

# Front-end Bootcamp

## CSS 기초 1

## 지난 강의

- html 기초
- HTML 코딩 = "콘텐츠의 의미를 전달하는 마크업"

# CSS 기초

# CSS의 주요 개념

- CSS: **cascading** style sheets
- 자주 보는 용어
  - 셀렉터(selector), 선택자
  - 키(key), 프로퍼티(property), 이름
  - 값, 밸류(value)
  - 상속(inherit)
  - 부모>자식, 형제+형제

# CSS 리셋

- CSS리셋: 특정 CSS 파일을 로드하여 브라우저 기본 스타일을 초기화(리셋) 한다
- 왜 필요할까요?
  - 브라우저마다 요소의 기본 스타일(CSS)가 다르다
  - 기본 스타일에 의존하고 있다.(상속때문에)
  - 같은 품질의 결과물을 얻을 수 없다.

# CSS 리셋의 종류와 역할

- [Eric Meyer's CSS Reset 2.0](#)
  - 브라우저의 기본 스타일을 전부 없앤다(리셋한다)
  - 제로에서 시작해야 한다.
  - 검증된 방법
- [normalize.css 1.0](#)
  - 완전히 리셋시키지 않고 브라우저 간의 차이만 줄여준다.
  - 브라우저 기본 스타일에 대한 이해가 필요하다.
  - github, bootstrap 등에서 쓰인다
- [html5 doctor reset](#)
- [Yahoo! \(YUI 3\) Reset CSS](#)
- [Universal selector](#)

# CSS 리셋 실습

- 각 리셋을 적용해본다
- 차이점 확인

# CSS 작성 다시보기

CSS선언(정의)블럭

```
{ 속성, 이름    구분자 값  
font-size:16px; 구분자  
color:#ccc;  
font-weight:normal  
} 구분자 생략
```

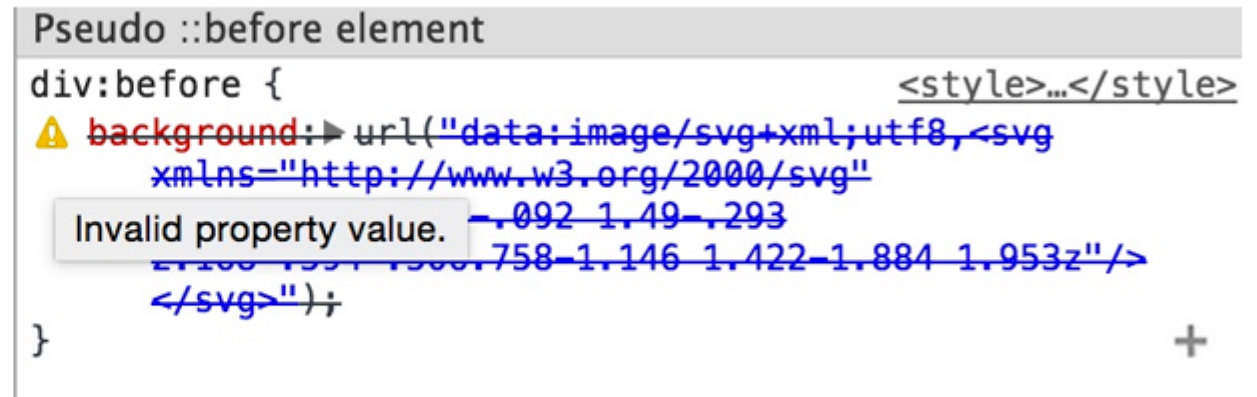
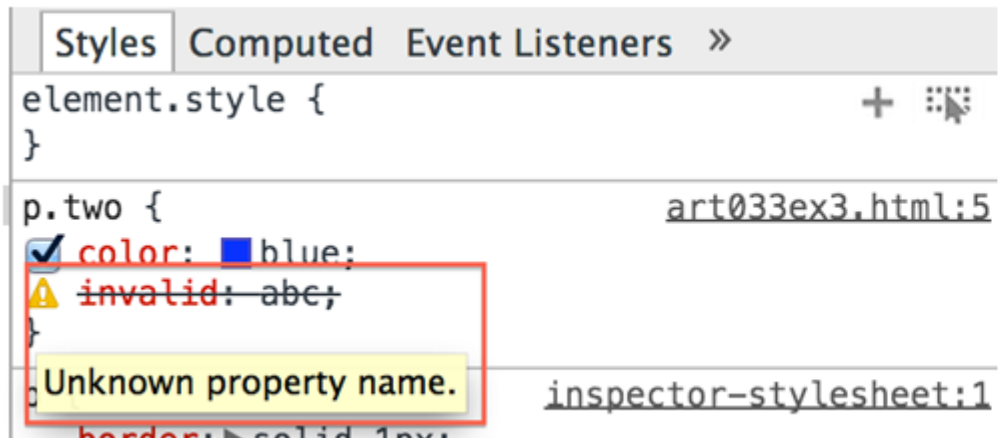


