HTML5

MD31

#2

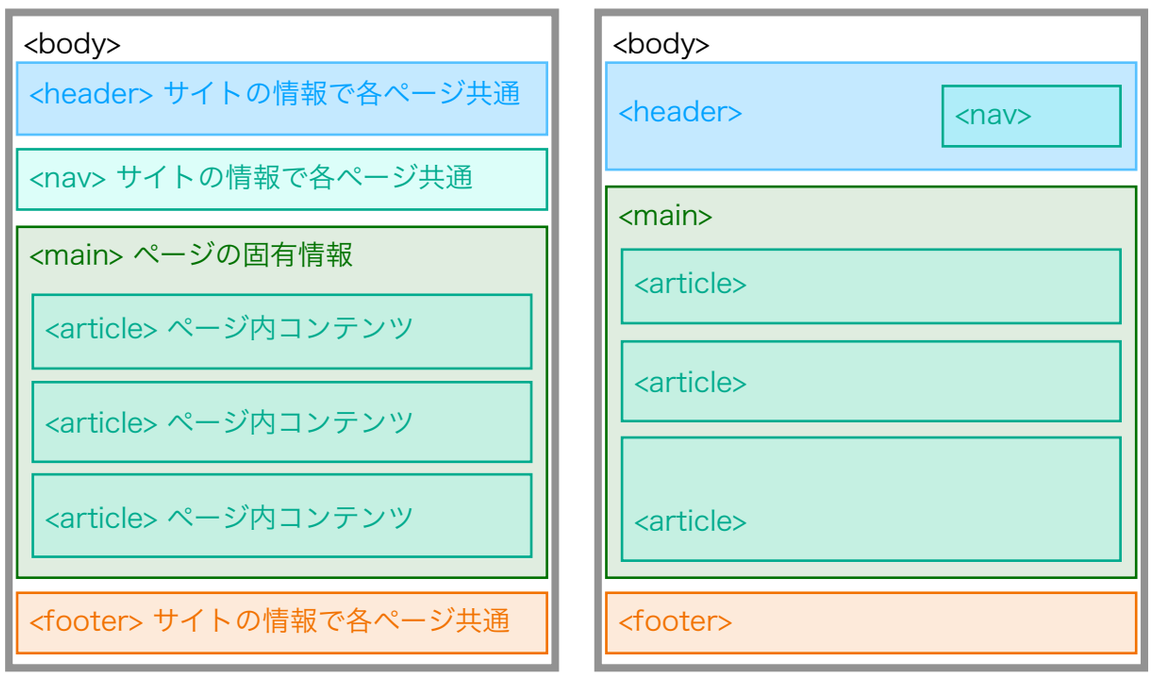
07 レイアウトの設計

コンテンツが縦に並べると1カラムレイアウト
横に並べると２カラム・3カラムレイアウト

基本レイアウトとセクショニングコンテンツ

1カラムレイアウト

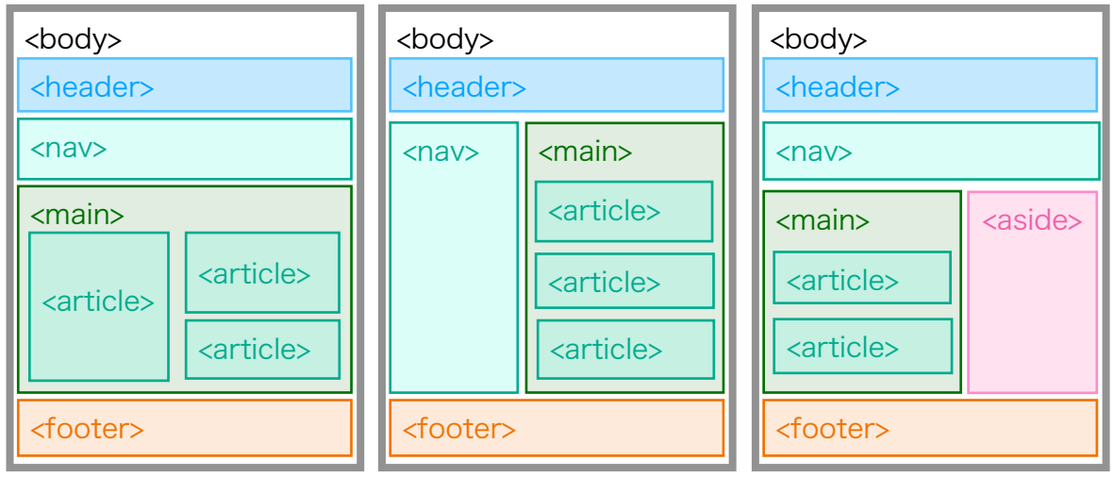
<article>内で画像とテキストや、<section>同士が横に並んでいても、
<article>が縦に並んでいれば1カラムレイアウト。
横幅の大きさに関係なく、柔軟にページの魅力を伝えることができるのが特徴。
テキストが少なめのサイトに向いている
スマートフォンなどモバイルデバイスでは1カラムレイアウトが基本になる。



