TechMeetup

UI개발 3팀 양주희

BEM

실무 사용 경험기

UI개발 3팀 양주희

B E M

실무 사용 경험기

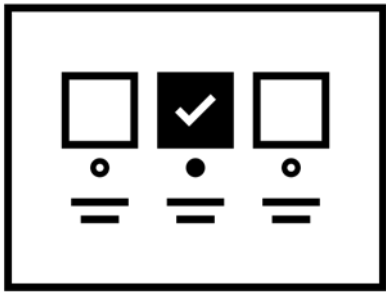
1. BEM이란?
2. 시행착오
3. 장단점

취향대로? 습관처럼?

클래스. 어떻게 정하시나요?

이마저도 곧 공통이 되리니…

UI 형태를 따르자.



작명센스 고갈

작명센스 고갈

네이밍 컨벤션

필요하지 않나


BEM

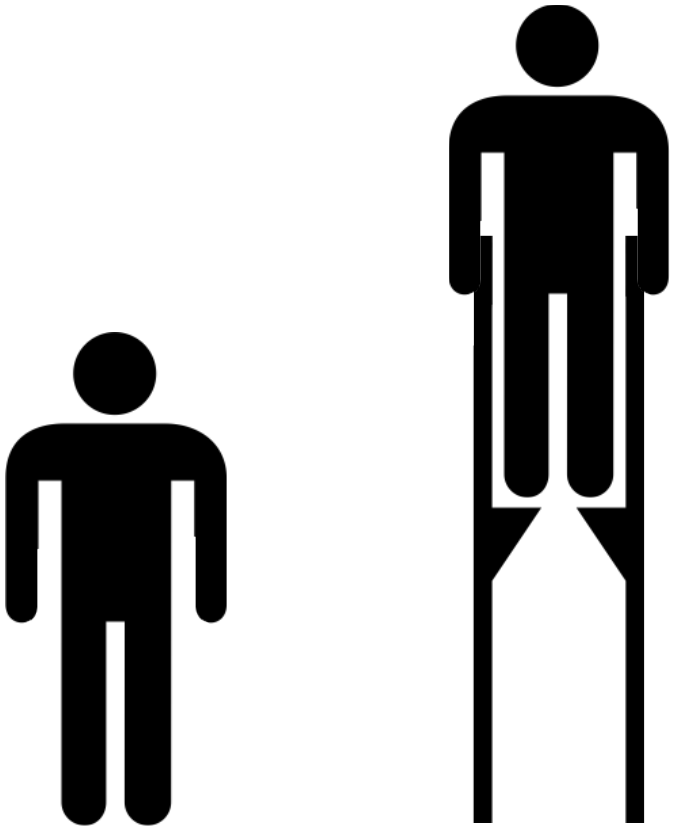
Block

Block

Element

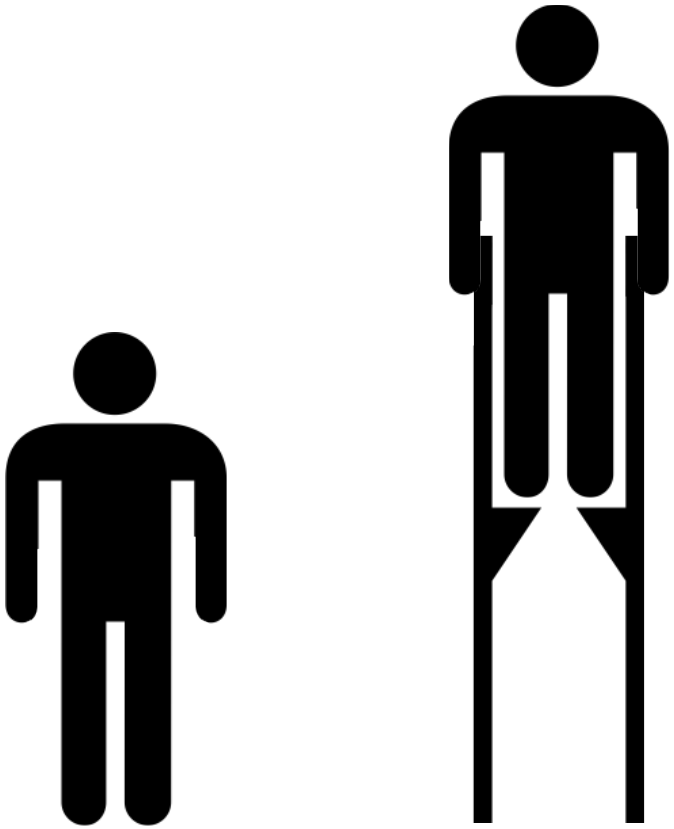
Block
Element
Modifier

Block__**E**lement--**M**odifier



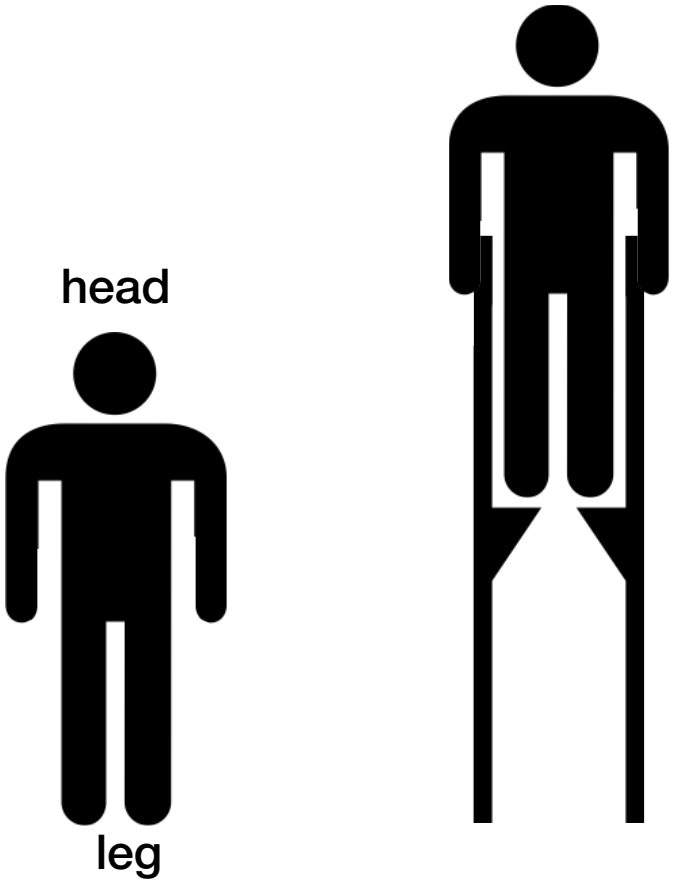
일반적인 방식

person



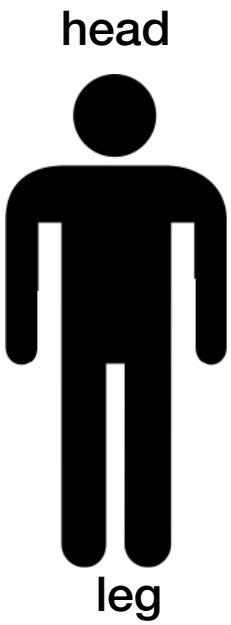
일반적인 방식

person

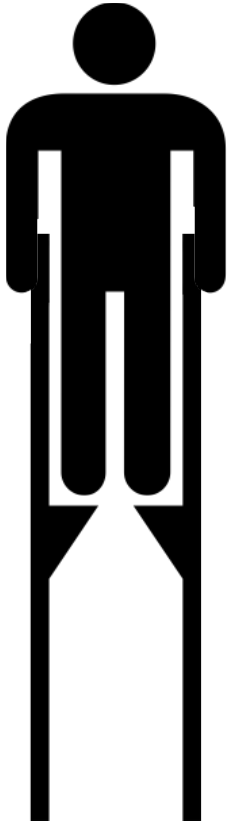


일반적인 방식

person

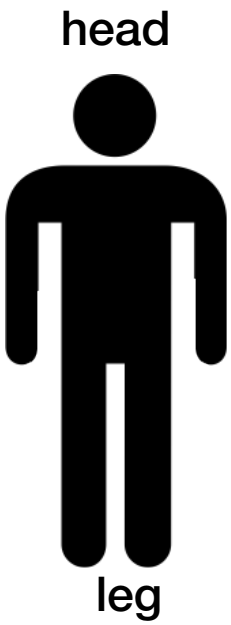


tall person

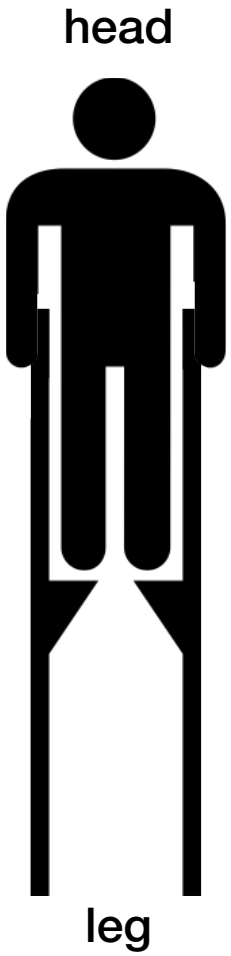


일반적인 방식

person

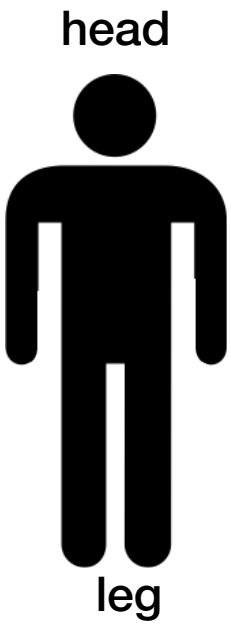


tall person

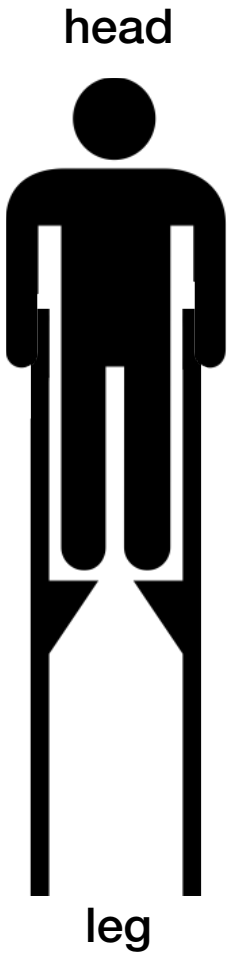


일반적인 방식

person

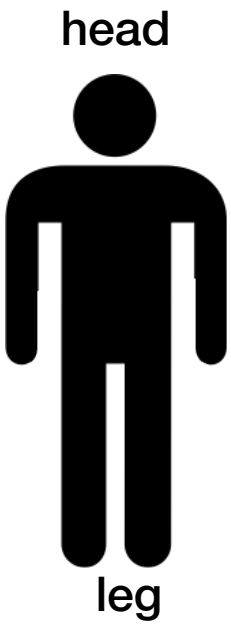


tall person

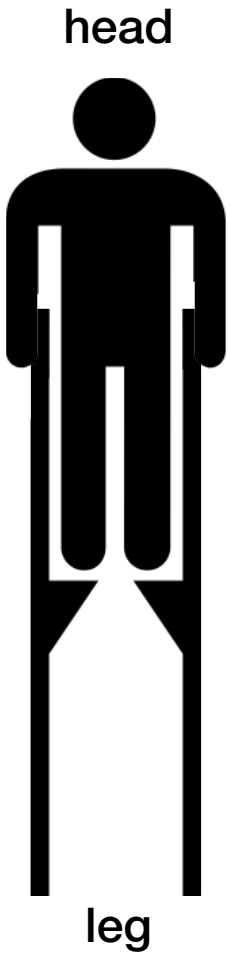



```
// html
<div class=“person">
  <span class="head"></span>
  <span class=“leg"></span>
</div>

<div class=“person tall">
  <span class="head"></span>
  <span class=“leg"></span>
</div>
```

