

Case1 : Input

케이스 주제

Form 요소 중 가장 기초가 되는 input 스타일링 문제입니다.

Figma에서 제공되는 수치를 확인해서 디자인과 같은 스타일을 작성합니다.

아래의 요구사항을 참고해서 모든 브라우저에서 같은 디자인이 보이도록 해야합니다.

기능 요구사항

1. 기본 input 스타일을 작성해주세요.

label과 input을 연결해주세요.

placeholder 색과 value 텍스트 색상을 디자인처럼 각각 다른 색으로 적용해주세요.

2. hover했을 경우 테두리색이 변경됩니다.

disabled 일 때는 hover가 되더라도 스타일이 변경되지 않도록 해주세요.

3. focus되었을 경우 테두리색이 변경됩니다.

readonly, disabled 일 때는 focus가 되더라도 스타일이 변경되지 않도록 해주세요.

4. readonly의 경우 배경색이 바뀝니다.

5. disabled의 경우 배경색과 글자색이 바뀝니다.

6. error 상태의 경우 input에 error 클래스명을 추가하면 테두리색, 글자색이 바뀌고 에러메시지가 보이도록 합니다.

문제

01-form-input

Default

이름

이름을 적어주세요.

이름

이름을 적어주세요.

이름을 적어주세요.

Readonly

홍길동

Disabled

이름을 적어주세요.

Error

123

잘못된 입력값입니다.

주요 학습 키워드

크로스브라우징을 고려한 input 스타일 적용
input과 label로 연결시켜 웹접근성 고려하기
input의 기본적인 속성을 사용해보기

작성해주셔야 하는 question 파일경로

./question/question.html
./question/question.scss

실행 방법

경로 ./question/question.html
question.html 열기

Case1 : Form input - 출제자 해설

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>01-form-input (solution)</title>
    <link rel="stylesheet" href="../demo.css">
    <link rel="stylesheet" href="solution.css">
</head>
<body>
    <!--
        <label> <input type="text">를 사용해서 주어진 요구사항을 완성하세요.
        디자인 시안에서는 Label, placeholder로 문구가 적혀있지만 디자인 예시이기 때문에,
        영상 강의에서는 실무라고 생각하고 '이름'을 입력받는 상황으로 진행하였습니다.
        readonly, disabled, error 케이스에서는 별도로 레이블을 연결하지 않았습니다.
    -->
    <div class="demo">
        <h1>01-form-input</h1>

        <h2>Default</h2>
        <div class="mgt32">
            <!-- label 활용법1 : 명시적 레이블 / input id 값을 label for로 연결시킨다 -->
            <!-- type here -->
            <label for="name" class="form-label">이름</label>
            <input id="name" type="text" placeholder="이름을 적어주세요." class="form-input">
        </div>

        <div class="mgt32">
            <!-- label 활용법2 : 임시적 레이블 / label으로 input을 감싼다. -->
            <!-- type here -->
            <label>
                <span class="form-label">이름</span>
                <input type="text" placeholder="이름을 적어주세요." class="form-input">
            </label>
        </div>

        <div class="mgt32">
            <!-- label 활용법3 : label을 적지 않는 경우 input에 aria-label=""을 적는다. -->
            <!-- https://developer.mozilla.org/ko/docs/Web/Accessibility/ARIA/ARIA_Techniques/Using_the_aria-label_attribute -->
            <!-- type here -->
            <input type="text" placeholder="이름을 적어주세요." aria-label="이름" class="form-input">
        </div>

        <h2> Readonly</h2>
        <div class="mgt32">
            <!-- type here -->
            <input type="text" placeholder="이름을 적어주세요." aria-label="이름" value="홍길동" class="form-input" readonly>
        </div>

        <h2> Disabled</h2>
        <div class="mgt32">
            <!-- type here -->
            <input type="text" placeholder="이름을 적어주세요." aria-label="이름" class="form-input" disabled>
        </div>

        <h2> Error</h2>
        <div class="mgt32">
            <!-- type here -->
            <input type="text" placeholder="이름을 적어주세요." aria-label="이름" value="123" class="form-input error">
            <p class="txt-error">잘못된 입력값입니다.</p>
        </div>
    </div>
</body>
</html>
```

```
// figma : https://www.figma.com/file/9FXkniEMPgZKtJY4GwP60z/Input?node-id=0%3A3

// color palette
$gray1: #333333;
$gray2: #4f4f4f;
$gray3: #828282;
$gray4: #bdbdbd;
$gray5: #e0e0e0;
$gray6: #f2f2f2;
$gray7: #f9f9f9;
$red: #eb5757;

