dグラフィック

HTML5とCSS3を使用して、3Dグラフィックエレメントをウェブページに追加することができます。

[「5分でわかるCSS3のグラフィックス機能」サンプル](#)



# おわりに

今回のプレゼンテーションでは、HTMLとCSSの基本から応用知識までを解説しました。これらの知識を習得することで、ウェブデザインのスキルを向上させ、クリエイティブなウェブサイトを作成することができます。