1. Display 속성 (block/inline 차이점, 각 예제 코드)

**display 속성: block vs inline**

display 속성 --> HTML 요소가 **웹페이지에서 어떻게 배치되고 차지하는 공간**을 결정.  
가장 기본적인 값은 block과 inline.

**1. block 요소**

* **특징**:
  + 한 줄을 **전체 차지**합니다 (다른 요소가 옆에 올 수 없음).
  + 항상 **새 줄**에서 시작합니다.
  + **너비와 높이를 설정**할 수 있습니다.
* **사용 예**:
  + <div>, <p>, <h1> 같은 요소.
* **예제 코드**:

<div style="display: block; background: lightblue; padding: 10px;">Block Element 1</div>

<div style="display: block; background: lightcoral; padding: 10px;">Block Element 2</div>

**결과**:

* Block Element 1과 Block Element 2는 **서로 다른 줄**에 배치된다.

**2. inline 요소**

* **특징**:
  + *콘텐츠 크기만큼*의 공간만 차지합니다 (옆에 다른 요소가 올 수 있음).
  + **새 줄을 시작하지 않습니다**.
  + 너비와 높이는 설정할 수 없습니다 (텍스트 크기와 같은 콘텐츠 크기로만 설정됨).
* **사용 예**:
  + <span>, <a>, <strong> 같은 요소가 기본적으로 inline 요소입니다.
* **예제 코드**:

<span style="display: inline; background: lightgreen; padding: 5px;">Inline Element 1</span>

<span style="display: inline; background: lightpink; padding: 5px;">Inline Element 2</span>

**결과**:

* Inline Element 1과 Inline Element 2가 **같은 줄**에 배치된다.

**3. block과 inline 차이점 요약**

| **속성** | **block** | **inline** |
| --- | --- | --- |
| 줄 바꿈 | 항상 새 줄에서 시작 | 같은 줄에 다른 요소와 함께 있음 |
| 차지 공간 | 한 줄 전체 차지 | 콘텐츠 크기만큼만 차지 |
| 너비/높이 설정 | 가능 | 불가능(텍스트 크기 같은 거로만 설정) |
| 예시 태그 | <div>, <p>, <h1> | <span>, <a>, <b> |

**4. block과 inline을 둘다 사용한 예제**

<!DOCTYPE html>

<html lang="en">

<head>

<style>

.block {

**display: block;**

background: lightblue;

padding: 10px;

margin-bottom: 10px;

}

.inline {

**display: inline;**

background: lightpink;

padding: 2px;

margin-right: 5px;

}

</style>

</head>

<body>

<!-- Block 요소 -->

<div class="block">Block Element 1</div>

<div class="block">Block Element 2</div>

<!-- Inline 요소 -->

<span class="inline">Inline Element 1</span>

<span class="inline">Inline Element 2</span>

</body>

</html>

**코드 결과**

Block Element 1

Block Element 2

Inline Element 1 Inline Element 2

**요약**

* **Block**: 큰 영역, 새 줄로 내려 감. 레이아웃을 나눌 때 사용. (책의 단락처럼, div ).

가로와 세로 크기를 모두 설정할 수 있다.

* **Inline**: 텍스트나 작은 부분을 꾸밀 때 사용. 줄 안에 이어 붙음. (문장 안의 단어처럼, span ).

들어있는 콘텐츠의 높이, 너비에 맞게 자동으로 크기가 결정된다.

1. HTML의 Layout ( +css layout 종류 설명, 각 예제 코드)

**HTML의 Layout**

HTML의 Layout은 웹페이지를 구성하는 요소들을 배치하는 방법을 의미한다. 웹페이지에서 텍스트, 이미지, 버튼 같은 요소들이 어디에, 어떻게 놓일지를 정하는 것. **HTML로 기본 구조를 만든 뒤, CSS를 사용해 레이아웃을 꾸미고 조정한다.**

**CSS Layout 종류**

**1. Flexbox (Flexible Box Layout)**

* **특징**: 한 방향(가로 or 세로)으로 요소를 정렬하고 공간을 효율적으로 나눔.
* **사용 예**: 아이템을 가로로 정렬하거나, **중앙에 정렬할** 때 적합.