// color variable rename
$f-bd-color: $gray5;
$f-bd-hover-color: $gray3;
$f-bd-focus-color: $gray1;
$f-bg-color: #fff;
$f-bg-readonly-color: $gray7;
$f-bg-disabled-color: $gray7;
$f-placeholder-color: $gray4;
$f-txt-color: $gray1;
$f-txt-disabled-color: $gray4;
$error-color: $red;

.form-label {
    display: block;
    margin-bottom: 10px;
    font-size: 14px;
    font-weight: 500;
    color: $gray1;
}

input {
    -webkit-appearance: none;
    &:focus {
        outline: none;
    }
}

.form-input {
    display: block;
    padding: 0 16px;
    width: 100%;
    height: 48px;
    border-radius: 4px;
    border: 1px solid $f-bd-color;
    background: $f-bg-color;
    font-size: 14px;
    color: $f-txt-color;
    &::placeholder { /* Chrome, Safari, Firefox */
        color: $f-placeholder-color;
    }
    &::ms-input-placeholder { /* IE, Edge */
        color: $f-placeholder-color;
    }
    &::ms-clear,
    &::ms-reveal { /* IE Input 태그 안 x 버튼 없애기 */
        display: none;
    }
}

&:not([readonly]):not(:disabled:not(.error)) {
    &:hover {
        border-color: $f-bd-hover-color;
    }

    &:focus {
        border-color: $f-bd-focus-color;
    }
}

// (주의) :read-only는 IE에서 적용되지 않습니다.
&[readonly] {
```

```

background-color: $f-bg-readonly-color;
cursor: default;
}

&:disabled {
background-color: $f-bg-disabled-color;
color: $f-txt-disabled-color;
cursor: not-allowed;
}

&.error {
border-color: $error-color;
color: $error-color;
& + .txt-error {
display: block;
}
}

.txt-error {
display: none;
margin-top: 8px;
font-size: 14px;
color: $error-color;
}

```

< Case01 : Form Input > 문제와 같이 보면 좋은 팁

다른강사님께서 해당 출제문제를 본 후, 같이 확인하면 좋겠다고 주신 의견입니다.

HTML

입력서식을 마크업 할 때 사용되는 input 요소의 경우 웹접근성 측면에서 1대1로 대응되는 레이블을 제공하는 것이 왜 필요한지, input 요소와 label 요소를 연결할 경우의 이점에 대한 MDN문서

 <https://developer.mozilla.org/ko/docs/Web/HTML/Element/label>

input 요소에 사용되는 placeholder 속성의 경우 label 요소의 역할과 다름에도 불구하고 간혹 동일하게 취급하거나 사용되는 경우가 있습니다. 하지만 placeholder 속성과 label의 차이점이 있습니다. placeholder은 정보의 종류의 분야 관련하여 사용자에게 간단한 힌트를 제공하는 문자열이고 label은 사용자 인터페이스에서 캡션을 나타냅니다.

 <https://developer.mozilla.org/ko/docs/Web/HTML/Element/Input#htmlattrdefplaceholder>

WAI-ARIA가 무엇인지, aria 관련 속성들에 대한 MDN문서

 <https://developer.mozilla.org/ko/docs/Web/Accessibility/ARIA>

aria-label, aria-labelledby 속성을 비교해보고 어느 상황에서 쓸 수 있는지 아래글을 참고하시면 좋을 것 같습니다.

 <https://developers.google.com/web/fundamentals/accessibility/semantics-aria/aria-labels-and-relationships?hl=ko>

입력 값에 대한 에러 메시지가 어느 입력 서식에 대한 에러인지 이해할 수 있도록 aria-invalid, aria-describedby 속성 등을 적용하여 코드를 개선해보면 좋을 것 같습니다.

 https://developer.mozilla.org/en-US/docs/Web/Accessibility/ARIA/ARIA_Techniques/Using_the_aria-describedby_attribute

 <https://www.w3.org/WAI/WCAG21/Techniques/aria/ARIA21>

에러 메시지는 invalid 상황이 발생했을 때 입력 서식으로 초점이 이동하면 그때 에러 내용을 읽어주게 됩니다. 이 때, 실시간으로 갱신되는 정보의 경우 aria-live 속성을 함께 사용하면 사용자에게 도움이 됩니다. (aria-live 예시 참고 영상)

 <https://www.youtube.com/watch?v=873O8JwI7K4>

CSS

```

input:focus {
outline: none;
}

```

위의 코드를 사용할 경우 웹접근성 관점에서 어떤 문제가 있는지 생각해 보면 좋을 것 같습니다. outline을 0 또는 none값을 주게되면 브라우저의 기본 포커스 스타일이 사라집니다. 만약 어떤 요소가 클릭하는 등 상호작용이 일어나는 요소라면 반드시 시각적으로 포커스 여부를 나타낼 수 있어야합니다. 포커싱이 중요한 이유는 마우스를 사용하지 못하는 키보드 사용자, 저시력 시각장애인 사용자에게 페이지탐색을 할 수 있게 도와주기 때문입니다.