## CSS 작성 체크리스트

- 블록이 잘 설정되었는가
- 속성이름이 잘 작성되었는가
- 값이 잘 작성되었는가
- 작성된 값을 해당 속성이 지원하는가
- 구분자가 빠지지 않았는가

## CSS 작성 오류 확인

- Chrome devtool의 오류 메시지를 확인
- property, key 오류 "Unknown property name"
- value 오류 "Invalid property value"



## 기본 속성 빠르게 보기

### 텍스트

```
font-size: 16px;  
font-weight: bold | normal;  
font-style: italic | oblique | normal;  
text-decoration: underline | line-through | overline;  
font-family: Helvetica, Arial, 'Nanum gothic', 'malgun gothic', sans-serif;  
text-align: left | right | center;  
text-indent: -9999px;  
line-height: 1.5 | 150% | 1.5em;  
letter-spacing: 1px;  
color: #ccc;
```

## 기본 속성 빠르게 보기

### 상자

```
width: 100px | 50%;  
height: 100px | 50%;  
padding: 10px;  
margin: 10px;  
border: 1px solid red;  
outline: 1px solid blue;  
background: #ddd;  
overflow: hiddne | visible | scroll | auto;
```

## 기본 속성 빠르게 보기

### 레이아웃

```
display: block | inline | inline-block;  
float: none | left | right;  
position: relative | absolute;
```

## shorthand, 단축된 값

- 분리 작성해야 할 CSS를 한 줄에 단축하여 설정
- 일부 속성을 빠지면? 그 자리에 디폴트값이 적용된다
- → 선택적으로 덮어쓰도록 설계해야

## border

```
.box {  
  border-width: 1px; // 굵기  
  border-style: solid; // 형태  
  border-color: #000; // 색상  
}
```

```
.box {border: 1px solid #000;}
```

## 선택적 오버라이드

```
.box {  
  border: 3px solid black;  
}  
  
.box.focus {  
  border-style: dotted; // style만 오버라이드  
}  
  
.box.highlight {  
  border-color: red; // color만 오버라이드  
}
```



## background

```
section {  
  background-color: #000; // 색상  
  background-image: url(images/bg.jpg); // 이미지 경로  
  background-repeat: no-repeat; // 반복  
  background-position: 50% 50%; // 배경 위치  
}  
  
section {background: #fff url(images/bg.jpg) 50% 50% no-repeat;}
```

## font

- 기본 : `font: font-size font-family`
- 추가키워드는 앞에 `font: bold italic`
- 줄간격(`line-height`) : `font: 100%/1.5`

```
h1 {  
  font-style: italic; // 스타일  
  text-decoration: underline; // 밑줄  
  font-weight: bold; // 굵기  
  font-size: 14px; // 크기  
  line-height: 1.2; // 줄간격  
  font-family: 'Nanum gothic', sans-serif; // 글꼴  
}
```

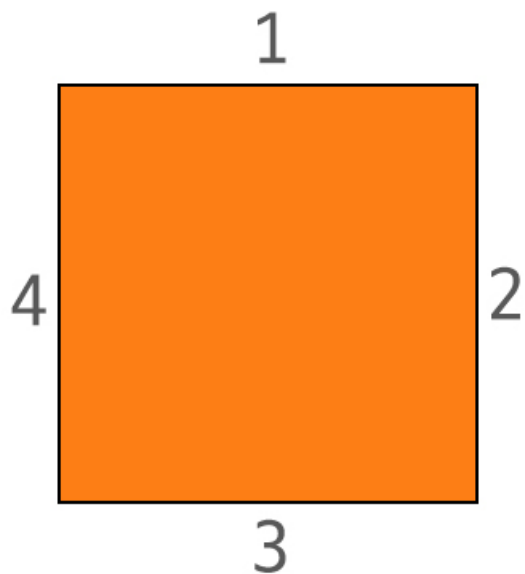
```
h1 {font: italic bold 14px/1.2 'Nanum gothic', sans-serif;}
```