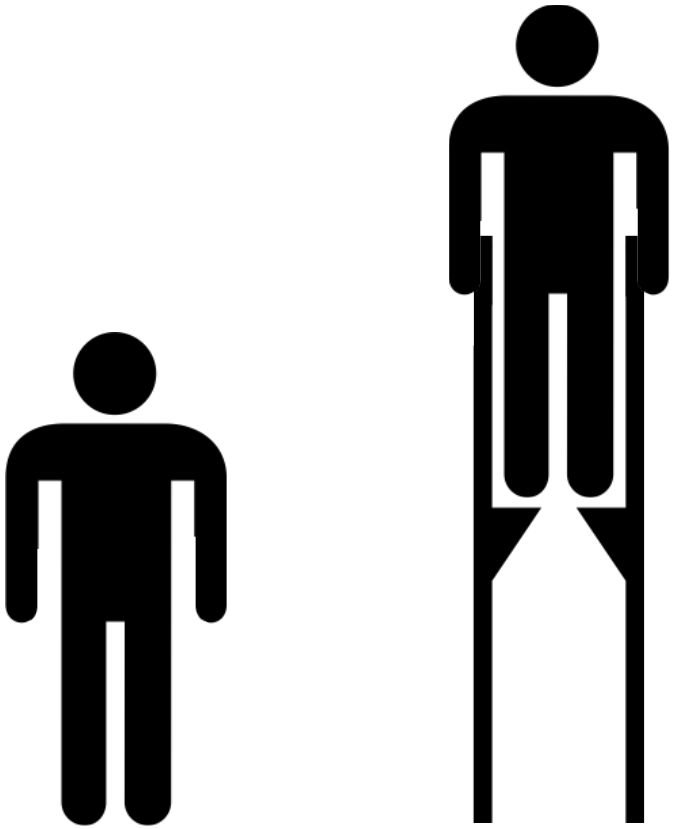
person



tall person

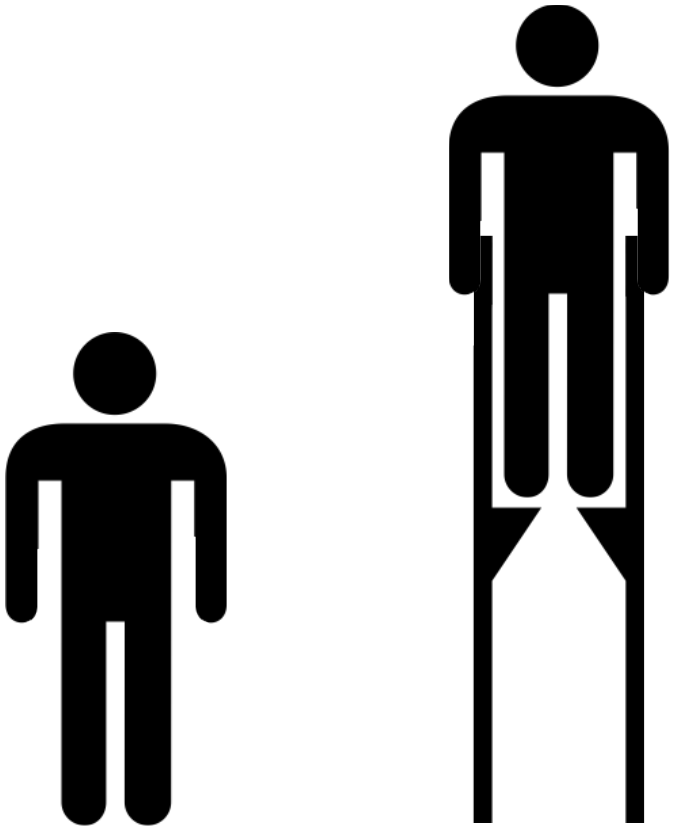



```
// html
<div class=“person">
  <span class="head"></span>
  <span class=“leg"></span>
</div>

<div class=“person tall">
  <span class="head"></span>
  <span class=“leg"></span>
</div>


 // css
 .person {}
 .person .head {}
 .person .leg {}

 .person.tall {}
 .person.tall .head {}
 .person.tall .leg {}
```



BEM 방식

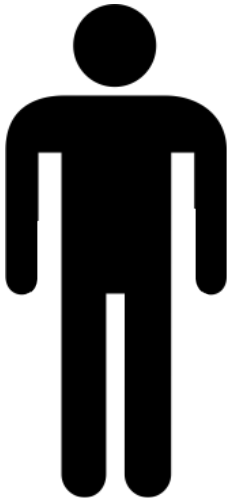
person



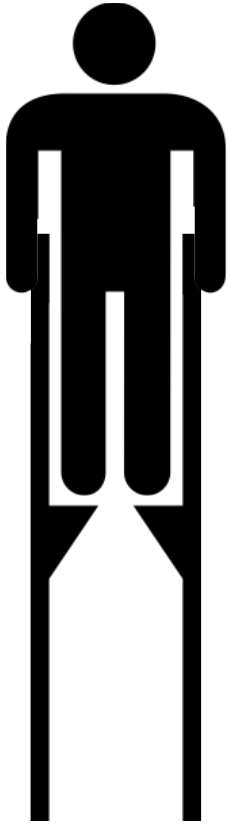
BEM 방식

person

person's head



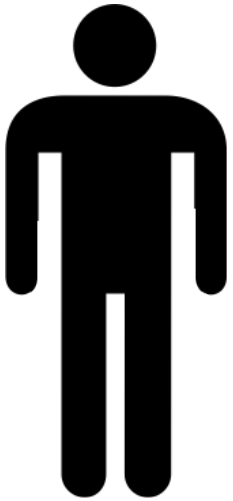
person's leg



BEM 방식

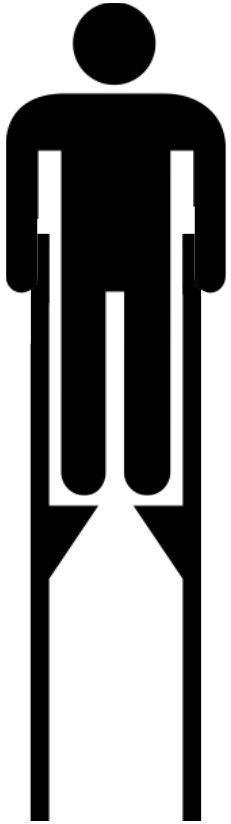
person

person’s head



person’s leg

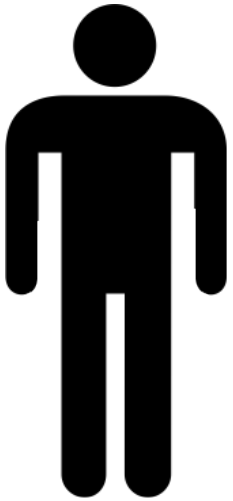
tall person



BEM 방식

person

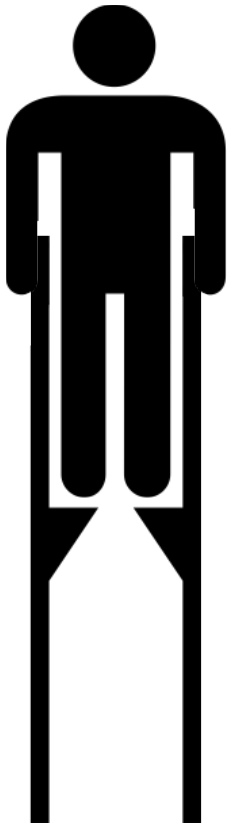
person's head



person's leg

tall person

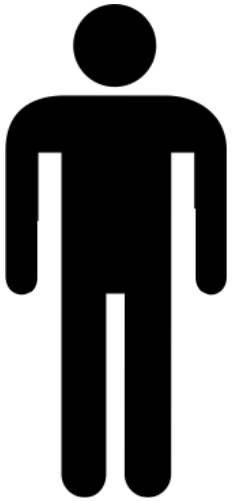
person's head



BEM 방식

person

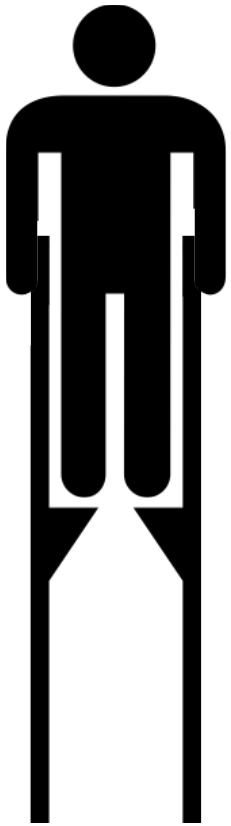
person's head



person's leg

tall person

person's head

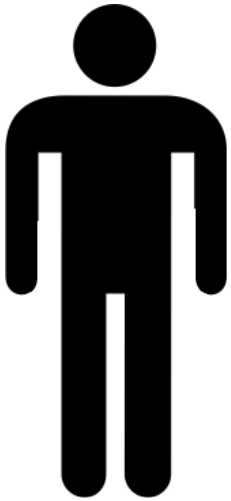


person's long leg

BEM 방식

person

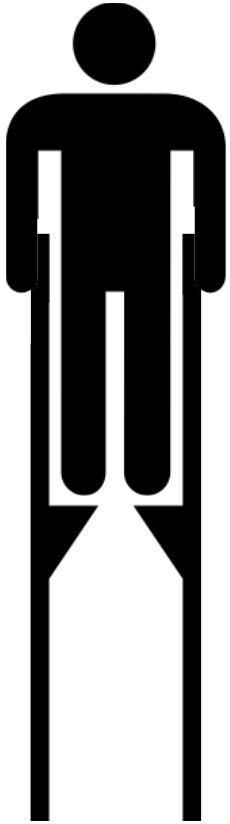
person's head



person's leg

tall person

person's head

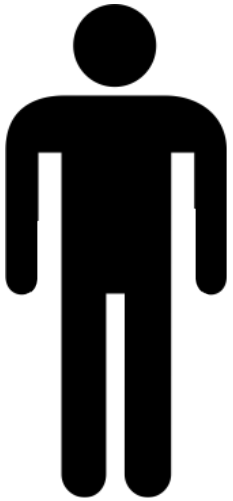


person's long leg

```
// html
<div class="person">
  <span class="person__head"></span>
  <span class="person__leg"></span>
</div>

<div class="person--tall">
  <span class="person__head"></span>
  <span class="person__leg--long"></span>
</div>
```