```
.form-input:not(.error):not([readonly]):not(:disabled):focus {  
    border-color: #333333;  
}
```

케이스 예제에서는 input 요소에 :focus 상황이 발생했을 때 기본 outline은 제거했지만 위의 코드처럼 border의 color를 설정하여 키보드 탐색 시에 어떤 서식에 초점이 있는지 인식할 수 있도록 제공되어있습니다.

이와 더불어 웹접근성 관점에서 color의 명도대비를 어떻게 가지고 가는 것이 좋은지 아래 문서를 참고하면 좋을 것 같습니다.

↳ <https://webaim.org/articles/contrast/>

↳ <https://colourcontrast.cc/>

Case2 : Textarea

케이스 주제

Form 요소 중 하나인 textarea 관련 문제입니다.

Figma에서 제공되는 수치를 확인해서 디자인과 같은 스타일을 작성합니다.

아래의 요구사항을 참고해서 모든 브라우저에서 같은 디자인이 보이도록 해야합니다.

기능 요구사항

1. `input` 스타일 코드를 활용하여 기본 `textarea` 스타일을 작성해주세요.

1. `hover`했을 경우 테두리색이 변경됩니다.

`readonly`, `disabled` 일 때는 `hover`가 되더라도 스타일이 변경되지 않도록 해주세요.

2. `focus`되었을 경우 테두리색이 변경됩니다.

`readonly`, `disabled` 일 때는 `focus`가 되더라도 스타일이 변경되지 않도록 해주세요.

3. `readonly`의 경우 배경색이 바뀝니다.

4. `disabled`의 경우 배경색과 글자색이 바뀝니다.

2. 세로 5줄 입력이 가능하도록 높이값을 지정해주세요.

3. `textarea`가 `resize` 되지 않도록 해주세요.

4. `textarea` 내부 스크롤 스타일을 변경

`width: 4px`

border-radius: 2px
background-color: rgba(black, 0.2)

5. (번외) `textarea` 태그를 사용하지 않고도 입력창을 구현할 수 있는 다른 방법을 찾아보세요.

문제

02-form-textarea

1. textarea 스타일 작성
2. 세로 5줄 입력이 가능하도록
3. resize 되지 않도록
4. 내부 스크롤 스타일링 (가로폭 4px, 배경색 rgba(black,0.2))

```
 Lorem ipsum dolor sit amet consectetur adipisicing elit.  
 Suscipit corporis unde, sequi tempore laborum  
 excepturi autem deleniti aliquid eum vitae sunt, beatae  
 omnis sapiente amet, nobis odio et cumque quia. Lorem  
 ipsum dolor sit amet consectetur adipisicing elit.  
 Reiciendis minus ex mollitia! Incidunt quas dolores
```

123

123

(번외) textarea 태그를 외에 입력창을 구현할 수 있는 다른 방법

주요 학습 키워드

textarea 스타일 적용
textarea의 기본적인 속성을 사용해보기

작성해주셔야 하는 question 파일경로