### line-height, 1.5 vs 1em

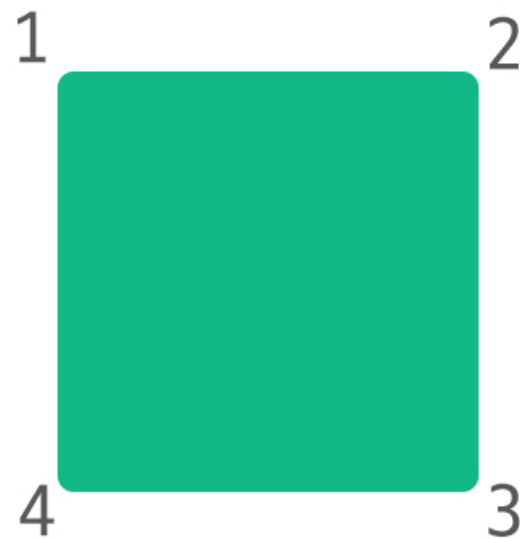
계산된 값이 자식으로 전달되는데, `em` 은 다시 계산하지 않는다.

## shorthand, 상 우 하 좌

- 4개가 한 세트
- `margin` , `padding` , `border-width` , `border-radius`



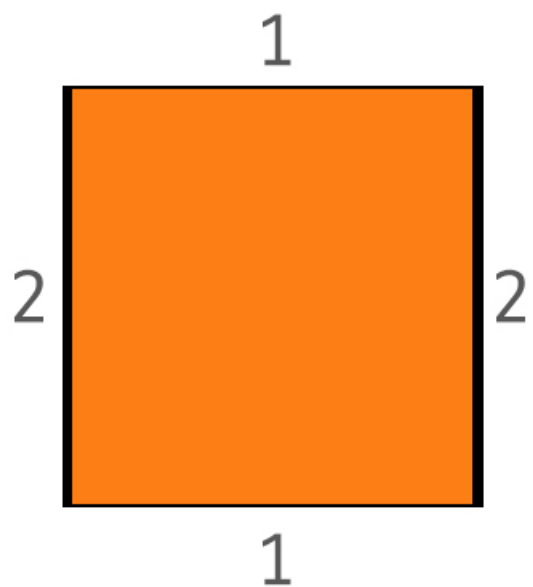
```
padding:1px 1px 1px 1px;  
border-width:1px 1px 1px 1px;
```



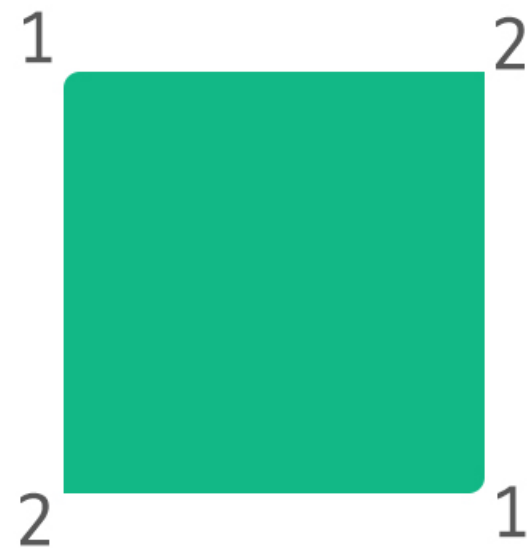
```
border-radius:10px 10px 10px 10px;
```

## shorthand, 상하 좌우

- 2개가 한 세트



```
padding:1px 5px;  
border-width:1px 5px;
```



```
border-radius:10px 0;
```

## margin, padding

```
div {  
  margin-top: 10px;  
  margin-right: 5px;  
  margin-bottom: 10px;  
  margin-left: 5px;  
  padding-top: 10px;  
  padding-right: 5px;  
  padding-bottom: 10px;  
  padding-left: 5px;  
}  
  
div {margin: 10px 5px 10px 5px; padding: 10px 5px 10px 5px;}  
div {margin: 10px 5px; padding: 10px 5px;}
```

## border-radius

```
div {  
  border-top-left-radius:10px;  
  border-top-right-radius:20px;  
  border-bottom-right-radius:30px;  
  border-bottom-left-radius:40px;  
}
```

```
div {border-radius:10px 20px 30px 40px;}
```

## 그밖에

transition, animation, transform...