<div style="**display: flex**; justify-content: center; align-items: center; height: 200px; background: lightblue;">

<div style="width: 100px; height: 100px; background: coral;">A</div>

<div style="width: 100px; height: 100px; background: teal;">B</div>

<div style="width: 100px; height: 100px; background: gold;">C</div>

</div>

* **결과**: 박스 A, B, C가 가로로 나란히 정렬되며, 가운데 위치.

**2. CSS Grid Layout**

* **특징**: 행(row)과 열(column)을 사용한 2차원 레이아웃. 복잡한 페이지 디자인에 유용.
* **사용 예**: 사진 갤러리, *카드 레이아웃* 등을 만들 때 적합.

<div style="**display: grid**; grid-template-columns: 1fr 1fr 1fr; gap: 10px; background: lightgray;">

<div style="background: coral;">1</div>

<div style="background: teal;">2</div>

<div style="background: gold;">3</div>

<div style="background: lightgreen;">4</div>

</div>

* **결과**: 박스 4개가 3개의 열로 나뉘어 배치. 각 열의 너비는 동일, 박스 간의 간격은 10px.

**3. Float Layout**

* **특징**: 원래는 텍스트를 감싸기 위해 만들어졌지만, 과거에는 요소를 배치하는 데 많이 사용됐다. 지금은 Flexbox나 Grid가 더 주로 사용된다.
* **사용 예**: 간단한 이미지를 정렬하거나 레이아웃을 나눌 때.

<div>

<div style="float: left; width: 30%; height: 100px; background: coral;">Left</div>

<div style="float: left; width: 70%; height: 100px; background: lightblue;">Right</div>

</div>

* **결과**: 첫 번째 박스가 왼쪽에, 두 번째 박스가 오른쪽에 배치.

**요약**

1. **Flexbox**: *한 줄*에 정렬하거나*중앙 정렬* 등 간단한 배치.
2. **Grid**: 행과 열을 사용해 *복잡한* 2차원 배치.
3. **Float**: 예전 방식, 단순 정렬에 사용되지만 이제는 잘 안 씀.
4. HTML의 박스모델 (content / padding / border / margin, 각 예제 코드)

**HTML의 박스 모델**

HTML의 **박스 모델(Box Model)**은 웹페이지에서 **각 요소가 차지하는 공간**을 정의하는 기본 규칙! 이걸 집에 비유하면

1. **Content (콘텐츠)**:
   * 집 안의 **방**. 글자, 이미지 같은 **실제 내용**이 들어가는 공간.
2. **Padding (패딩)**:
   * 방 안에서 **벽과 가구 사이의 여유 공간**. 콘텐츠와 경계선(Border) 사이의 간격.
3. **Border (테두리)**:
   * 방을 감싸는 **벽**. 콘텐츠와 패딩을 둘러싸는 테두리.
4. **Margin (마진)**:
   * 집과 다른 집 사이의 **외부 공간**. 요소와 요소 사이의 간격.