```
./question/question.html
```

```
./question/question.scss
```

실행 방법

경로 ./question/question.html

question.html 열기

Case2 : Form textarea - 출제자 해설코드

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>02-form-textarea (solution)</title>
  <link rel="stylesheet" href="../demo.css">
  <link rel="stylesheet" href="solution.css">
</head>
<body>
  <!-- textarea를 사용해서 주어진 요구사항을 완성하세요. -->
  <div class="demo">
    <h1>02-form-textarea</h1>

    <label class="form-label mgt32" for="textareal">
      1. textarea 스타일 작성<br>
      2. 세로 5줄 입력이 가능하도록<br>
      3. resize 되지 않도록<br>
      4. 내부 스크롤 스타일링 (가로폭 4px, 배경색 rgba(black,0.2))
    </label>
    <div class="mgt16">
      <!-- type here -->
      <textarea rows="5" class="form-textarea" placeholder="내용을 입력하세요">Lorem ipsum dolor sit amet consectetur adipisicing elit. Sunt, nisi!</textarea>
      <textarea rows="5" class="form-textarea mgt16" placeholder="내용을 입력하세요" readonly>123</textarea>
      <textarea rows="5" class="form-textarea mgt16" placeholder="내용을 입력하세요" disabled>123</textarea>
    </div>

    <label class="form-label mgt32" for="textarea2">
      (번외) textarea 태그를 외에 입력창을 구현할 수 있는 다른 방법
    </label>
    <div class="mgt16">
      <div class="form-textarea" contenteditable="true"></div>
    </div>
  </div>
</body>
</html>
```

```
// figma : https://www.figma.com/file/9FXkniEMPgZKtJY4GwP60z/Input?node-id=0%3A3

// color palette
$gray1: #333333;
$gray2: #4f4f4f;
$gray3: #828282;
$gray4: #BDBDBD;
$gray5: #E0E0E0;
$gray6: #F2F2F2;
$gray7: #F9F9F9;
$red: #EB5757;

// color variable rename
$f-bd-color: $gray5;
$f-bd-hover-color: $gray3;
$f-bd-focus-color: $gray1;
$f-bg-color: #fff;
$f-bg-readonly-color: $gray7;
$f-bg-disabled-color: $gray7;
$f-placeholder-color: $gray4;
$f-txt-color: $gray1;
$f-txt-disabled-color: $gray4;
$error-color: $red;

.form-label {
  display: block;
  margin-bottom: 10px;
  font-size: 14px;
  font-weight: 700;
```

```
    color: $f-txt-color;
}

input, textarea {
  -webkit-appearance: none;
  &:focus {
    outline: none;
  }
}

.form-input,
.form-textarea {
  display: block;
  padding: 0 16px;
  width: 100%;
  height: 48px;
  border-radius: 4px;
  border: 1px solid $f-bd-color;
  background: $f-bg-color;
  font-size: 14px;
  color: $f-txt-color;
  // 스크롤 스타일링을 해보세요.
  &::-webkit-scrollbar {
    width: 4px;
  }

  &::-webkit-scrollbar-thumb {
    border-radius: 2px;
    background-color: rgba(black, 0.2);
  }

  &::placeholder { /* Chrome, Safari, Firefox */
    color: $f-placeholder-color;
  }

  &::ms-input-placeholder { /* IE, Edge */
    color: $f-placeholder-color;
  }

  &.error {
    border-color: $error-color;
    color: $error-color;
    & + .txt-error {
      display: block;
    }
  }

  &:not(.error):not([readonly]):not(:disabled) {
    &:hover {
      border-color: $f-bd-hover-color;
    }

    &:focus {
      border-color: $f-bd-focus-color;
    }
  }

  // (주의) :read-only는 IE에서 적용되지 않습니다.
  &[readonly] {
    background-color: $f-bg-readonly-color;
    cursor: default;
  }

  &:disabled {
    background-color: $f-bg-disabled-color;
    color: $f-txt-disabled-color;
    cursor: not-allowed;
  }
}

.form-textarea {
  padding: 16px;
  height: auto;
  resize: none;
}

.txt-error {
  display: none;
}
```

```
    margin-top: 8px;
    font-size: 14px;
    color: $error-color;
}
```

<Case02 : Form Textarea> 문제와 같이 보면 좋은 팁

다른강사님께서 해당 출제문제를 본 후, 같이 확인하면 좋겠다고 주신 의견입니다.

HTML

코드에서 textarea 요소에 적절한 label 요소가 제공되도록 수정해보면 좋을 것 같습니다. 그 이유는 모든 입력 서식은 1대1로 대응되는 label 요소를 가져야하기 때문입니다. 만약 label 요소가 없다면 aria-label이나 aria-labelledby 속성의 활용이 도움이 될 것 입니다.
input 요소에 사용되는 placeholder 속성의 경우 label 요소의 역할과 다름에도 불구하고 간혹 동일하게 취급하거나 사용되는 경우가 있습니다. 하지만 placeholder 속성과 label의 차이점이 있습니다. placeholder은 정보의 종류의 분야 관련하여 사용자에게 간단한 힌트를 제공하는 문자열이고 label은 사용자 인터페이스에서 캡션을 나타냅니다.

☞ <https://developer.mozilla.org/ko/docs/Web/HTML/Element/Input#htmlattrdefplaceholder>

예시)

```
<textarea id="textareal" class="form-textarea" rows="5" aria-label="가입인사" placeholder="내용을 입력하세요."></textarea>
```

웹접근성 관점에서 입력 값에 대한 에러 메시지가 어느 입력 서식에 대한 에러인지 이해할 수 있도록 aria-invalid, aria-describedby 속성을 적용하여 좀 더 양질의 코드를 경험하고 활용할 수 있도록 코드를 수정하는 것을 권장합니다.

