4. CSS 박스 모델(Box Model)

콘텐츠(Content)

CSS 박스 모델(Box Model)은 **모든 HTML 요소를 직사각형 박스 형태로 다루는 시각적 모델**이다.
이 모델의 핵심 구성 요소 중 가장 안쪽에 위치한 것이 바로 **콘텐츠(Content)**이다.

---

1. 콘텐츠(Content)란?

콘텐츠 영역(content box)은 **HTML 요소의 실제 내용이 표시되는 공간**이다.
이 영역에는 다음과 같은 내용이 포함된다:

- 텍스트
- 이미지
- 버튼 안의 아이콘
- 입력 필드의 값
- 기타 HTML 요소 내부의 표시되는 정보

즉, 콘텐츠는 사용자에게 **직접 보여지는 정보 그 자체**이다.

---

2. 콘텐츠 영역의 크기

기본적으로 콘텐츠 영역의 크기는 다음과 같은 속성에 의해 결정된다:

```
width: [너비];
height: [높이];
```

```
div {
  width: 200px;
  height: 100px;
}
```

이렇게 설정한 `width`와 `height`는 **기본적으로 콘텐츠 영역의 크기만 지정**하며,
테두리(border), 패딩(padding), 마진(margin)은 포함되지 않는다.

---

3. `box-sizing`에 따른 크기 계산

기본값: `content-box`

```
box-sizing: content-box;
```

- `width`와 `height`는 **오직 콘텐츠 영역의 크기만** 의미
- 패딩과 보더는 여기에 **추가로 계산됨**

```
총 너비 = width + padding-left/right + border-left/right
총 높이 = height + padding-top/bottom + border-top/bottom
```

대안: `border-box`

```
box-sizing: border-box;
```

- `width`와 `height`는 콘텐츠 + 패딩 + 보더까지 포함한 **전체 박스 크기**로 계산됨
- 실무에서는 이 값을 **초기화 스타일로 자주 사용**함

```
* {
  box-sizing: border-box;
}
```

---

4. 콘텐츠 영역 예시

```
<div class="box">안녕하세요</div>
```

```
.box {
  width: 150px;
  height: 50px;
  background-color: lightblue;
}
```

- `div`의 콘텐츠는 텍스트 `"안녕하세요"`
- 그 텍스트가 들어갈 수 있도록 `150×50px`의 콘텐츠 박스가 생성됨
- 여기에 패딩, 테두리, 마진 등을 추가할 수 있음

---

5. 콘텐츠와 레이아웃의 관계

- 콘텐츠의 길이나 양에 따라 **박스의 크기가 자동 조절**되기도 함
- 예: `height`를 명시하지 않으면, 텍스트 양에 따라 높이가 자동으로 증가
- 이미지 콘텐츠는 `object-fit`, `max-width`, `aspect-ratio` 등을 통해 조절 가능

---

6. 콘텐츠 오버플로우와 관련 속성

콘텐츠가 콘텐츠 박스를 넘칠 경우:

```
overflow: visible;   /* 기본값: 넘쳐도 표시함 */
overflow: hidden;    /* 넘친 부분을 잘라냄 */
overflow: scroll;    /* 항상 스크롤 표시 */
overflow: auto;      /* 넘칠 경우에만 스크롤 표시 */
```

```
white-space: nowrap;  /* 줄바꿈 방지 */
text-overflow: ellipsis;  /* 넘친 텍스트 ... 처리 */
```

결론

> 콘텐츠는 박스 모델에서 **가장 중심이 되는 실제 표시 영역**이며,
> `width`, `height`, `overflow`, `box-sizing` 등을 통해 콘텐츠의 배치와 표시 방식을 정교하게 제어할 수 있다.

패딩(Padding)

패딩(Padding)은 CSS 박스 모델(Box Model)의 네 가지 주요 구성 요소 중 두 번째로,
콘텐츠(Content)와 테두리(Border) 사이의 안쪽 여백(내부 여백)을 말한다.
즉, **내용과 테두리 사이의 공간**이다.