**박스 모델 구조**

| Margin | <- 요소와 요소 사이의 외부 공간

| Border | <- 요소를 감싸는 테두리

| Padding | <- 콘텐츠와 테두리 사이의 내부 여백

| Content | <- 실제 내용이 들어가는 공간

**예제 코드**

<!DOCTYPE html>

<html lang="en">

<head>

<style>

.box {

width: 200px; /\* Content의 너비 \*/

height: 100px; /\* Content의 높이 \*/

padding: 20px; /\* 콘텐츠와 테두리 사이 여백 \*/

border: 5px blue; /\* 테두리 크기와 색상 \*/

margin: 30px; /\* 요소와 요소 사이 외부 공간 \*/

background-color: lightgray; /\* Content의 배경 색상 \*/

}

</style>

</head>

<body>

<div class="box">This is Content</div>

</body>

</html>

**위 코드 결과**

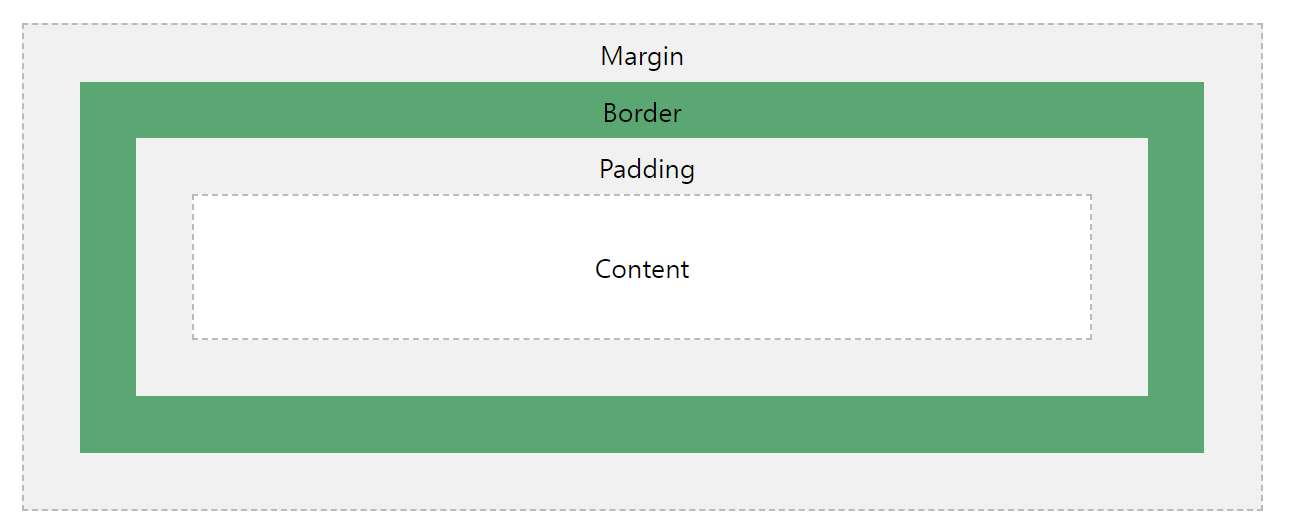
This is Content

1. **Content**:
   * 텍스트 "This is Content"가 들어있는 **200px x 100px** .
2. **Padding**:
   * 콘텐츠와 테두리(Border) 사이에 **20px** 여백.
3. **Border**:
   * 콘텐츠와 패딩을 둘러싸는 **파란색 테두리(5px)**.
4. **Margin**:
   * 박스와 다른 요소 또는 화면 가장자리 사이에 **30px** 외부 공간.

**더 쉽게! 박스 모델의 레이어**

* 콘텐츠(Content) → 패딩(Padding) → 테두리(Border) → 마진(Margin) 순서로 쌓여있다.
* **Padding**은 내부 공간, **Margin**은 외부 공간!!

**눈여겨볼 부분**

1. **HTML 박스**:
   * 마치 상자를 여러 겹으로 싸놓은 모습.  
     

이미지 출처: <https://www.w3schools.com/css/css_boxmodel.asp>

1. **레이아웃**: 어디에 배치할까, like 설계도!  
   코드가 실제로 공간을 차지할 때 **Content + Padding + Border + Margin** 모두 포함.  
   즉, 우리가 <div>를 200px이라고 지정해도, 실제 화면에서는 이 값에 Padding, Border, Margin이 더해지면서 훨씬 큰 영역을 차지하게 된다!

**요약**

박스 모델은 HTML 요소의 공간 구조를 이해하는 기본 개념:

* **Content**: 실제 내용.
* **Padding**: 내용과 테두리 사이 여백.
* **Border**: 테두리.
* **Margin**: 요소와 다른 요소 사이의 공간.