☞ https://developer.mozilla.org/en-US/docs/Web/Accessibility/ARIA/ARIA_Techniques/Using_the_aria-describedby_attribute

☞ <https://www.w3.org/WAI/WCAG21/Techniques/aria/ARIA21>

예시)

```
<div class="mgt16">
  <textarea class="form-textarea error" rows="5" placeholder="내용을 입력" aria-describedby="errorMessage" aria-invalid="true"></textarea>
  <p class="txt-error" id="errorMessage">에러 메시지</p>
</div>
```

에러 메시지는 invalid 상황이 발생했을 때 입력 서식으로 초점이 이동하면 그때 에러 내용을 읽어주게 됩니다. 그러므로 실시간으로 갱신되는 정보의 경우 aria-live 속성을 함께 사용하는 것도 도움이 됩니다.

aria-live 예시 참고 영상

☞ <https://www.youtube.com/watch?v=873O8Jwl7K4>

contenteditable 속성

☞ https://developer.mozilla.org/ko/docs/Web/HTML/Global_attributes/contenteditable

CSS

```
input:focus {
  outline: none;
}
```

위의 코드를 사용할 경우 웹접근성 관점에서 어떤 문제가 있는지 생각해 보면 좋을 것 같습니다. outline을 0 또는 none값을 주게되면 브라우저의 기본 포커스 스타일이 사라집니다. 만약 어떤 요소가 클릭하는 등 상호작용이 일어나는 요소라면 반드시 시각적으로 포커스 여부를 나타낼 수 있어야합니다. 포커스링이 중요한 이유는 마우스를 사용하지 못하는 키보드 사용자, 저시력 시각장애인 사용자에게 페이지탐색을 할 수 있게 도와주기 때문입니다.

```
.form-input:not(.error):not([readonly]):not(:disabled):focus {
  border-color: #333333;
}
```

케이스 예제에서는 input 요소에 :focus 상황이 발생했을 때 기본 outline은 제거했지만 위의 코드처럼 border의 color를 설정하여 키보드 탐색 시에 어떤 서식에 초점이 있는지 인식할 수 있도록 제공되어 있습니다.

이와 더불어 웹접근성 관점에서 color의 명도대비를 어떻게 가지고 가는 것이 좋은지 아래 문서를 참고하면 좋을 것 같습니다.

↳ <https://webaim.org/articles/contrast/>

↳ <https://colourcontrast.cc/>

Case3 : checkbox & switch(toggle)

케이스주제

폼태그 중 체크박스 사용하기 & 체크박스를 이용한 스위치(토글) 버튼 문제

기능 요구사항

checkbox

1. 해당 스프라이트를 이용한 단독 체크박스 타입 작성

normal 타입
checked 타입
disabled 타입
checked & disabled 타입

2. 해당 스프라이트를 이용한 체크박스 + 텍스트 타입 작성

normal 타입
checked 타입
disabled 타입
checked & disabled 타입

3. 해당 스프라이트를 이용한 체크박스 + 텍스트 2줄이상 길어질 경우 타입 작성

4. 체크박스 옆의 텍스트를 선택해도 css만으로 checked/unchecked 기능 구현

체크박스 디자인 변화되도록 스타일 작성

switch

1. 단독 스위치 타입 작성

OFF 타입
ON 타입
OFF & disabled 타입
ON & disabled 타입

2. 스위치 + 텍스트 타입 작성

OFF 타입
ON 타입
OFF & disabled 타입
ON & disabled 타입

3. 텍스트 + 스위치로 반대의 경우 타입 작성

문제

체크박스 단독 타입



체크박스 + 텍스트 타입

Normal

Checked

Disabled

Checked Disabled

체크박스 + 텍스트 2줄이상 길어진 타입

Normal
 Normal
 Normal

Checked
 Checked
 Checked

Disabled
 Disabled
 Disabled

Checked
 Checked
 Checked

스위치 단독 타입



스위치 + 텍스트 타입

normal checked normal disabled checked disabled

스위치 + 텍스트 반대 타입

normal checked normal disabled checked disabled

주요 학습 키워드

주어진 체크박스 스프라이트로 체크박스 스타일 적용
체크박스의 기본적인 상태속성들 사용
웹접근성을 위한 input 초점 시각화
웹접근성을 위한 label요소 클릭시, 해당 input 동작
체크박스를 사용한 스위치(토글) 버튼 구현
스위치 on/off 애니메이션 효과
마크업구조가 동일한 상태에서 반대 타입 구현