2カラムレイアウト

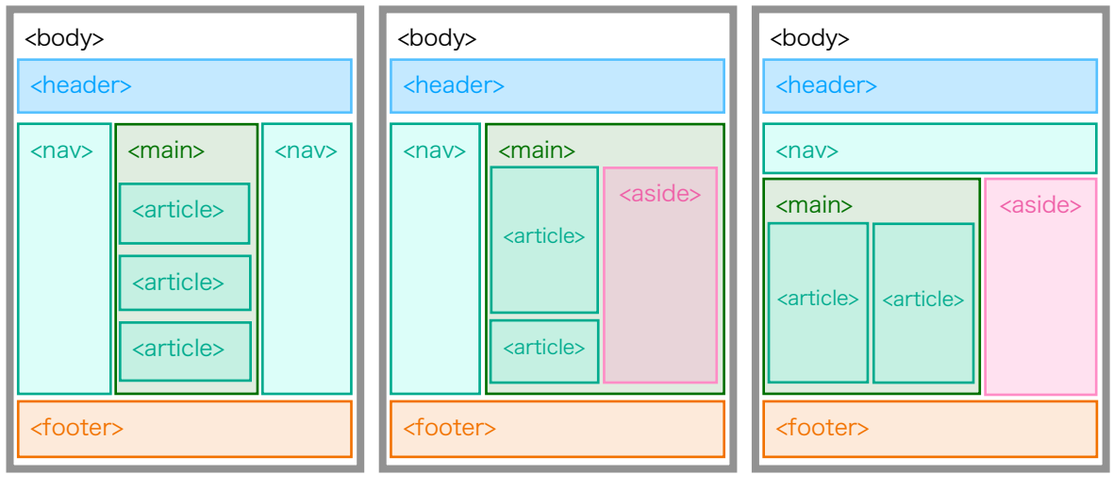
<article>を左右に並べるレイアウトと、<nav>と<main>を左右に並べるレイアウト、<main>と<aside>を左右に並べるレイアウトの３パターンがある。

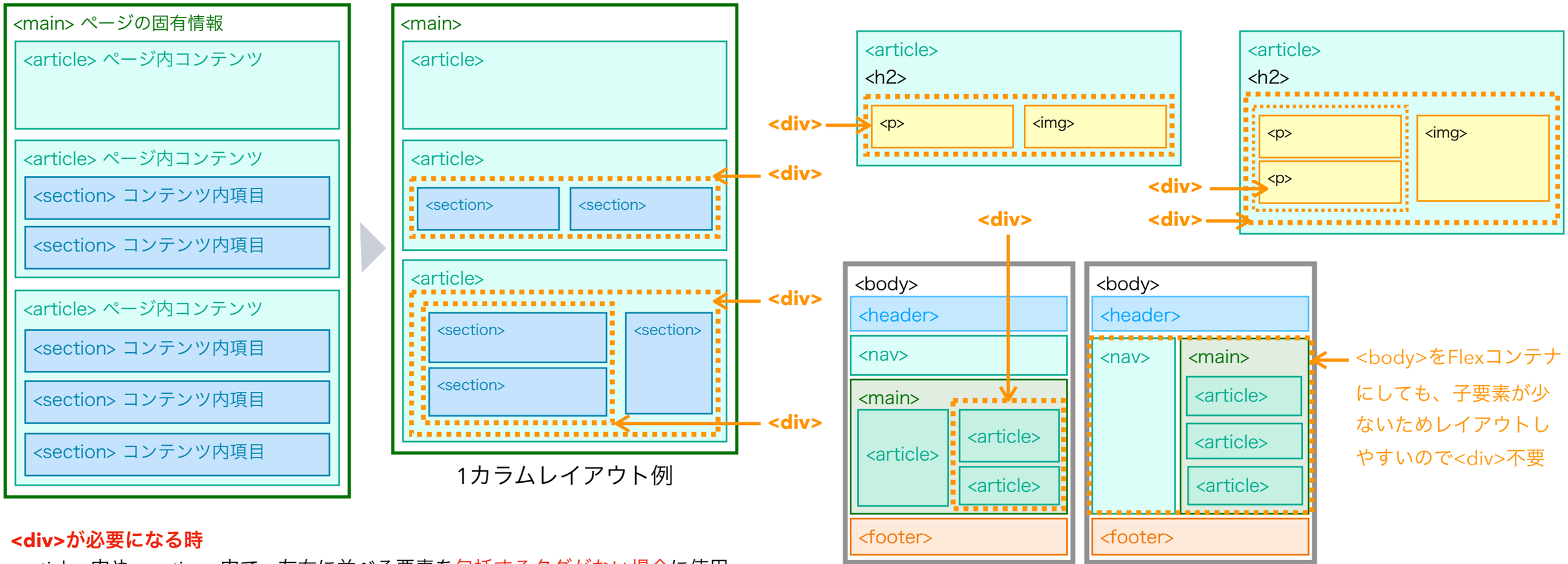
一般的なPC・タブレット用のレイアウトで、画面幅が狭いにスマートフォンには向かない。