# 태그와 관계 짓기. 셀렉터(선택자)

## 관계 선택자

### 타입 선택자(type selector)

- 기본 선택자
- 태그 엘리먼트

```
div {font-size:20px;}
```

### 하위 선택자 (descendant selector)

- 스페이스 로 구분
- 하위의 모든 엘리먼트

```
div p {font-size:14px;}  
.box .heading {color: black}
```

## 자식 선택자(child selector)

- 바로 아래있는 자식 엘리먼트만

```
ul > li {font-size:16px;}  
ul > li > a {color:orange;}
```

## 인접 형제 선택자 (adjacent sibling selector)

- 바로 뒤에 따라오는(병렬의) 엘리먼트

```
a + a {border-left:2px dotted red;}  
ul li + li {border-top:1px solid #ccc;}
```



## 일반 형제 선택자(**general sibling selector**)

- 같은 부모를 공유하는 모든 엘리먼트

```
<header>  
  <h1>큰 제목</h1>  
  <h2>작은 제목 첫 번째</h2>  
  <h2>작은 제목 두 번째</h2>  
</header>
```

```
h1 ~ h2 {font-size: 16px;}
```

## 속성 선택자

- 엘리먼트의 속성을 기준으로 선택

```
<a href="http://google.com" target="_blank" title="go google">구글로 이동하기</a>
```

```
a[title] {color:blue;} // title 속성이 있는 링크
a[target=_blank] {text-decoration:underline;} // 외부연결(_blank) 링크
a[title~=google] {color:black;} // 'google'이 포함된 링크
a[title*=gle] {color:black;} // '-gle'이 포함된 링크

a[href^=https] {font-size:16px;} // 암호화(SSL)된 링크
a[href^=mailto] {font-size:16px;} // 메일링크
a[href$=.pdf] {font-style:italic;} // pdf파일 다운로드
a[href$=.zip] {font-style:italic;} // zip파일 다운로드
```

- \* 포함된
- ^ 시작하는
- \$ 끝나는
- ~ 포함된(단어 단위)

## 가상 클래스(pseudo-class)선택자

**<a>**

- 링크의 현재 상태
- **LoVe/HaTe** :Link - :Visited - :Hover - :Active 의 순서로 정의

**:after, :before**

- 존재하지 않는 가상의 엘리먼트를 선택

```
.box {position:relative}
.box:before {
  content:" ";
  position:absolute;
  top:0;
  right:0;
  width:20px;
  height:20px;
  background:red;
}
```

```
a[href]:after {content: "[" attr(href) "]; }
```

## 구조적 가상 클래스 선택자

**:first-child**

- 첫번째 엘리먼트

**:last-child**

- 마지막 엘리먼트

**:nth-child(n), :nth-child(2n)**

- n번째 엘리먼트

**:nth-last-child(n)**

- 마지막에서 n번째 엘리먼트

## 구조적 가상 클래스 선택자

```
<ul>  
  <li>이름</li>  
  <li>나이</li>  
  <li>성별</li>  
  <li>소개</li>  
</ul>
```

```
ul li:first-child {}  
ul li:last-child {}  
  
ul li:nth-child(3) {}  
ul li:nth-child(2n) {}  
ul li:nth-child(2n-1) {}  
  
ul li:nth-last-child(2) {}
```

# 케스케이드 지정 순서

- 케스케이드 = 우선순위
- 우선순위가 높은 것이 적용된다. 낮은 것은 무시
- 중요도, 명시도, 작성(로딩) 순서

## 선택자 중요도

1. !important가 지정된 스타일
2. 작성된 스타일
3. 브라우저 기본 스타일

## 선택자 명시도

1. `!important`
2. `id ( 100 )`
3. `class ( 10 )`
4. 태그 엘리먼트( 1 )