---

1. 패딩의 개념

```
+-----------------------------+
|          border             |
|  +----------------------+   |
|  |       padding        |   |
|  |   +---------------+  |   |
|  |   |   content     |  |   |
|  |   +---------------+  |   |
|  +----------------------+   |
+-----------------------------+
```

- 콘텐츠가 **너무 붙어보이는 것**을 막고 **시각적인 여유 공간**을 주는 역할
- 텍스트, 이미지 등의 콘텐츠와 경계선 사이를 띄우기 위해 사용
- **배경색과 배경 이미지도 패딩 영역에 적용**된다

---

2. 기본 문법

```
padding: 값;
```

한 줄 요약 문법

작성 방식	의미
`padding: 20px;`	모든 방향에 20px
`padding: 10px 20px;`	위아래 10px, 좌우 20px
`padding: 10px 15px 20px;`	위 10px, 좌우 15px, 아래 20px
`padding: 5px 10px 15px 20px;`	위, 오른쪽, 아래, 왼쪽 순

---

3. 방향별 속성

속성	설명
`padding-top`	위쪽 패딩
`padding-right`	오른쪽 패딩
`padding-bottom`	아래쪽 패딩
`padding-left`	왼쪽 패딩

예시:

```
div {
  padding-top: 10px;
  padding-left: 20px;
}
```

---

4. 예제

```
<div class="box">안쪽 여백이 있는 상자</div>
```

```
.box {
  padding: 20px;
  background-color: lightblue;
  border: 1px solid navy;
}
```

→ 콘텐츠 주변에 20px의 여유 공간이 생기며, 배경색은 패딩까지 채워진다.

---

5. `box-sizing`에 따른 크기 해석

- 기본값인 `box-sizing: content-box`에서는,

  ```
  총 너비 = width + padding + border
  ```

- 실무에서 많이 쓰는 `box-sizing: border-box`를 설정하면,

  ```
  * {
    box-sizing: border-box;
  }
  ```

  `width` 안에 padding과 border까지 **포함되어 계산**된다

---

6. 패딩 vs 마진의 차이

항목	패딩(Padding)	마진(Margin)
위치	콘텐츠 안쪽	콘텐츠 바깥
영향	배경색 적용됨	배경색 적용 안됨
용도	내부 간격 조정	외부 간격 조정
박스 크기에 영향	`content-box` 일 때는 영향 없음	항상 영향 없음

---

7. 실무 활용 팁

- 버튼, 입력창, 카드 컴포넌트 등에서 **내용이 너무 붙지 않도록** 하기 위해 필수
- 모바일 반응형 UI에서 **터치 영역 확보** 용도로 자주 사용
- 텍스트 콘텐츠의 **가독성 향상**에 매우 중요

결론

> 패딩은 콘텐츠를 감싸는 **내부 여백 영역**으로,
> 사용자 인터페이스에서 **가독성과 미적 균형**을 잡는 데 매우 중요한 요소다.
> 특히 배경이 콘텐츠만이 아니라 **패딩까지 포함해 칠해진다는 점**을 기억하자.

테두리(Border)

테두리(Border)는 CSS 박스 모델에서 **패딩(Padding) 바깥쪽에 위치한 선**으로,
요소의 경계선을 시각적으로 표현하거나 **강조 효과**, **구분선**, **입체감 표현** 등에 사용된다.

---

1. 테두리의 위치

박스 모델 기준:

```
+----------------------------+
|         border             |
|  +----------------------+  |
|  |       padding        |  |
|  |   +--------------+   |  |
|  |   |   content    |   |  |
|  |   +--------------+   |  |
|  +----------------------+  |
+----------------------------+
```

---

2. 기본 문법

```
border: [두께] [스타일] [색상];
```

예시

```
div {
  border: 2px solid black;
}
```

→ 두께 2px, 실선(solid), 색상은 검정

---

3. 세부 속성 분리

각 속성은 개별로도 지정 가능:

속성	설명
`border-width`	테두리 두께
`border-style`	테두리 선의 종류
`border-color`	테두리 색상

예:

```
div {
  border-width: 1px;
  border-style: dashed;
  border-color: gray;
}
```

---

4. 방향별 테두리 설정

속성	설명
`border-top`	위쪽 테두리
`border-right`	오른쪽 테두리
`border-bottom`	아래쪽 테두리
`border-left`	왼쪽 테두리

```css
.box {
  border-top: 2px solid red;
  border-bottom: 1px dotted blue;
}
```

---

5. 선 스타일 종류

스타일	설명
`none`	없음 (기본값)
`solid`	실선
`dashed`	점선
`dotted`	점선 (작은 점)
`double`	이중선
`groove`, `ridge`	입체감 있는 선
`inset`, `outset`	박스 안/밖으로 돌출된 효과

---

6. 테두리 둥글게: `border-radius`

```css
div {
  border-radius: 10px;
}
```

- 모서리를 둥글게 만듦
- 원형으로 만들려면 `50%` 사용

```css
img {
  border-radius: 50%;
}
```

---

7. 배경과의 관계

- **배경색(`background-color`)은 테두리 안쪽(padding까지)**에 적용됨
- **테두리는 그 외곽 경계선으로 시각적 구분 역할**을 함

---

8. `outline`과의 차이점

항목	`border`	`outline`
위치	요소 **내부 테두리**	요소 **외부 테두리**
크기 계산 포함	포함됨	포함되지 않음
둥글기 조정	가능 (`border-radius`)	불가능
레이아웃 영향	있음	없음
접근성 강조에 주로 사용	❌	✅

---

9. 실무 예제

입력 필드 포커스 테두리

```
input:focus {
  border: 2px solid blue;
}
```

강조 카드 디자인

```
.card {
  border: 1px solid #ccc;
  border-radius: 8px;
  padding: 16px;
}
```

결론

> `border`는 웹 요소에 **선명한 경계와 구조적 구분**을 제공하고,
> **디자인의 시각적 안정성**을 높이는 중요한 스타일 속성이다.
> `border`, `padding`, `box-sizing`을 함께 이해하면 요소의 전체 크기 계산에 강해진다.

마진(Margin)

마진(Margin)은 CSS 박스 모델에서 **요소의 바깥쪽 여백(외부 여백)**을 의미한다.
요소와 요소 사이의 **공간을 조정**하며, **레이아웃 배치와 간격 제어의 핵심 도구**로 사용된다.

---

1. 마진의 위치

박스 모델에서 마진은 가장 바깥쪽에 위치한다.

```
+----------------------------+ ← margin
| +------------------------+  |
| |        border          |  |
| |  +------------------+  |  |
| |  |     padding      |  |  |
| |  |  +------------+  |  |  |
| |  |  |  content   |  |  |  |
| |  |  +------------+  |  |  |
| |  +------------------+  |  |
| +------------------------+  |
+----------------------------+
```

---

2. 기본 문법

```
margin: 값;
```

단축 문법

작성 방식	의미
`margin: 20px;`	모든 방향에 20px
`margin: 10px 20px;`	위·아래 10px, 좌·우 20px
`margin: 10px 15px 20px;`	위 10px, 좌·우 15px, 아래 20px
`margin: 5px 10px 15px 20px;`	위, 오른쪽, 아래, 왼쪽 순

---

3. 방향별 속성

속성	설명
`margin-top`	위쪽 마진
`margin-right`	오른쪽 마진
`margin-bottom`	아래쪽 마진
`margin-left`	왼쪽 마진

```
.box {
  margin-top: 20px;
  margin-bottom: 30px;
}
```

---

4. 자동 마진 (`margin: auto`)

```
div {
  width: 300px;
  margin: 0 auto;
}
```

- 좌우 마진을 자동으로 계산하여 **가운데 정렬**함
- 부모 요소가 `block` 또는 `flex`, `grid`일 때 유효

---

5. 마진 병합(Margin Collapse)