person

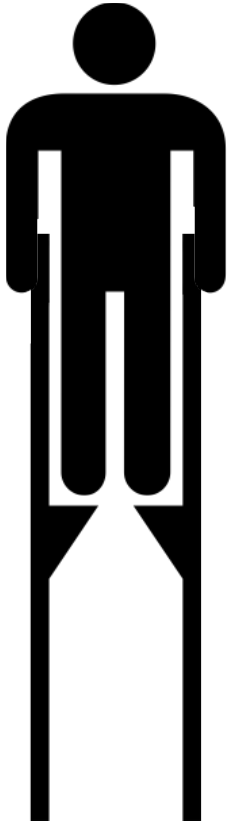
person's head



person's leg

tall person

person's head



person's long leg

```
// html
<div class="person">
  <span class="person__head"></span>
  <span class="person__leg"></span>
</div>

<div class="person--tall">
  <span class="person__head"></span>
  <span class="person__leg--long"></span>
</div>
```

```
// css
.person {}
  .person__head {}
  .person__leg {}

.person--tall {}
  .person--tall .person__head {}
  .person__leg--long {}
```

클래스가 길어진다.

클래스 하나만으로
요소를 선택할 수 있다.

객체지향적

딱 여기까지만 알고

일단 써보기로 했습니다.

Sass의 중첩기능 

- ✔ 가독성이 좋아짐
- ✔ 블록과 속성을 기준으로 자동 들여쓰기가 된다.
- ✔ 클래스 풀네임으로 검색하기 어렵다...

```scss
// scss
.person {
  &__head {}
  &__leg {}

  &--tall {
    &__leg {}
  }
}
```

```css
// css
.person {}
  .person__head {}
  .person__leg {}

.person--tall {}
  .person--tall__leg {}
```

확신이 안선다

확신이 안선다

영쩜오엠디 더

필요하지 않나

적응하기 힘든 **클래스 길이**

적응하기 힘든 **클래스 길이**

블록은 기준은 어디까지인가?

적응하기 힘든 **클래스 길이**

블록은 기준은 어디까지인가?

스크립트 처리를 위한 **클래스 제어**

적응하기 힘든 **클래스 길이**

블록은 기준은 어디까지인가?

스크립트 처리를 위한 **클래스 제어**

자손 클래스는 쓰면 안되나?

너무 긴 클래스

너무 긴 클래스

```
▼<div class="footer__sns-share-indicator__wrapper">
   <span class="ico_bar"></span>
  ▶<div class="footer__sns-share-indicator">…</div>
  </div>
▼<ul class="footer__sns-share-list">
  ▶<li class="footer__sns-share-list-item footer__sns-share-list-item--facebook">…</li>
  ▶<li class="footer__sns-share-list-item footer__sns-share-list-item--twitter">…</li>
  </ul>
```

너무 긴 클래스

```
▼<div class="footer__sns-share-indicator__wrapper">
   <span class="ico_bar"></span>
  ▶<div class="footer__sns-share-indicator">…</div>
  </div>
▼<ul class="footer__sns-share-list">
  ▶<li class="footer__sns-share-list-item footer__sns-share-list-item--facebook">…</li>
  ▶<li class="footer__sns-share-list-item footer__sns-share-list-item--twitter">…</li>
  </ul>
```

너무 긴 클래스

```
.footer__sns-share-indicator__wrapper
```

너무 긴 클래스

※ 아주 극단적인 예 입니다.

```
.footer__sns-share-indicator__wrapper
```

너무 긴 클래스

완전체형 / 축약형

어떤 것을 선호하시나요?

너무 긴 클래스

완전체형 / 축약형

어떤 것을 선호하시나요?

.wrapper	.wrp
.button	.btn
.content	.cnt
.schedule	.sch
.calendar	.cal

너무 긴 클래스

완전체형 / 축약형

어떤 것을 선호하시나요?

.button__wrapper	.button__wrp	.btn__wrp	.btn_wrp
.button__content	.btn__content	.btn__cnt	.btn_content
.schedule__calendar	.schedule__cal	.sch__cal	.sch_cal

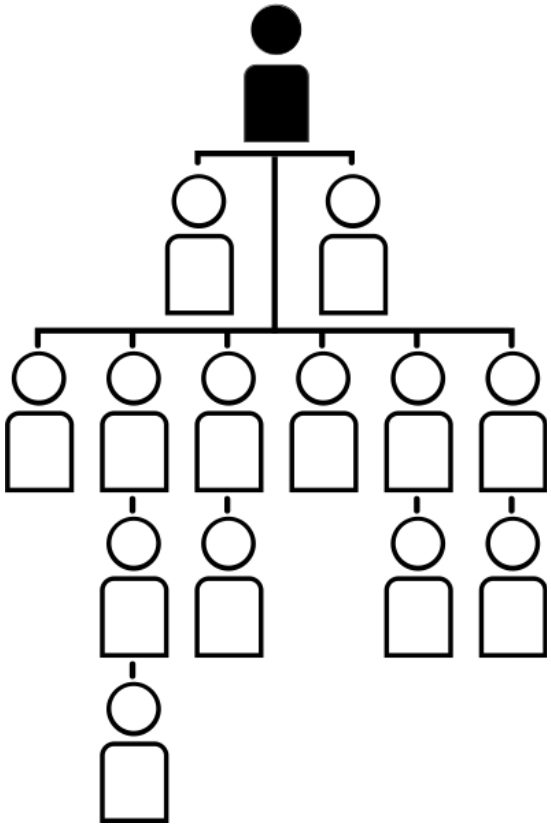
너무 긴 클래스

가독성이 떨어지지 않을 정도로

“적 당 히”