선택자의 점수를 계산하여 우선순위 적용

## 소스의 순서

- 중요도, 명시도가 같다면 나중에 선언된 것이 우선

# 상속

- 상속: 부모의 속성이 자식에게 전달
- 상속이 없다면 모든 엘리먼트에 대해 속성을 작성해야 함
- 상대값(%)을 사용하고자 할 때는 기준값이 됨

## 상속되지 않는 속성이 존재

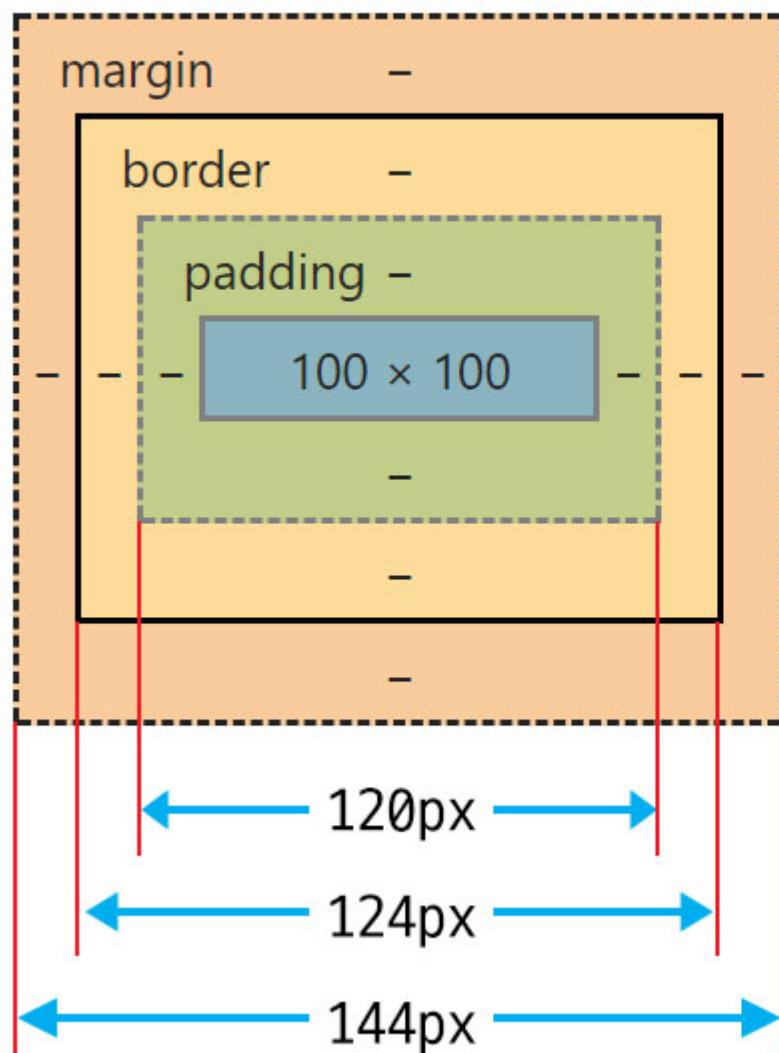
- `border` , `margin` , `background` ...

## 강제 상속

```
p {background:inherit}
```



# 박스모델 Box model



```
.box {  
  padding:10px;  
  border:2px solid red;  
  margin:10px;  
}
```

## 박스모델의 단점 보완, box-sizing

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

## outline

```
.box {outline: 1px solid red;}
```

## 문서의 흐름과 float

- HTML은 위에서 아래로 흐르는 문서
- float = 띄운다 = 흐름에서 벗어난다
- left, right, none
- 주로 레이아웃을 위해 쓰인다

## float 예제

```
<div class="tiger">
  
  <p>
    <strong>호랑이</strong>
    먹이사슬의 정점에 선 포식자로 타고난 맹수다.
  </p>
</div>
<div class="lion">
  사자 내용 들어갈 자리
</div>
```

```
.tiger {}
.tiger img {float: left; width: 30%;}
.tiger p {float: right; width: 70%;}
```

## float을 clear하는 5가지 방법

- (부모) float에 float
- (부모) overflow:hidden `overflow:hidden`
- 다음 요소, 빈(보이지 않는) 요소 넣기 `display:block; clear:both`
- :after(가상 선택자) `content:" "; display:block; clear:both;`
- (부모) display:inline-block으로 (틈 주의)