- **세로 방향의 마진(위/아래)이 겹칠 경우 병합되어 하나만 적용**
- 보통 **큰 쪽의 값 하나만 적용됨**

예시:

```
p {
  margin: 20px 0;
}
```

```
<p>첫 문단</p>
<p>두 번째 문단</p>
```

→ 두 문단 사이 여백은 40px이 아니라 **20px**

---

6. 마진과 인라인 요소

- 인라인 요소(`span`, `a`, `em` 등)에는 수직 마진(top/bottom)이 **거의 적용되지 않음**
- 수평 마진(left/right)은 적용됨

---

7. 마진 vs 패딩 차이

항목	마진(Margin)	패딩(Padding)
위치	요소 바깥쪽	요소 안쪽
배경색 영향	없음	배경색 적용됨
요소 간 거리 조정	O	X
크기 계산 영향	없음 (`width` / `height`와 별도)	`content-box`에서는 영향 없음, `border-box`에서는 포함됨

---

8. 실무 예제

카드 간격 띄우기

```
.card {
  margin-bottom: 20px;
}
```

가운데 정렬

```
.container {
  width: 600px;
  margin: 0 auto;
}
```

마진 초기화 (Reset)

```
* {
  margin: 0;
  padding: 0;
  box-sizing: border-box;
}
```

결론

> 마진은 요소 간 간격을 제어하는 **레이아웃 설계의 핵심 도구**이며,
> `margin: auto`와 `margin collapse` 등 **독특한 동작 특성**을 잘 이해해야 예측 가능한 UI를 만들 수 있다.

`box-sizing` 속성

`box-sizing`은 CSS에서 요소의 **전체 크기를 어떻게 계산할 것인지**를 결정하는 매우 중요한 속성이다.
기본적으로 HTML 요소는 `width`와 `height`에 **padding과 border가 포함되지 않지만**,
`box-sizing`을 사용하면 그 계산 방식을 바꿀 수 있다.

---

1. 기본 문법

```
element {
  box-sizing: content-box | border-box;
}
```

---

2. 주요 값

① `content-box` (기본값)

- `width`와 `height`는 **콘텐츠 영역만** 포함
- padding과 border는 **추가로 계산됨**
- 따라서 실제 요소의 전체 크기는 **계산된 크기보다 커짐**

```
div {
  width: 200px;
  padding: 20px;
  border: 5px solid black;
  box-sizing: content-box;
}
```

📐 **총 너비 계산**:

```
200 (width)
+ 20*2 (padding)
+ 5*2 (border)
= 250px
```

---

② `border-box`

- `width`와 `height`에 **padding과 border까지 포함**
- 콘텐츠 크기를 자동으로 줄여 전체 크기를 **정확히 맞춰줌**
- 반응형 디자인, 컴포넌트 설계 시 **가장 많이 사용됨**

```
div {
  width: 200px;
  padding: 20px;
  border: 5px solid black;
  box-sizing: border-box;
}
```

📐 **총 너비 계산**:

```
width 전체 = 200px
→ 콘텐츠 영역은 200 - 20*2 - 5*2 = 150px
```

---

3. 비교 요약

구분	`content-box` (기본값)	`border-box`
포함 범위	콘텐츠만	콘텐츠 + 패딩 + 보더

구분	`content-box` (기본값)	`border-box`
총 크기 계산	width + padding + border	width 그대로
레이아웃 제어	복잡하고 계산 필요	직관적이고 관리 쉬움
실무 활용도	낮음	높음 (거의 표준)

---

4. 실무에서는 `border-box` 기본 설정

모던 웹 개발에서는 거의 대부분 다음과 같은 초기화 코드를 사용한다:

```
*,
*::before,
*::after {
  box-sizing: border-box;
}
```

- 모든 요소와 가상 요소(::before, ::after 포함)에 `border-box` 적용
- 전체 레이아웃이 예측 가능해지고, **너비·높이 계산 실수 방지**

---

5. 활용 예

```
.box {
  width: 300px;
  padding: 20px;
  border: 5px solid gray;
  box-sizing: border-box;
}
```

→ `.box`의 **총 너비는 정확히 300px**
→ 안쪽 콘텐츠 영역만 자동으로 조절됨

결론

> `box-sizing`은 CSS에서 요소의 크기 계산 방식을 제어하는 속성으로,
> `border-box`를 사용하면 **더 직관적이고 정확한 레이아웃 구성**이 가능하다.
> 그래서 실무에서는 거의 항상 `border-box`로 설정해두는 것이 표준이다.