작성해주셔야 하는 question 파일경로

checkbox

```
./question/case3-1_checkbox/html/checkbox.html
```

```
./question/case3-1_checkbox/scss/checkbox.scss
```

switch

```
./question/case3-2_switch/html/switch.html
```

```
./question/case3-2_switch/scss/switch.scss
```

실행 방법

checkbox

경로 `./question/case3-1_checkbox/html/checkbox.html`

checkbox.html 열기

checkbox 문제는 `./question/case3-1_checkbox/images/ico_comm.png` 스프라이트 이용

switch

경로 `./question/case3-2_switch/html/switch.html`

switch.html 열기

Case3-1 : checkbox - 출제자 해설코드

```
<!DOCTYPE html>
<html lang="ko">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>check box</title>
    <link href="https://fonts.googleapis.com/css2?family=Noto+Sans+KR:wght@400;500&display=swap" rel="stylesheet">
    <link rel="stylesheet" href="../css/checkbox.css">
</head>
<body>
    <div class="container-doc">

        <!-- icon 클래스 -->
        <span class="ico_comm ico_choice"></span>

        <div class="box_choice">
            <strong class="tit_choice">체크박스 단독 타입</strong>

            <!-- normal E/입 -->
            <div class="item_choice type_alone">
                <input type="checkbox" id="agreeChk1_1" class="inp_choice">
                <label for="agreeChk1_1" class="lab_choice">
                    <span class="ico_comm ico_choice"></span>
                </label>
            </div>

            <!-- checked E/입 -->
            <div class="item_choice type_alone">
                <input type="checkbox" id="agreeChk1_2" class="inp_choice" checked="checked">
                <label for="agreeChk1_2" class="lab_choice">
                    <span class="ico_comm ico_choice"></span>
                </label>
            </div>

            <!-- disabled E/입 -->
            <div class="item_choice type_alone">
                <input type="checkbox" id="agreeChk1_3" class="inp_choice" disabled="disabled">
                <label for="agreeChk1_3" class="lab_choice">
                    <span class="ico_comm ico_choice"></span>
                </label>
            </div>

            <!-- checked, disabled 타입 -->
            <div class="item_choice type_alone">
                <input type="checkbox" id="agreeChk1_4" class="inp_choice" checked="checked" disabled="disabled">
                <label for="agreeChk1_4" class="lab_choice">
                    <span class="ico_comm ico_choice"></span>
                </label>
            </div>
        </div>

        <div class="box_choice">
            <strong class="tit_choice">체크박스 + 텍스트 타입</strong>

            <!-- normal E/입 -->
            <div class="item_choice">
                <input type="checkbox" id="agreeChk2_1" class="inp_choice">
                <label for="agreeChk2_1" class="lab_choice">
                    <span class="ico_comm ico_choice"></span>
                    normal
                </label>
            </div>

            <!-- checked E/입 -->
            <div class="item_choice">
                <input type="checkbox" id="agreeChk2_2" class="inp_choice" checked="checked">
                <label for="agreeChk2_2" class="lab_choice">
                    <span class="ico_comm ico_choice"></span>
                </label>
            </div>
        </div>
    </div>

```

```

        checked
    </label>
</div>

<!-- disabled E|입 -->
<div class="item_choice">
    <input type="checkbox" id="agreeChk2_3" class="inp_choice" disabled="disabled">
    <label for="agreeChk2_3" class="lab_choice">
        <span class="ico_comm ico_choice"></span>
        normal disabled
    </label>
</div>

<!-- checked, disabled E|입 -->
<div class="item_choice">
    <input type="checkbox" id="agreeChk2_4" class="inp_choice" checked="checked" disabled="disabled">
    <label for="agreeChk2_4" class="lab_choice">
        <span class="ico_comm ico_choice"></span>
        checked disabled
    </label>
</div>
</div>

<div class="box_choice">
    <strong class="tit_choice">체크박스 + 텍스트 2줄이상 길어진 타입</strong>

    <!-- normal E|입 -->
    <div class="item_choice">
        <input type="checkbox" id="agreeChk3_1" class="inp_choice">
        <label for="agreeChk3_1" class="lab_choice">
            <span class="ico_comm ico_choice"></span>
            normal normal normal
        </label>
    </div>

    <!-- checked E|입 -->
    <div class="item_choice">
        <input type="checkbox" id="agreeChk3_2" class="inp_choice" checked="checked">
        <label for="agreeChk3_2" class="lab_choice">
            <span class="ico_comm ico_choice"></span>
            checked checked checked
        </label>
    </div>

    <!-- disabled E|입 -->
    <div class="item_choice">
        <input type="checkbox" id="agreeChk3_3" class="inp_choice" disabled="disabled">
        <label for="agreeChk3_3" class="lab_choice">
            <span class="ico_comm ico_choice"></span>
            normal disabled normal disabled normal disabled
        </label>
    </div>

    <!-- checked, disabled E|입 -->
    <div class="item_choice">
        <input type="checkbox" id="agreeChk3_4" class="inp_choice" checked="checked" disabled="disabled">
        <label for="agreeChk3_4" class="lab_choice">
            <span class="ico_comm ico_choice"></span>
            checked disabled checked disabled checked disabled
        </label>
    </div>
</div>
</body>
</html>