## 부모 float

```
.tiger {float:left;}  
.tiger img {float:left;}
```

## overflow:hidden

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

## 빈 요소, 다음 요소에서 clear

```
.lion {clear:both;}
```

## 가상선택자

```
.tiger {content:" "; display:block; clear:both;}
```

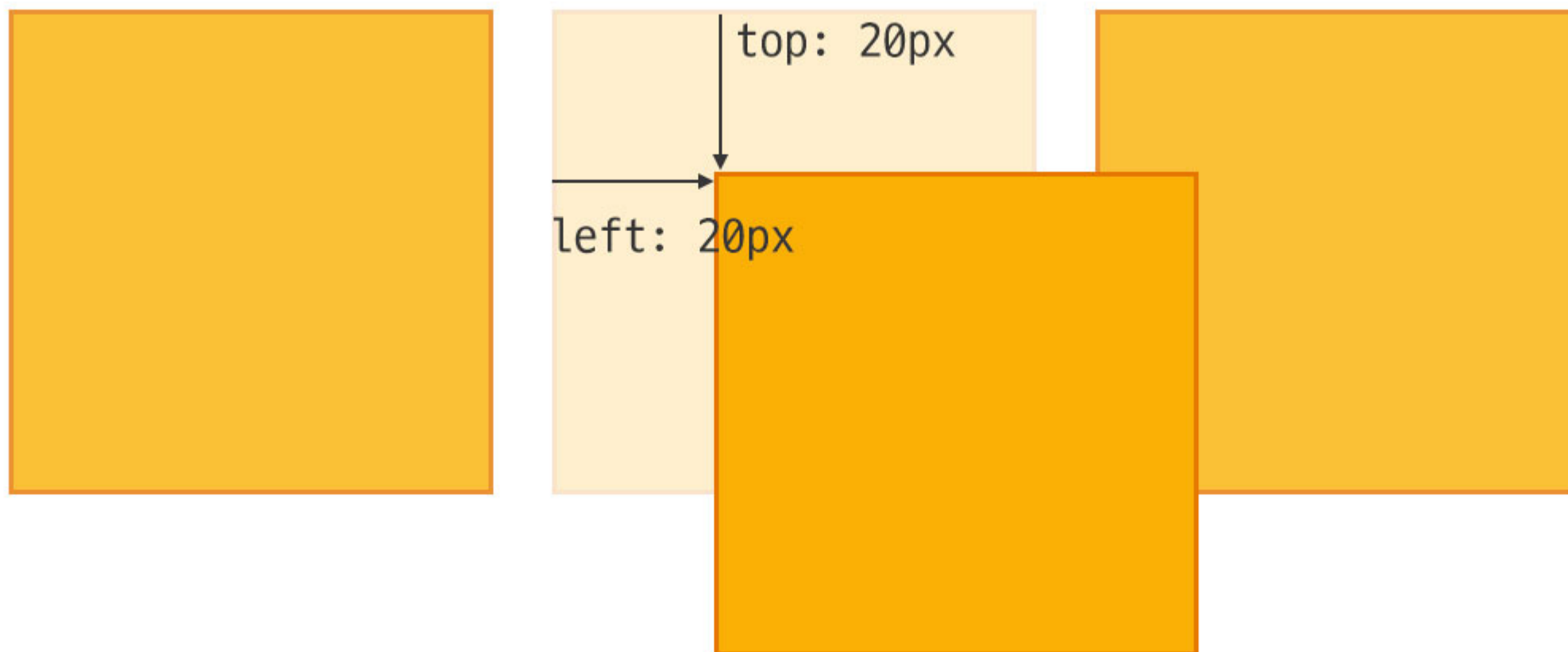
## display:inline-block

```
.tiger {display:inline-block}
```

# position

- `static` : 기본값
  - `relative` : 상대 위치. 하위 콘텐츠의 기준점이 된다.
  - `absolute` : 절대 위치. **position**값이 설정된 부모를 기준으로 한 위치
  - `fixed` : **화면(viewport)**를 기준으로 한 위치
- 
- `top, right, bottom, left` 를 수치(px, %)로 설정할 수 있다.
  - `auto` 로 리셋할 수 있다.