마진 겹침(Margin Collapse)

마진 겹침(Margin Collapse)은 CSS에서 **세로 방향(margin-top / margin-bottom)** 마진이 특정 조건에서 **겹쳐서 하나로 합쳐지는 현상**이다.
이는 수직 레이아웃을 자동으로 정리하려는 브라우저의 동작 방식이며, 때로는 **예상치 못한 레이아웃 문제**를 유발할 수 있다.

---

1. 언제 마진이 겹침(Collapse) 되는가?

A. 형제 요소 사이의 마진

두 요소가 **연속으로 배치**될 때, 위쪽 요소의 `margin-bottom`과 아래쪽 요소의 `margin-top`이 **겹친다**.

```
p {
  margin: 20px 0;
}
```

```
<p>첫 문단</p>
<p>두 번째 문단</p>
```

→ 두 문단 사이의 여백은 20px + 20px = 40px이 아니라
큰 값인 20px 하나만 적용됨

B. 부모와 자식 간 마진

자식 요소의 `margin-top` 또는 `margin-bottom`이 부모와 **겹쳐지는 경우**:

```
<div class="parent">
  <p class="child">내용</p>
</div>
```

```
.parent {
  background: lightgray;
}

.child {
  margin-top: 30px;
}
```

→ `parent`에 `padding`이나 `border`가 없고, `overflow`가 `visible`이면
자식의 마진이 부모 바깥으로 삐져나옴
즉, `child`의 `margin-top`이 `parent` 바깥쪽에 적용됨

C. 빈 요소 사이에서

처럼 내용이 없고, 마진만 있는 요소는 상하 마진이 겹칠 수 있다.

```
div {
  margin: 50px 0;
}
```

→ 위아래 50px씩 있어도 **총 100px이 아니라 50px만 적용**

---

2. 겹치지 않는 경우

다음과 같은 상황에서는 **마진 겹침이 발생하지 않는다**:

- 수평 마진 (`margin-left`, `margin-right`) → 절대 겹치지 않음
- 요소 사이에 **border, padding, content**가 존재
- 부모에 다음 중 하나라도 설정된 경우:
  - `overflow: hidden | auto | scroll`
  - `display: flex | grid`
  - `position: absolute | fixed`
  - `padding`, `border` 값 존재

---

3. 마진 겹침 해결 방법

✅ `padding` 또는 `border` 추가

```
.parent {
  padding-top: 1px; /* 또는 border-top: 1px solid transparent; */
}
```

→ 자식의 마진이 부모 밖으로 안 나가고 **내부에 적용됨**

✅ `overflow: hidden` 사용

```
.parent {
  overflow: hidden;
}
```

→ 마진이 겹치는 것을 방지하고 **레이아웃을 강제로 정리**함

✅ `display: flex` 또는 `position: relative`

```
.parent {
  display: flex;
  flex-direction: column;
}
```

또는

```
.parent {
  position: relative;
}
```

→ 부모가 일반 블록이 아니게 되면 마진 병합이 발생하지 않음

---

4. 시각적으로 이해하기

```
[ p1 ] ← margin-bottom: 20px
[ p2 ] ← margin-top: 30px

→ 실제 간격: max(20, 30) = 30px
```

결론

> 마진 겹침(Margin Collapse)은 **세로 방향 마진을 자동 조정**하여 시각적 중복을 피하려는 CSS의 기본 동작이지만,
> 예상과 다르게 여백이 사라지거나 늘어나 **레이아웃 오류를 유발할 수 있다.**
> 이를 방지하려면 `padding`, `overflow`, `display`, `position` 등을 적극 활용해야 한다.