블록의 기준

블록의 기준



블록의 기준

- **블록**에는

  부모 블록이 있고

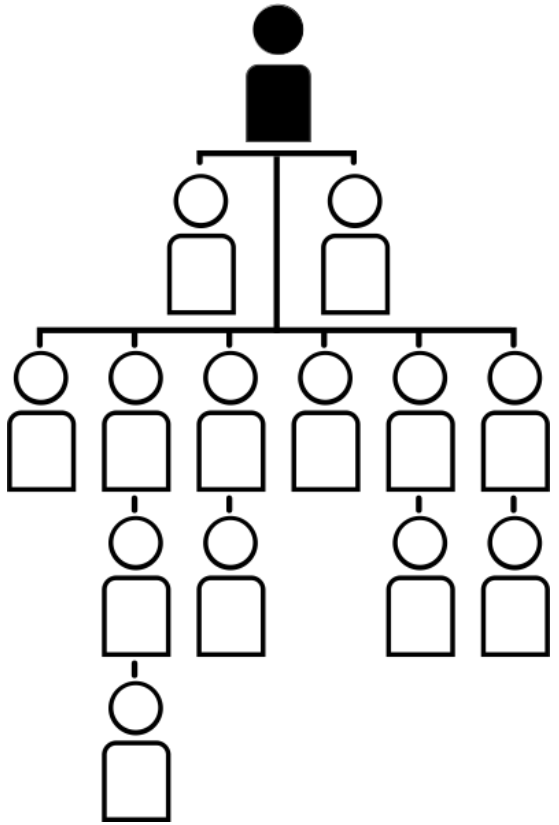
  할부모 블록이 있고

  증조할부모 블록이 있고…

- **요소도**

  나름의 블록이고

  자손들도 있을 테고…



블록의 기준

- **블록**에는

  부모 블록이 있고

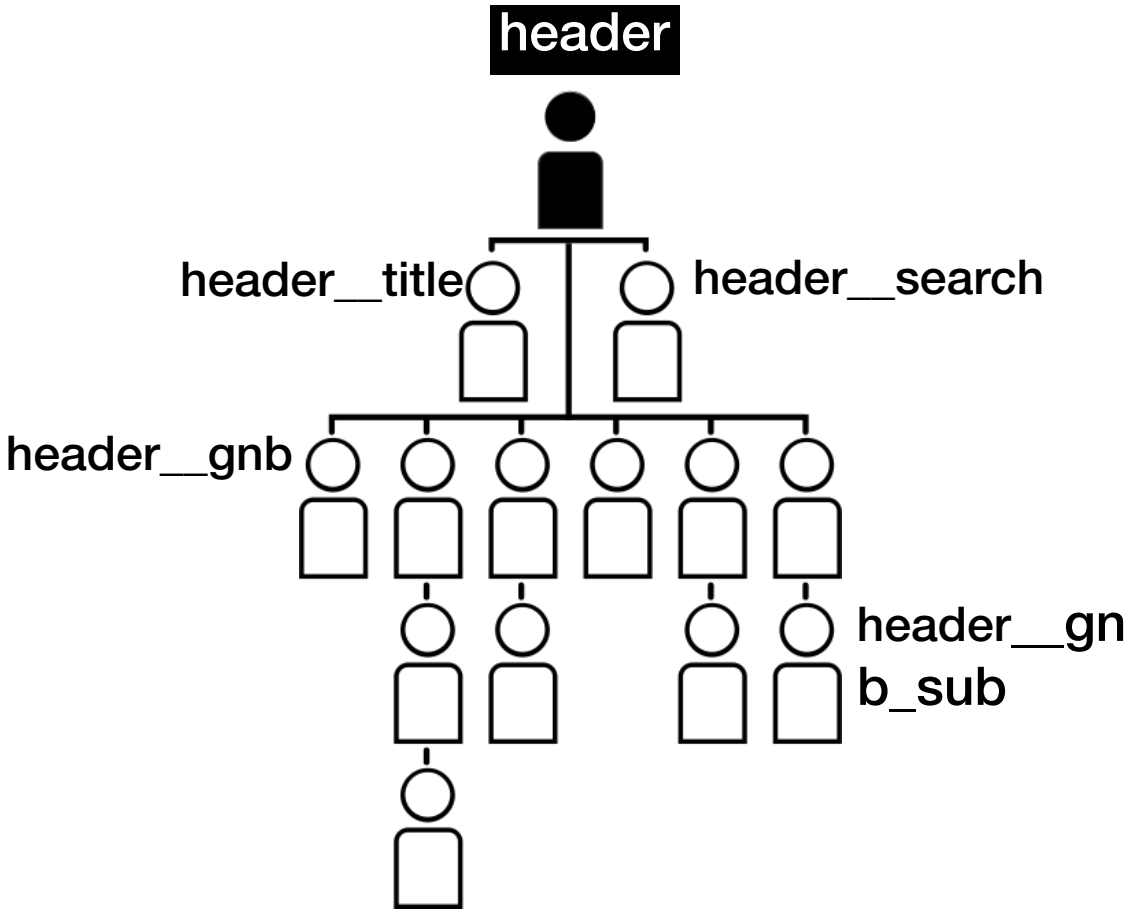
  할부모 블록이 있고

  증조할부모 블록이 있고…

- **요소도**

  나름의 블록이고

  자손들도 있을 테고…



블록의 기준

- **블록**에는

  부모 블록이 있고

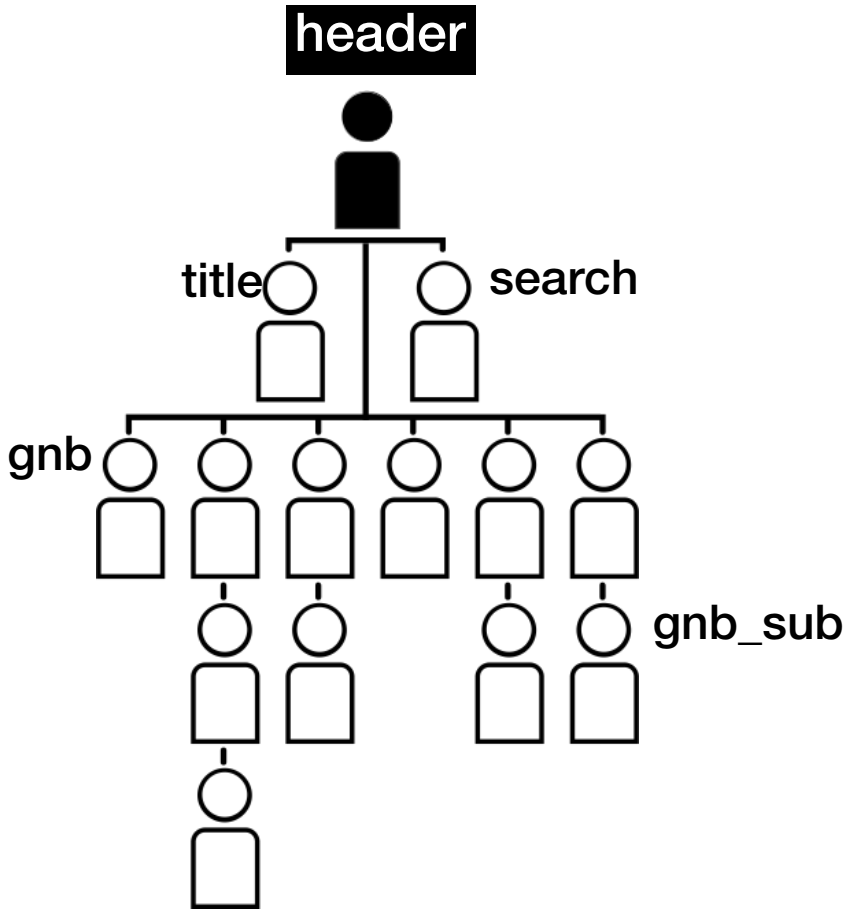
  할부모 블록이 있고

  증조할부모 블록이 있고…

- **요소도**

  나름의 블록이고

  자손들도 있을 테고…



블록의 기준

최상단의 .header를 기준으로 잡은 경우

```
<div class="header">

  <div class="header__title">
    <h1 class="header__title_txt">타이틀</h1>
    <p class="header__title_subtxt">서브타이틀</p>
  </div>

  <div class="header__search">
    <input class="header__search_input" type="search">
    <button class="header__search_btn">
        <span class="header__search_txt">검색
          <span class="“header__search_txt_ico”"></span>
        </span>
    </button>
  </div>

</div>
```

블록의 기준

UI 덩어리로 나눈 경우

```
<div class="header">

  <div class="title">
    <h1 class="title__txt">타이틀</h1>
    <p class="title__subtxt">서브타이틀</p>
  </div>

  <div class="search">
    <input class="search__input" type="search">
    <button class="search__btn">
        <span class="search__txt">검색
          <span class="search__ico"></span>
        </span>
    </button>
  </div>

</div>
```

블록의 기준

모든 DOM의 구조를 반영할 필요가 없다.

“2뎁스까지만”

클래스 제어

클래스 제어

.hover .on .dim

〈!-- [D] 클래스 추가해주세요. --〉

```
<li class="list__item">...</li>
```

클래스 제어

.hover .on .dim

〈!-- [D] 클래스 추가해주세요. --〉

```
// 기존 방식
<li class="list__item hover">…</li>
```

클래스 제어

.hover .on .dim

〈!-- [D] 클래스 추가해주세요. --〉

```
// 기존 방식
<li class="list__item hover">…</li>

// 속성 추가된 클래스 추가
<li class="list__item list__item--hover">…</li>
```

클래스 제어

.hover .on .dim

〈!-- [D] 클래스 추가해주세요. --〉

```
// 기존 방식
<li class="list__item hover">…</li>

// 속성 추가된 클래스 추가
<li class="list__item list__item--hover">…</li>

// 속성 추가된 클래스로 변경 (@extend)
<li class="list__item--hover">…</li>
```

위험한 @extend

안 그래도 긴 클래스

클래스 두 개 쓰긴 찝찝하니

@extend의 혜택을 누려볼까.

```
// html
<li class="list__item">...</li>
<li class="list__item--hover">...</li>


// scss
.list {
  &__item {
            height:10px;
    &--hover {
      @extend .list__item;
      color:#fff;
    }
  }
}

// css
.list__item, .list__item--hover {
  height: 10px;
}

.list__item--hover {
  color: #fff;
}
```

```
.board__btn + .board__btn, .board__btn--big + .board__btn, .apply__btn + .board__btn, .apply__btn--cancle + .board__btn, .board__btn--small + .board__btn, .myapply__btn + .board__btn, .materials__content__download + .board__btn, .partner__btn + .board__btn, .header__loginbtn + .board__btn, .media__watch-btn + .board__btn, .guide__btn + .board__btn, .magazine__apply + .board__btn, .board__btn + .board__btn--big, .board__btn--big + .board__btn--big, .apply__btn + .board__btn--big, .apply__btn--cancle + .board__btn--big, .board__btn--small + .board__btn--big, .myapply__btn + .board__btn--big, .materials__content__download + .board__btn--big, .partner__btn + .board__btn--big, .header__loginbtn + .board__btn--big, .media__watch-btn + .board__btn--big, .guide__btn + .board__btn--big, .magazine__apply + .board__btn--big, .board__btn + .apply__btn, .board__btn--big + .apply__btn, .apply__btn + .apply__btn, .apply__btn--cancle + .apply__btn, .board__btn--small + .apply__btn, .myapply__btn + .apply__btn, .materials__content__download + .apply__btn, .partner__btn + .apply__btn, .header__loginbtn + .apply__btn, .media__watch-btn + .apply__btn, .guide__btn + .apply__btn, .magazine__apply + .apply__btn, .board__btn + .apply__btn--cancle, .board__btn--big + .apply__btn--cancle, .apply__btn + .apply__btn--cancle, .apply__btn--cancle + .apply__btn--cancle, .board__btn--small + .apply__btn--cancle, .myapply__btn + .apply__btn--cancle, .materials__content__download + .apply__btn--cancle, .partner__btn + .apply__btn--cancle, .header__loginbtn + .apply__btn--cancle, .media__watch-btn + .apply__btn--cancle, .guide__btn + .apply__btn--cancle, .magazine__apply + .apply__btn--cancle, .board__btn + .board__btn--small, .board__btn--big + .board__btn--small, .apply__btn + .board__btn--small, .apply__btn--cancle + .board__btn--small, .board__btn--small + .board__btn--small, .myapply__btn + .board__btn--small, .materials__content__download + .board__btn--small, .partner__btn + .board__btn--small, .header__loginbtn + .board__btn--small, .media__watch-btn + .board__btn--small, .guide__btn + .board__btn--small, .magazine__apply + .board__btn--small, .board__btn + .myapply__btn, .board__btn--big + .myapply__btn, .apply__btn + .myapply__btn, .apply__btn--cancle + .myapply__btn, .board__btn--small + .myapply__btn, .myapply__btn + .myapply__btn, .materials__content__download + .myapply__btn, .partner__btn + .myapply__btn, .header__loginbtn + .myapply__btn, .media__watch-btn + .myapply__btn, .guide__btn + .myapply__btn, .magazine__apply + .myapply__btn, .board__btn + .materials__content__download, .board__btn--big + .materials__content__download, .apply__btn + .materials__content__download, .apply__btn--cancle + .materials__content__download, .board__btn--small + .materials__content__download, .myapply__btn + .materials__content__download, .materials__content__download + .materials__content__download, .partner__btn + .materials__content__download, .header__loginbtn + .materials__content__download, .media__watch-btn + .materials__content__download, .guide__btn + .materials__content__download, .magazine__apply + .materials__content__download, .board__btn + .partner__btn, .board__btn--big + .partner__btn, .apply__btn + .partner__btn, .apply__btn--cancle + .partner__btn, .board__btn--small + .partner__btn, .myapply__btn + .partner__btn, .materials__content__download + .partner__btn, .partner__btn + .partner__btn, .header__loginbtn + .partner__btn, .media__watch-btn + .partner__btn, .guide__btn + .partner__btn, .magazine__apply + .partner__btn, .board__btn + .header__loginbtn, .board__btn--big + .header__loginbtn, .apply__btn + .header__loginbtn, .apply__btn--cancle + .header__loginbtn, .board__btn--small + .header__loginbtn, .myapply__btn + .header__loginbtn, .materials__content__download + .header__loginbtn, .partner__btn + .header__loginbtn, .header__loginbtn + .header__loginbtn, .media__watch-btn + .header__loginbtn, .guide__btn + .header__loginbtn, .magazine__apply + .header__loginbtn, .board__btn + .media__watch-btn, .board__btn--big + .media__watch-btn, .apply__btn + .media__watch-btn, .apply__btn--cancle + .media__watch-btn, .board__btn--small + .media__watch-btn, .myapply__btn + .media__watch-btn, .materials__content__download + .media__watch-btn, .partner__btn + .media__watch-btn, .header__loginbtn + .media__watch-btn, .media__watch-btn + .media__watch-btn, .guide__btn + .media__watch-btn, .magazine__apply + .media__watch-btn, .board__btn + .guide__btn, .board__btn--big + .guide__btn, .apply__btn + .guide__btn, .apply__btn--cancle + .guide__btn, .board__btn--small + .guide__btn, .myapply__btn + .guide__btn, .materials__content__download + .guide__btn, .partner__btn + .guide__btn, .header__loginbtn + .guide__btn, .media__watch-btn + .guide__btn, .guide__btn + .guide__btn, .magazine__apply + .guide__btn, .board__btn + .magazine__apply, .board__btn--big + .magazine__apply, .apply__btn + .magazine__apply, .apply__btn--cancle + .magazine__apply, .board__btn--small + .magazine__apply, .myapply__btn + .magazine__apply, .materials__content__download + .magazine__apply, .partner__btn + .magazine__apply, .header__loginbtn + .magazine__apply, .media__watch-btn + .magazine__apply, .guide__btn + .magazine__apply, .magazine__apply + .magazine__apply { margin-left: 10px; }
```