```

```

@charset "UTF-8";

/* reset */
*{font-weight:500;font-family:"Noto Sans KR";color:#625F67}
input[type="checkbox"]{margin:0}

/* sprite */
.ico_comm{display:inline-block;overflow:hidden;font-size:0;line-height:0;background:url(..../images/ico_comm.png) no-repeat 0 0;text-indent:-9999px}
.ico_choice{width:18px;height:18px;background-position:0 0}

```

```

/* content */
.box_choice{margin:20px 0}
.tit_choice{display:block;margin:10px 0}

/* checkbox */
/* checkbox 스타일 작성 영역 */
.item_choice{
    display:inline-block;position:relative;max-width:150px;vertical-align:top;
    .inp_choice{
        position:absolute;top:0;left:0;width:18px;height:18px;
        appearance:none;-webkit-appearance:none;-moz-appearance:none;cursor:pointer;
        &:checked + .lab_choice .ico_choice{background-position:-20px 0;pointer-events:none}
        &:disabled,
        &:checked:disabled{cursor:not-allowed}
        &:disabled + .lab_choice,
        &:checked:disabled + .lab_choice{cursor:not-allowed}
        &:disabled + .lab_choice .ico_choice{background-position:-40px 0}
        &:checked:disabled + .lab_choice .ico_choice{background-position:-60px 0}
    }
    .lab_choice{
        display:block;position:relative;padding-left:26px;font-size:14px;line-height:18px;cursor:pointer;
        .ico_choice{position:absolute;top:0;left:0;background-color:#ffff}
    }
    &.type_alone .lab_choice{min-width:18px;min-height:18px;padding-left:0}
}

```

< Case03 : Form checkbox > 문제와 같이 보면 좋은 팁

다른강사님께서 해당 출제문제를 본 후, 같이 확인하면 좋겠다고 주신 의견입니다.

HTML

checkbox 단독 타입의 경우 label 요소에 label 텍스트가 존재 하지 않는 구조이므로
 aria-label 속성을 활용하여 checkbox가 어떤 용도인지를 이해할 수 있도록 수정해보면 좋을 것 같습니다. 그 이유는 모든 입력 서식은 1대1로 대응되는 레이블이 필요하기 때문입니다.

 <https://developer.mozilla.org/ko/docs/Web/HTML/Element/label>

CSS

CSS Sprite와 IR 기법도 함께 연결해서 생각해보면 좋을 것 같습니다. 그리고 현재 solution에서 사용하고 있는 text-indent를 활용한 IR기법의 경우 성능 저하를 불러올 수 있기에 다른 방법을 사용하여 수정하면 좋을 것 같습니다.

 <https://nuli.navercorp.com/community/article/1132804?email=true>

appearance 속성을 사용하는 vendor prefix의 경우 CSS 작업 시 일일이 부여하는 방식보다 Sass의 Mixin을 활용하거나 추후 Webpack 환경에서 PostCSS Plugin인 autoprefixer로 처리할 수 있습니다.

pointer-events의 설명 및 참고예시 입니다.