**position: relative**



```
{position:relative; top:20px; left:20px;}
```

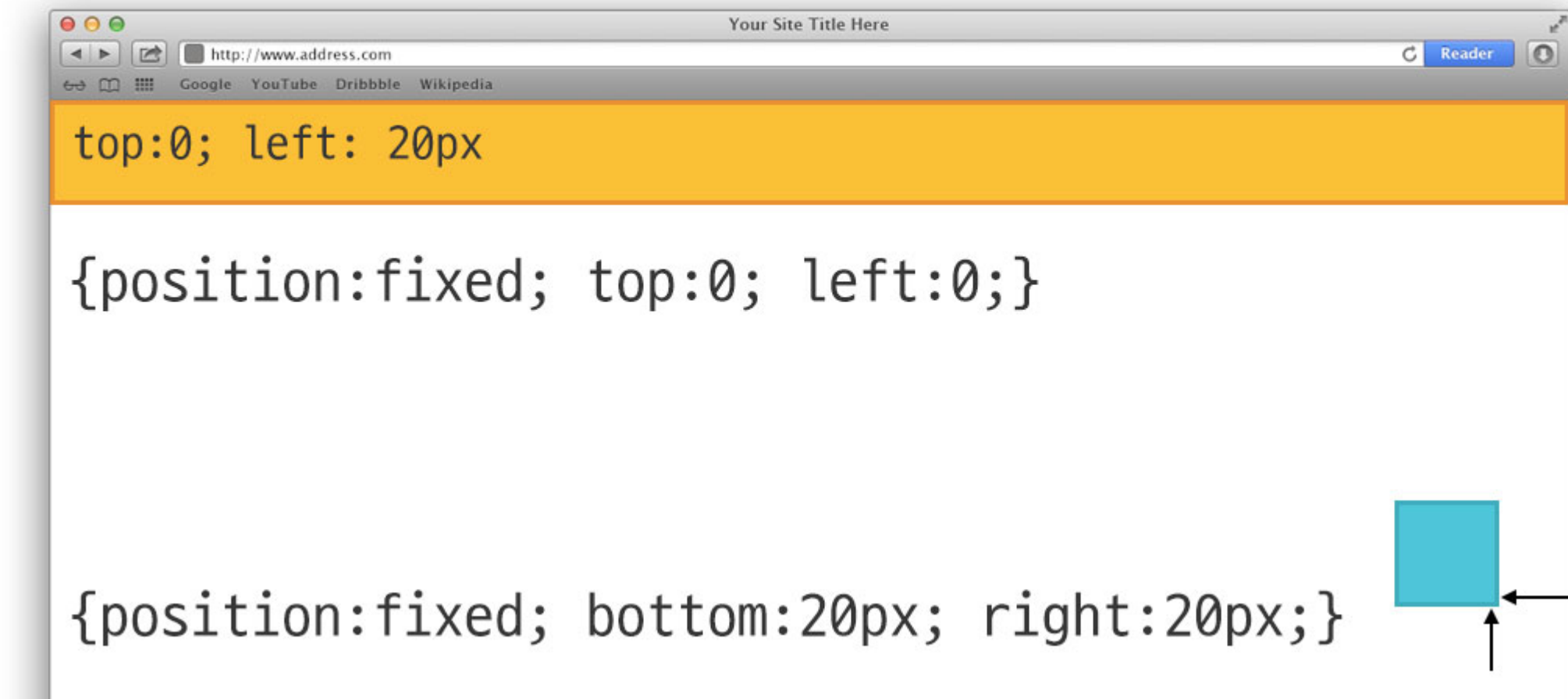


**position: absolute**



```
{position:absolute; top:20px; left:20px;}
```

## position: fixed



## z-index

- `z-index` : 3차원 공간에서의 레이어 순서
- `absolute` , `fixed` 값을 가진 엘리먼트에 `z-index` 를 선언
- `stacking`

```
<div class="box">  
  <div class="title">제목</div>  
  <div class="content">내용</div>  
</div>
```

```
.box {position: relative;}  
.box .title {position: absolute;top:0;left: 0;z-index: 0;}  
.box .content {position: absolute;top: 10px;left: 0;z-index: 1}
```