클래스 제어

.hover .on .dim

〈!-- [D] 클래스 추가해주세요. --〉

```
// 기존 방식
<li class="list__item hover">…</li>

// 속성 추가된 클래스 추가
<li class="list__item list__item--hover">…</li>

// 속성 추가된 클래스로 변경 (@extend)
<li class="list__item--hover">…</li>
```

클래스 제어

.hover .on .dim

〈!-- [D] 클래스 추가해주세요. --〉

```
// 기존 방식
<li class="list__item hover">…</li>

// 속성 추가된 클래스 추가
<li class="list__item list__item--hover">…</li>
```

클래스 제어

.hover

```
<li class="list__item hover">…</li>
```

&--hover

```
<li class="list__item list__item--hover">…</li>
```

클래스 제어

.hover

<li class="list__item hover">…

: 공통의 속성일 경우

&--hover

<li class="list__item list__item--hover">…

클래스 제어

.hover

<li class="list__item hover">…

: 공통의 속성일 경우

&--hover

<li class="list__item list__item--hover">…

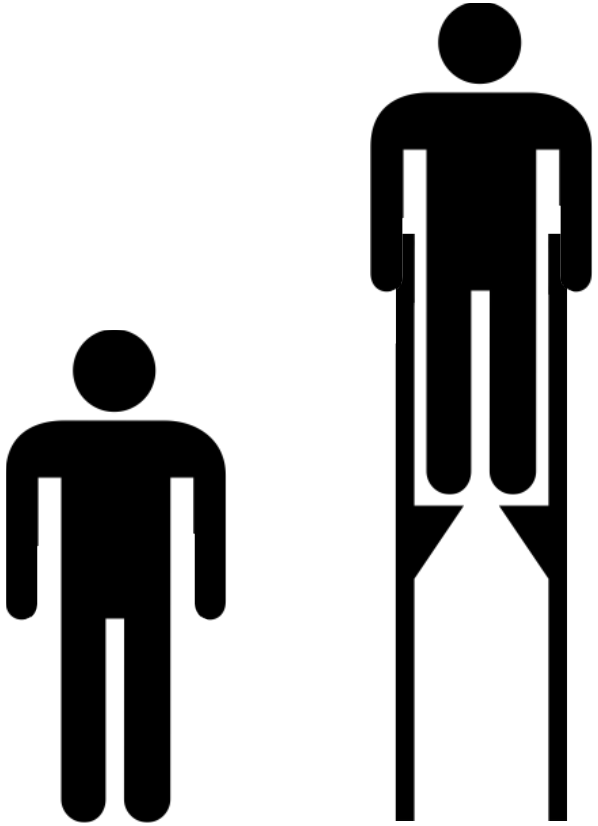
: 이 요소에만 해당할 경우

그때 그때

“상황에 따라”

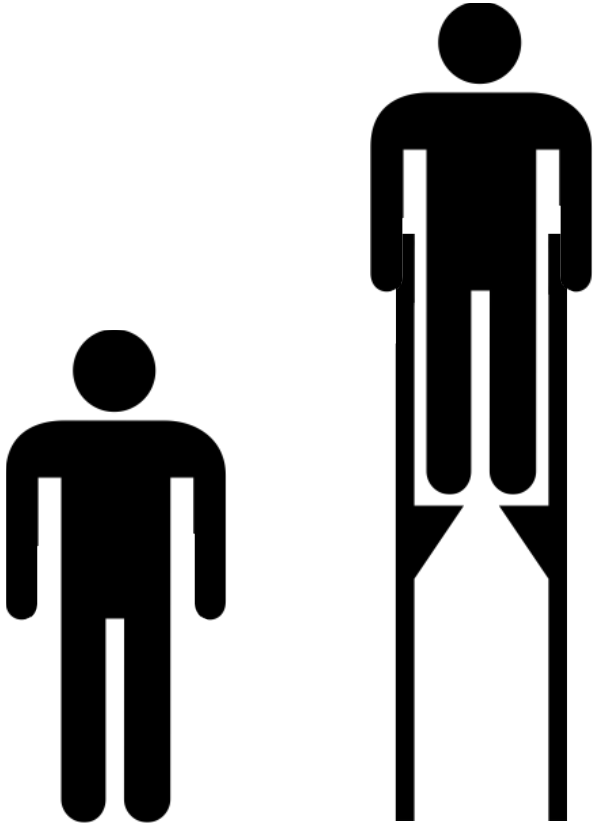
자손 선택자 금지?

자손 선택자 금지?



자손 선택자 금지?

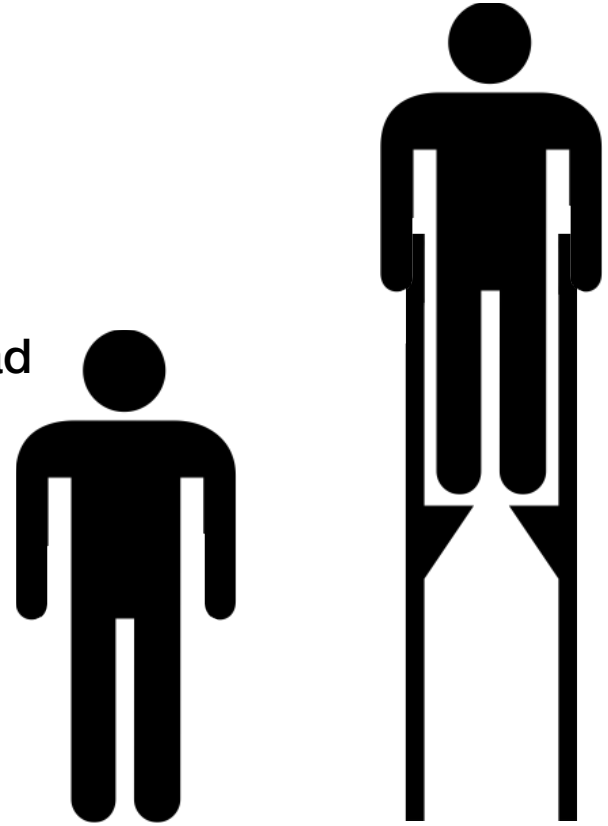
.person



자손 선택자 금지?

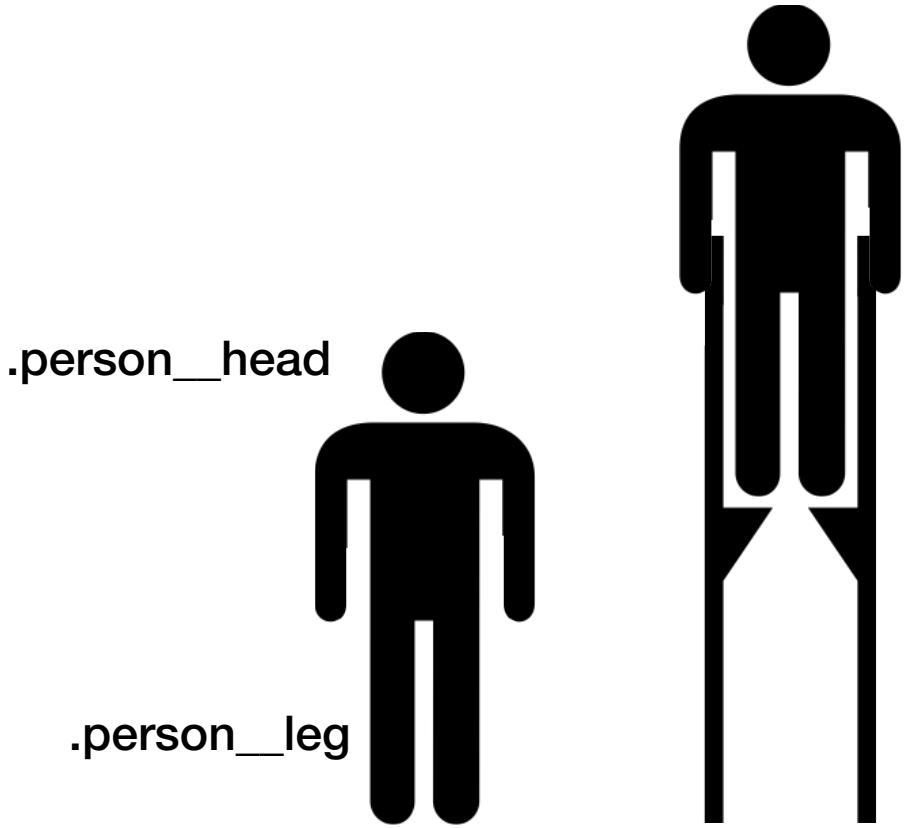
.person

.person__head



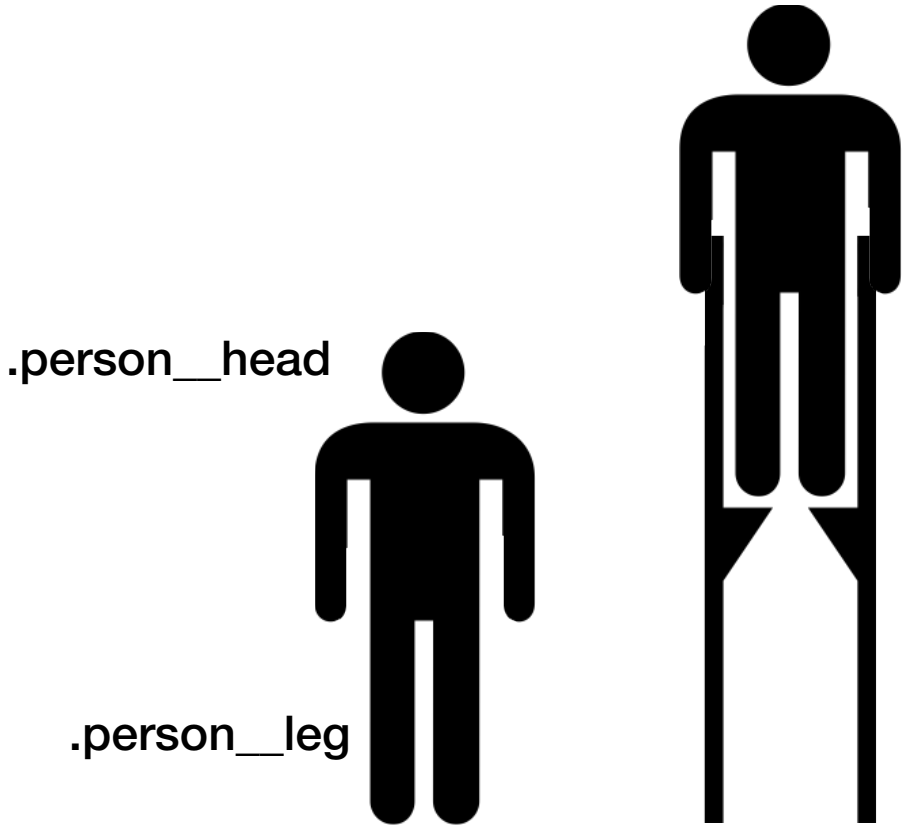
자손 선택자 금지?