3カラムレイアウト

情報量・ページ数が多いサイトに向き、複数のナビゲーションを配置し、操作性を向上するために採用するレイアウト。
画面幅が狭いスマートフォンには向かない。
<nav>を両サイドに並べるレイアウトや、<nav>と<aside>を両サイドに並べるレイアウト、<article>や<aside>を横に並べるレイアウトなど、パターンは豊富。





1カラムレイアウト例

** が必要になる時**

<article>内や<section>内で、左右に並べる要素を包括するタグがない場合に使用

また、2カラム以上の時の1カラムが、上下に分割される1カラムで使用

主にFlexBoxでのレイアウトで、Flexコンテナとして が必要になることが多い

コンテンツレイアウトとセクショニングコンテンツ

1カラムレイアウトでは、<article>内で<section>が左右に並んでいても「1カラムレイアウト」になります。

<p>や<img>が左右を並べる場合も同様ですが、左右に並べる際には が必要になってきます。

** の使い方**

はデザイン目的で使用するグループ化の役割を持つタグです。

主にレイアウトで使用することが多くなります。

<section>や<p><img>など、<article>内の<h2>以外の要素を左右に並べる場合は、必ず が必要になってくると覚えておくと良いでしょう。

ナビゲーションなど、<li>を左右に並べる場合は<ul>が包括する親要素になるので は不要です。

左右に並べる要素を包括する親要素がない場合、必ず で囲むようにするということになります。

レスポンシブデザインでは、PCとモバイルデバイスで大きくレイアウトが変わることがあるため、2カラム/3カラムの基本レイアウトで を使うのはお勧めしません。<nav>と<main>を左右に並べるなど2カラムでレイアウトするために、 で包括してもレスポンシブデザインで差し支えがないのであれば問題ありませんが、基本レイアウトで無理に を使うこと必要はないでしょう。

08 コーディングの流れ

HTMLコーディングの流れを意識しながら作業することで、
作業時間の短縮や記述ミスを未然に防ぐ

コーディングワークフロー

最も多いコーディング時のミスがタグの閉じ忘れです。ミスにどの時点で気付けるかによって作業時間に影響してきます。このような無駄・非効率を無くすためにも、無意識に作業ができるまでは次の「流れ」に沿ってコーディングを進めることをお勧めします。

コーディングの流れを完全に覚え身に付けることで、自分がよくするミスの癖を知ることになり、作業スピードも効率も上げることができます。

コーディングの流れ

1. HTMLファイルの作成

2. viewportメタタグを記述

<meta name="viewport" content="width=device-width,initial-scale=1">
viewportメタタグはレスポンシブデザインに必要な記述
デバイスごとの画面サイズに応じて最適な表示に変換してくれる

3. <title>のタイトル文を入力し、ファイルを保存

4. 文章を全て段落<p>で入力

画像<img>を表示する部分はaltテキストを入力しておく

5. 段落<p>文章の内容に合わせて適切なタグに変換

見出し→リスト→画像→テーブル→フォーム→リンク

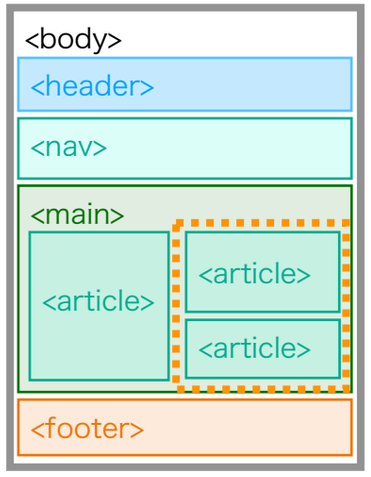
6. ページの全体の文書構造をまとめる

<header> , <nav> , <main> , <footer>

7. セクショニングコンテンツで文章をまとめる

<article> , <aside> , <section>

8. レイアウト用のボックス図を作成し、必要な箇所を でまとめる（id名の入力）



レイアウト用のボックスは詳細に描くほど、CSSの作業効率が上がる
制作ワークフローの「プランニング」のフェイズで、ワイヤーフレームまたはラフデザインの段階で作成できる仕様書の一つ

9. リセットCSSを参照する

10.レイアウト用のCSSファイルを作成・保存し、レイアウト飲み完成させる

主にwidth・margin・display（flex/grid系）・float・clear・overflowなど

11.ページ専用のCSSファイルを作成・保存し、スタイリングとレイアウトの微調整を行う

（全ページ共通するスタイル部分は別ファイルへ移動させる）