박스 모델을 알면 웹페이지의 디자인과 레이아웃을 더 쉽게 다룰 수 있다~~

**여기서 잠깐!!**

**"박스 모델(Box Model)"** = **"레이아웃(Layout)"** 이 아님!!!

**차이점**

1. **박스 모델(Box Model)**:
   * **각 HTML 요소**가 화면에서 **얼마나 많은 공간을 차지하는지**를 설명하는 모델.
   * 콘텐츠(Content), 패딩(Padding), 테두리(Border), 마진(Margin)으로 구성.
2. **레이아웃(Layout)**:
   * 웹페이지 전체에서 **각 요소들을 어떻게 배치**할지 설계하는 방식.
   * 박스 모델뿐만 아니라 **CSS의 Flexbox**, **Grid**, **Positioning** 등 여러 기술을 사용해 요소를 배치.
   * 예시: header는 맨 위, main은 중앙, footer는 아래에 배치.

**비유로 정리**

**레이아웃**: 집 전체에서 **방들을** 어떻게 배치할지 설계하는 것.  
**박스 모델**: 집 안에서 **방 하나의** 크기와 구조를 정하는 것.

**결론**

박스 모델은 레이아웃을 이해하기 위한 더 작은 기본 개념.  
**박스 모델 = 레이아웃**은 아님!

1. HTML의 id, class ( 둘의 차이점, 각 예제 코드)

HTML에서 **id**와 **class**는 1️ 요소(element)를 식별하고 스타일링하거나 2️ JavaScript로 조작하기 위해 사용.

**1. id**

* **고유 식별자**: 하나의 HTML 문서에서 특정 요소에만 사용할 수 있음. 즉, id는 "한 번만" 사용.
* **특정 요소를 지목할 때 유용**: JavaScript나 CSS에서 특정 *하나*의 요소에만 적용하고 싶을 때 사용.

**특징**

* 한 문서에서 **고유**해야 함.
* CSS에서는 # 기호로 선택.

**예제**

<!DOCTYPE html>

<html>

<head>

<style>

/\* id 스타일링 \*/

**#unique-box** {

color: white;

background-color: blue;

padding: 10px;

}

</style>

</head>

<body>

<div **id="unique-box">**나는 id로 스타일링된 박스!</div>

</body>

</html>

**2. class**

* **공통 스타일**: 비슷한 스타일을 공유하는 여러 요소에 적용.
* **그룹화 용도**: 여러 개의 요소에 동일한 스타일을 적용하거나 JavaScript로 함께 조작할 때 사용.

**특징**

* 같은 클래스 이름을 여러 요소에 재사용.
* CSS에서는 . 기호로 선택.

**예제**

<!DOCTYPE html>

<html>

<head>

<style>

/\* class 스타일링 \*/

**.shared-style** {

color: black;

background-color: lightpink;

padding: 10px;

margin: 5px;

}

</style>

</head>

<body>

<div **class="shared-style">**나는 class로 스타일링된 첫 번째 박스!</div>

<div class="shared-style">나는 class로 스타일링된 두 번째 박스!</div>

</body>

</html>

**차이점 요약**

| **특징** | **id** | **class** |
| --- | --- | --- |
| **고유성** | 문서 내에서 **1번만 사용 가능** | 여러 요소에서 **반복 사용 가능** |
| **CSS 선택자** | #id-name | .class-name |
| **사용 용도** | 특정 하나의 요소에 스타일 적용 | 여러 요소에 공통 스타일 적용 |

\*여기서 *문서 내에서 1번만 사용가능*이라는 표현은

**< 올바른 사용 >**

<div id="header">이건 헤더 영역입니다</div>

<p id="intro">이건 소개 문구입니다</p>

여기서는 **header**와 **intro**라는 id가 각각 *다른* 요소에 붙어 있다.  
id는 각각 하나의 요소에만 적용되므로 잘 작성된 코드이다.

**< 🚨잘못된🚨 사용 >**

<div id="header">첫 번째 헤더</div>

<div id="header">두 번째 헤더</div>

여기서는 **두 개의 <div> 요소에 같은 id header가 적용되어** 있다.  
JavaScript나 CSS에서 #header를 사용하면 어떤 요소를 선택해야 할지 🚨혼란🚨

**비유**

* **id**: 주민등록번호처럼 **유일한 식별자** (한 사람*당* 하나만 가질 수 있음).
* **class**: 학교 반 이름처럼 **여러 사람이 공유** (같은 반 학생들이 *모두* 같은 이름을 가짐).