.person



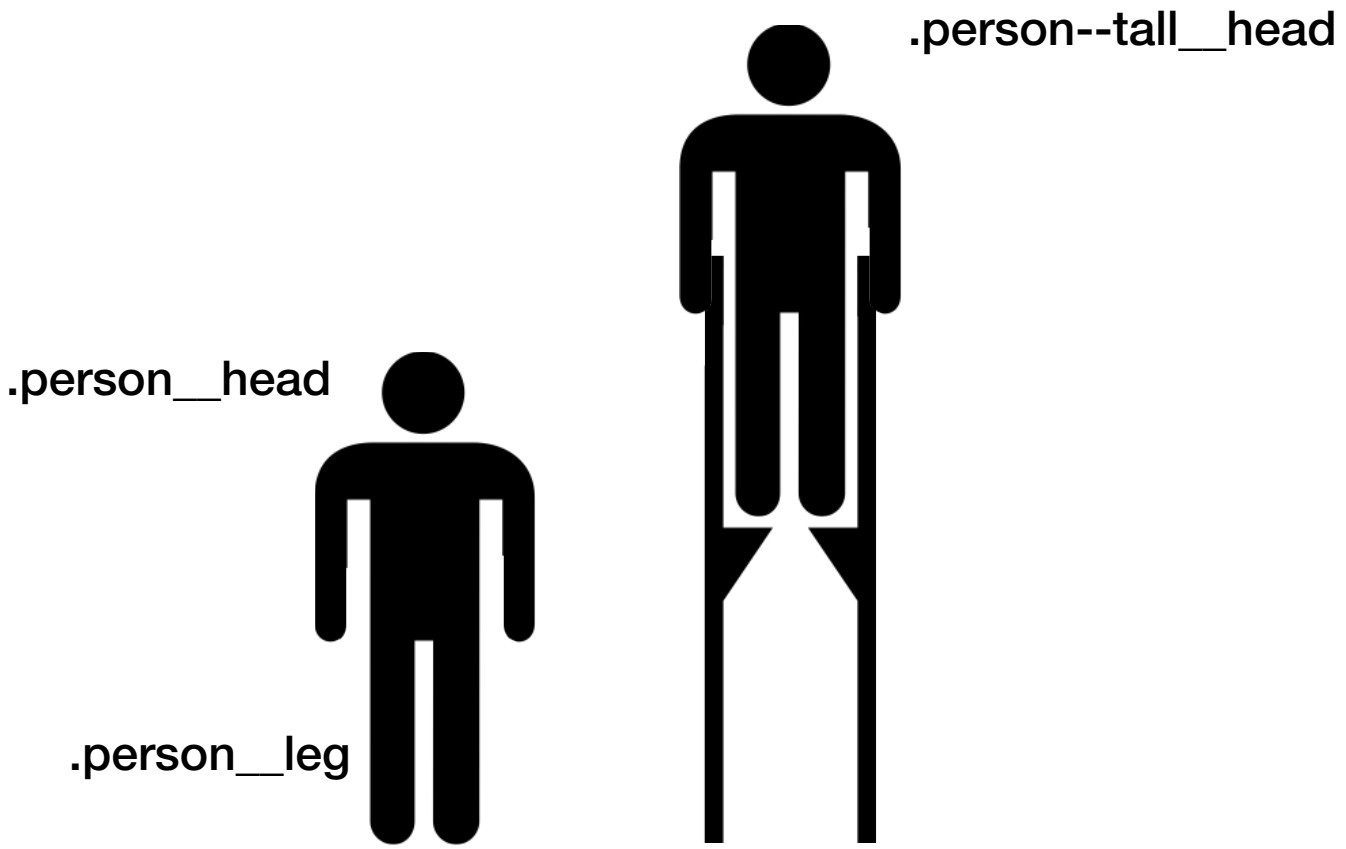
자손 선택자 금지?

.person .person--tall



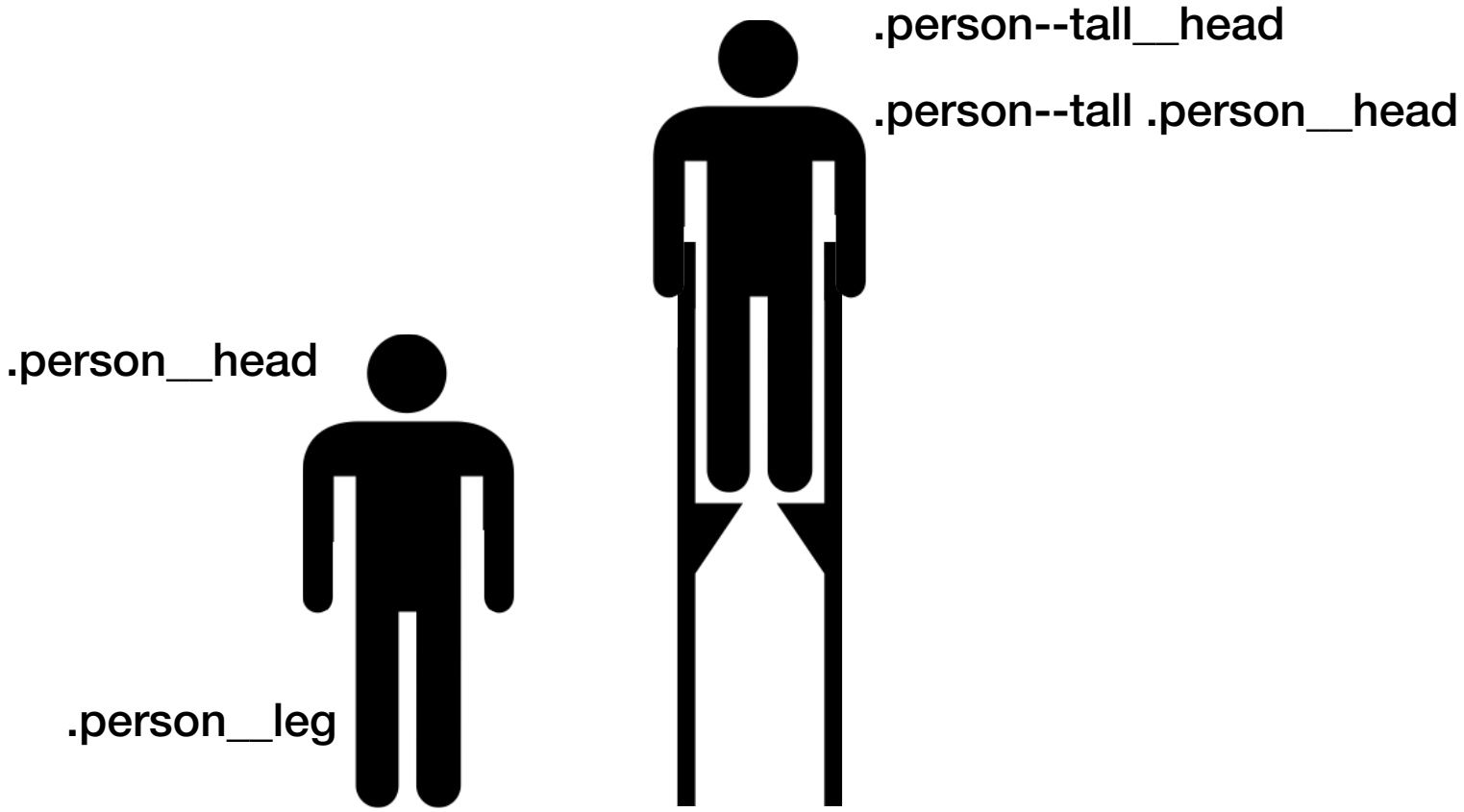
자손 선택자 금지?

.person **.person--tall**



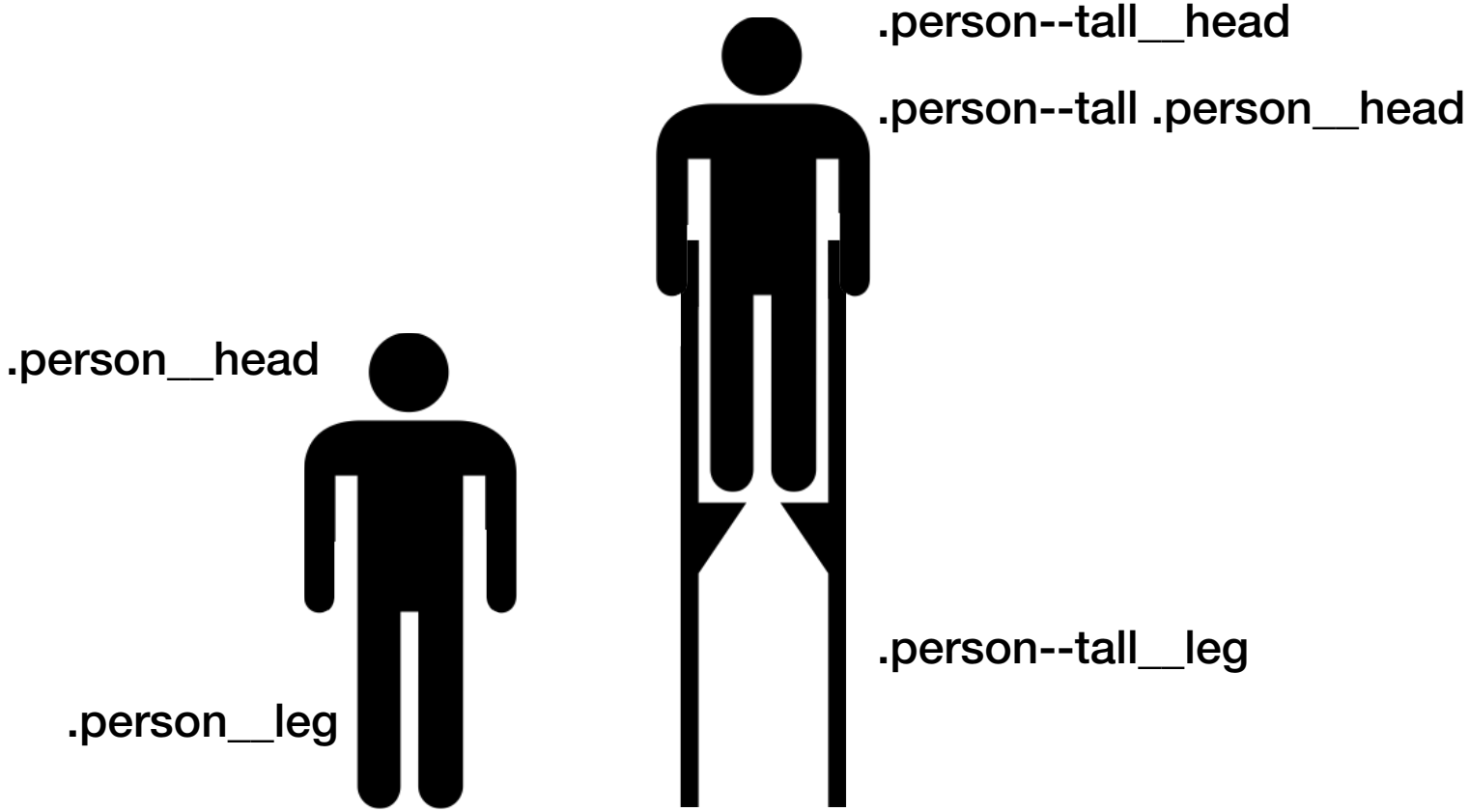
자손 선택자 금지?

.person **.person--tall**



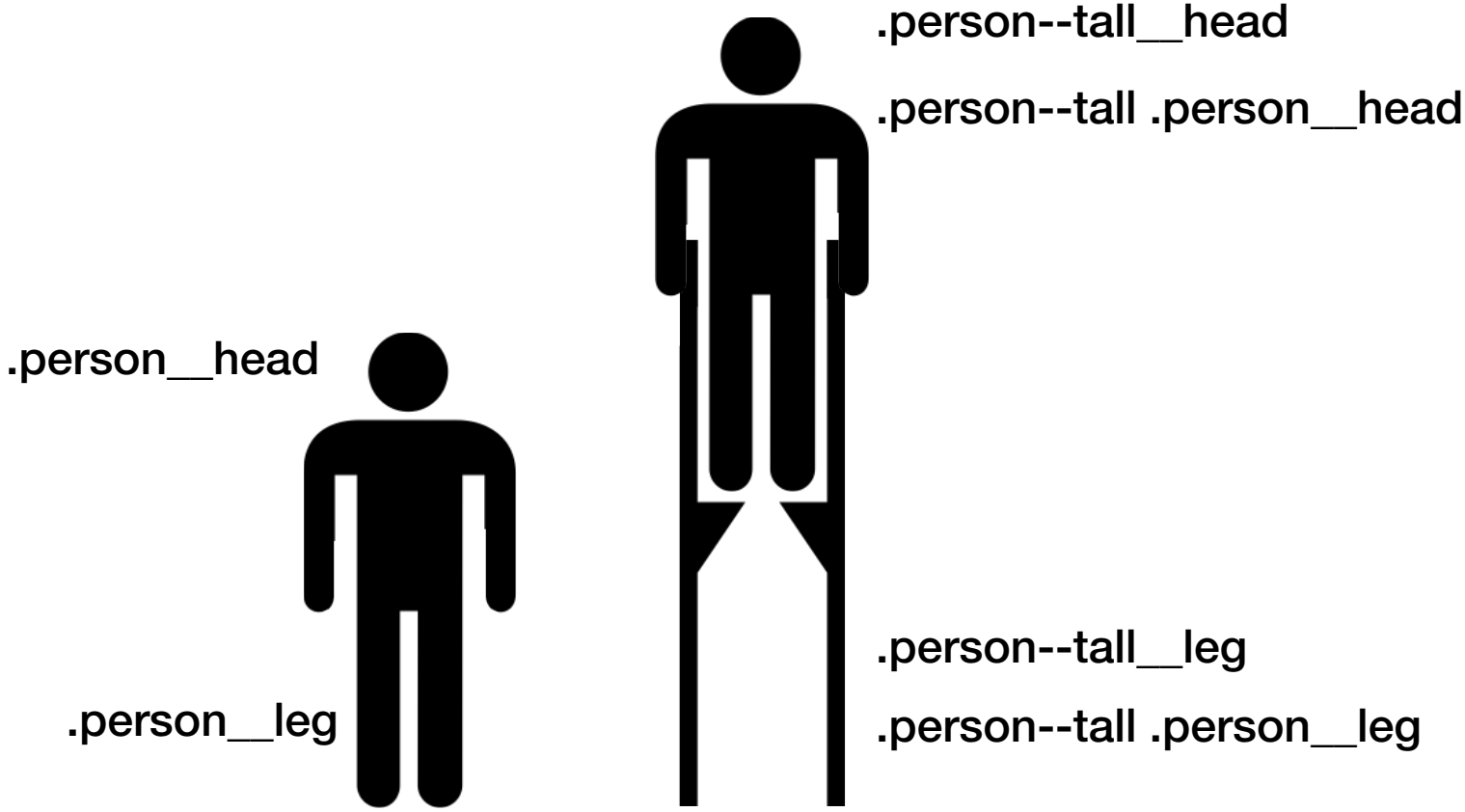
자손 선택자 금지?

.person **.person--tall**



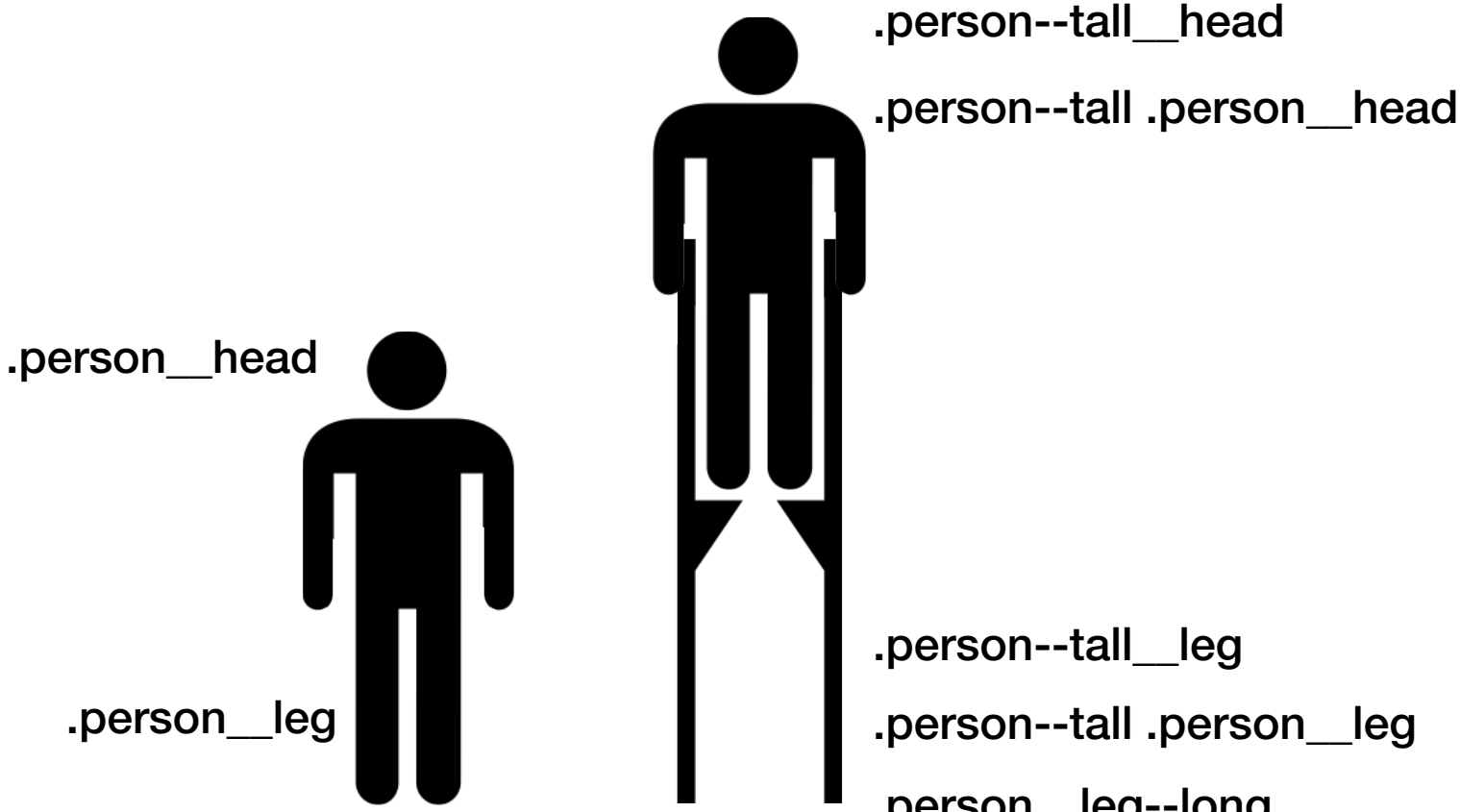
자손 선택자 금지?

.person **.person--tall**



자손 선택자 금지?

.person **.person--tall**



자손 선택자 금지?

- ✔ .person에 .long leg가 있다.

- ✔ .tall person에 평범한 .leg가 있다.

자손 선택자 금지?

- ✔ **.person에 .long leg가 있다.**

  .person__leg--long

- ✔ **.tall person에 평범한 .leg가 있다.**

자손 선택자 금지?

- ✔ **.person에 .long leg가 있다.**

  .person__leg--long

- ✔ **.tall person에 평범한 .leg가 있다.**

  .person--tall .person__leg

자손 선택자 금지?

- ✔ **.person에 .long leg가 있다.**

  .person__leg--long

- ✔ **.tall person에 평범한 .leg가 있다.**

  .person--tall .person__leg

  .person--tall__leg

자손 선택자 금지?

- ✔ **.person에 .long leg가 있다.**

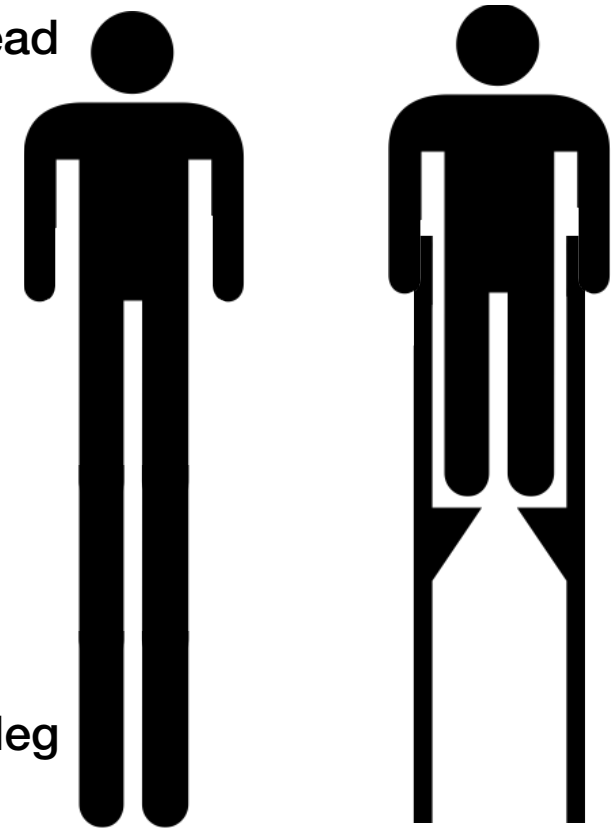
  .person__leg--long

- ✔ **.tall person에 평범한 .leg가 있다.**

  .person--tall .person__leg

자손 선택자 금지?

.person--tall



.person__head

.person--tall__head

.person--tall .person__head

.person__leg

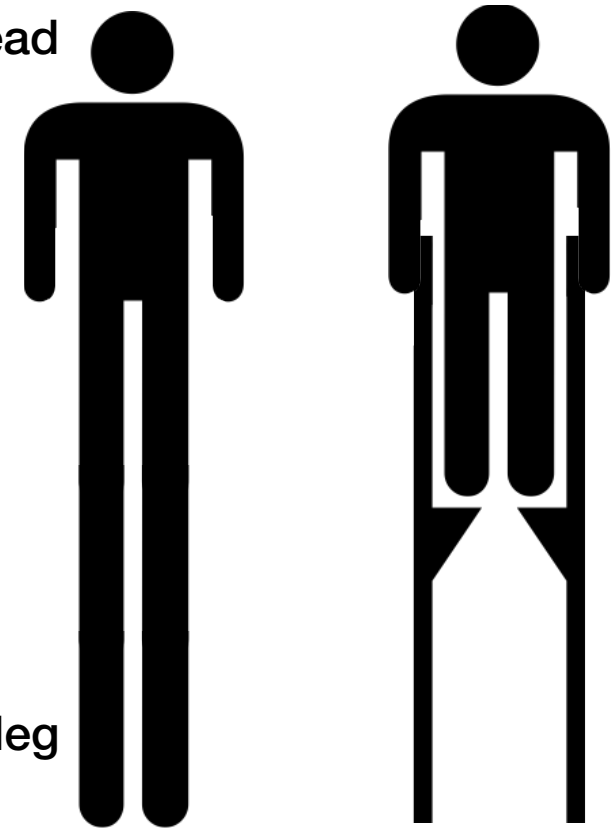
.person--tall__leg

.person--tall .person__leg

.person__leg--long

자손 선택자 금지?

.person--tall



.person__head

.person--tall__head

.person--tall .person__head

.person--tall .person__leg

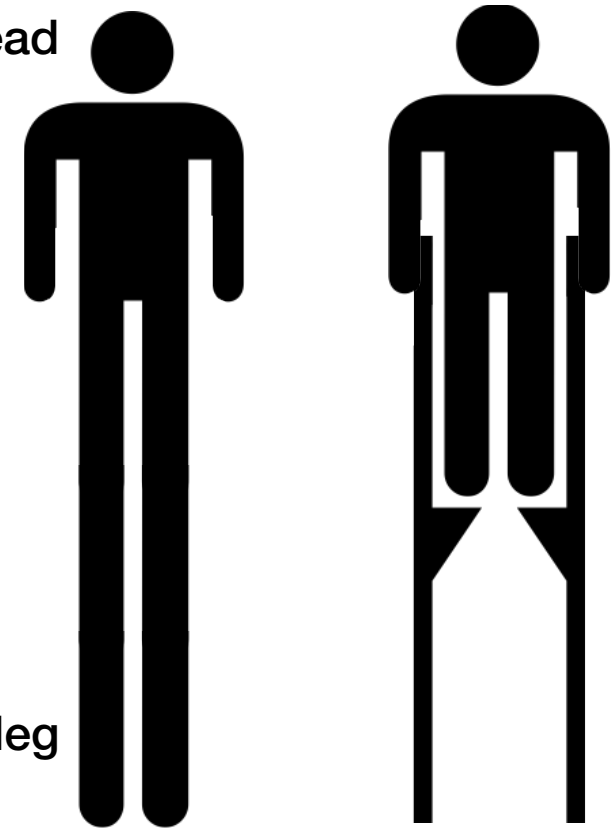
.person--tall__leg

.person--tall .person__leg

.person__leg--long

자손 선택자 금지?

.person--tall



.person__head

.person--tall__head

.person--tall .person__head

.person--tall .person__leg

.person--tall__leg

.person__leg--long

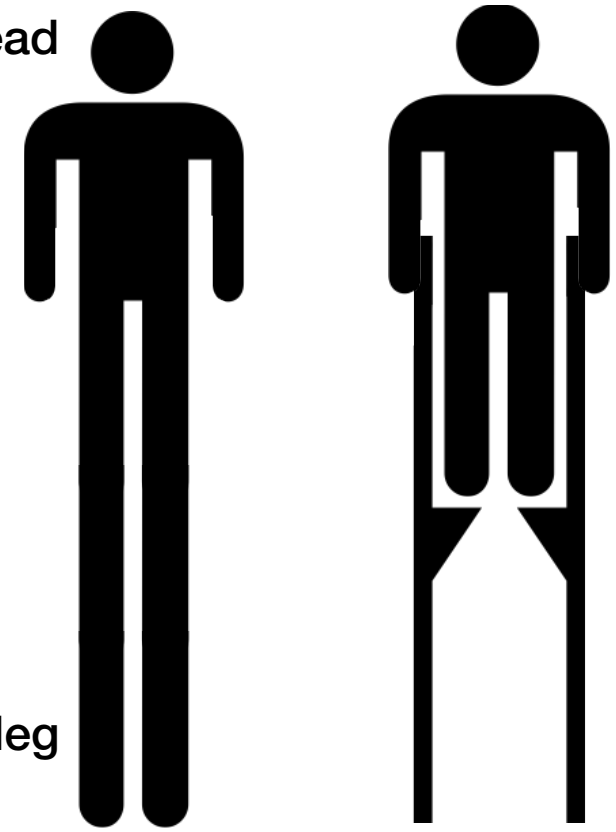
자손 선택자 금지?

.person--tall

.person__head

.person--tall__head

.person--tall .person__head



.person--tall .person__leg

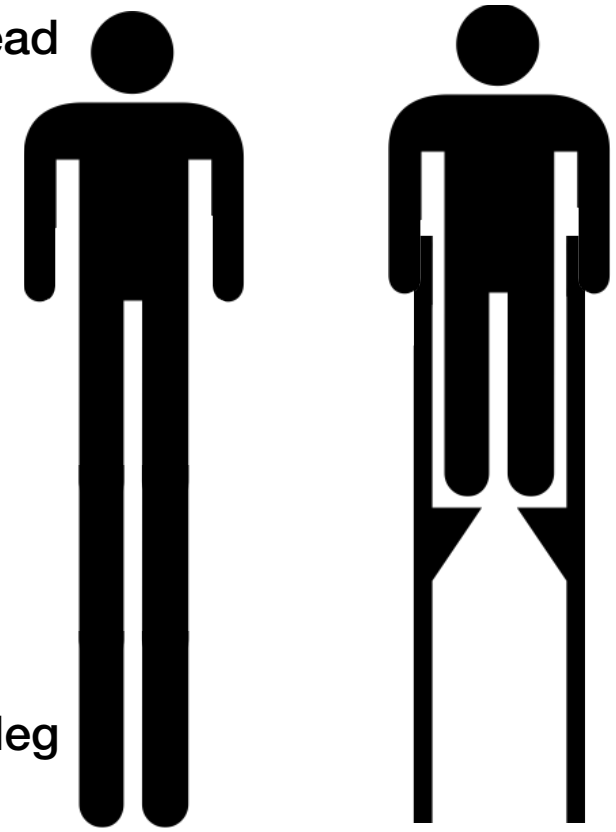
.person__leg--long

자손 선택자 금지?

.person--tall

.person__head

.person--tall .person__head



.person--tall .person__leg

.person__leg--long

자손 선택자 금지?

속성이 적용된 블록을 기준으로 요소를 수정해야 하는 경우

“사용한다.”

여전히 애매하지만

장단점을 찾아봤습니다.

가독성이 좋다.

가독성이 좋다.

선택자 우선 순위를 신경 쓰지 않아도 된다.

가독성이 좋다.

선택자 우선 순위를 신경 쓰지 않아도 된다.

재활용하기 쉽다.

길어진 클래스에 적응하기 힘들다.

길어진 클래스에 적응하기 힘들다.

BEM의 늪에 빠질지도 모른다.

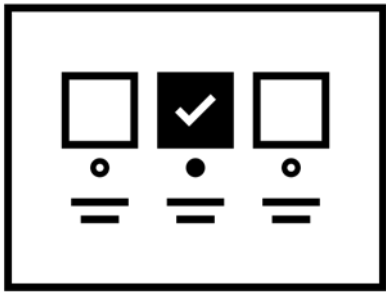
길어진 클래스에 적응하기 힘들다.

BEM의 늪에 빠질지도 모른다.

디자인 수정 시, 클래스를 바꿔야 한다.

공통 UI만 좀 일찍 정리해주면

참 좋을 텐데요.



마무리.

BEM 방식을 사용하면

클래스가 **무슨 일**을 하는지,

어디에 사용되는지,

다른 클래스들과는 **어떤 연관**이 있는지,

한 눈에 알아
볼 수 있다고
하던데...예…

들어주셔서
감사합니다.