 <https://developer.mozilla.org/ko/docs/Web/CSS/pointer-events>

 <https://css-tricks.com/almanac/properties/p/pointer-events/>

Case3-2 : Switch - 출제자 해설코드

```
<!DOCTYPE html>
<html lang="ko">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>switch(toggle)</title>
    <link href="https://fonts.googleapis.com/css2?family=Noto+Sans+KR:wght@400;500&display=swap" rel="stylesheet">
    <link rel="stylesheet" href="../css/switch.css">
</head>
<body>
    <div class="container-doc">

        <div class="box_choice">
            <strong class="tit_choice">스위치 단독 타입</strong>

            <!-- normal 터치 -->
            <div class="item_choice type_alone">
                <input type="checkbox" id="switch1_1" class="inp_switch">
                <label for="switch1_1" class="lab_choice"></label>
            </div>

            <!-- checked 터치 -->
            <div class="item_choice type_alone">
                <input type="checkbox" id="switch1_2" class="inp_switch" checked="checked">
                <label for="switch1_2" class="lab_choice"></label>
            </div>

            <!-- disabled 터치 -->
            <div class="item_choice type_alone">
                <input type="checkbox" id="switch1_3" class="inp_switch" disabled="disabled">
                <label for="switch1_3" class="lab_choice"></label>
            </div>

            <!-- checked, disabled 터치 -->
            <div class="item_choice type_alone">
                <input type="checkbox" id="switch1_4" class="inp_switch" checked="checked" disabled="disabled">
                <label for="switch1_4" class="lab_choice"></label>
            </div>
        </div>

        <div class="box_choice">
            <strong class="tit_choice">스위치 + 텍스트 타입</strong>

            <!-- normal 터치 -->
            <div class="item_choice">
                <input type="checkbox" id="switch2_1" class="inp_switch">
                <label for="switch2_1" class="lab_choice">
                    <span class="txt_switch">normal</span>
                </label>
            </div>

            <!-- checked 터치 -->
            <div class="item_choice">
                <input type="checkbox" id="switch2_2" class="inp_switch" checked="checked">
                <label for="switch2_2" class="lab_choice">
                    <span class="txt_switch">checked</span>
                </label>
            </div>

            <!-- disabled 터치 -->
            <div class="item_choice">
                <input type="checkbox" id="switch2_3" class="inp_switch" disabled="disabled">
                <label for="switch2_3" class="lab_choice">
                    <span class="txt_switch">normal disabled</span>
                </label>
            </div>

            <!-- checked, disabled 터치 -->
        </div>
    </div>

```

```

<div class="item_choice">
    <input type="checkbox" id="switch2_4" class="inp_switch" checked="checked" disabled="disabled">
    <label for="switch2_4" class="lab_choice">
        <span class="txt_switch">checked disabled</span>
    </label>
</div>
</div>

<div class="box_choice">
    <strong class="tit_choice">스위치 + 텍스트 반대 타입</strong>

    <!-- normal 티입 -->
    <div class="item_choice type_reverse">
        <input type="checkbox" id="agreeChk3_1" class="inp_switch">
        <label for="agreeChk3_1" class="lab_choice">
            <span class="txt_switch">normal</span>
        </label>
    </div>

    <!-- checked 티입 -->
    <div class="item_choice type_reverse">
        <input type="checkbox" id="agreeChk3_2" class="inp_switch" checked="checked">
        <label for="agreeChk3_2" class="lab_choice">
            <span class="txt_switch">checked</span>
        </label>
    </div>

    <!-- disabled 티입 -->
    <div class="item_choice type_reverse">
        <input type="checkbox" id="agreeChk3_3" class="inp_switch" disabled="disabled">
        <label for="agreeChk3_3" class="lab_choice">
            <span class="txt_switch">normal disabled</span>
        </label>
    </div>

    <!-- checked, disabled 티입 -->
    <div class="item_choice type_reverse">
        <input type="checkbox" id="agreeChk3_4" class="inp_switch" checked="checked" disabled="disabled">
        <label for="agreeChk3_4" class="lab_choice">
            <span class="txt_switch">checked disabled</span>
        </label>
    </div>
</div>
</body>
</html>

```

```

@charset "UTF-8";

/* reset */
*{font-weight:500;font-family:"Noto Sans KR";font-size:14px;line-height:24px;color:#625F67}
input[type="checkbox"]{margin:0}

/* content */
.box_choice{margin:20px 0}
.tit_choice{display:block;margin:10px 0}

/* color */
$color1 : #4056c7;
$color2 : #97b3cd;
$color3 : #fff;

/* checkbox */
/* switch 스타일 작성 영역 */
.item_choice{
    display:inline-block;position:relative;vertical-align:top;
    .inp_switch{
        position:absolute;top:0;left:0;width:100%;height:100%;
        appearance:none;-webkit-appearance:none;-moz-appearance:none;cursor:pointer;
        &:checked + .lab_choice::before{background-color: $color1;}
        &:checked + .lab_choice::after{left:20px}
        &:disabled,
        &:disabled + .lab_choice,
        &:checked:disabled,
        &:checked:disabled + .lab_choice{cursor:not-allowed}
    }
}

```

```
&:disabled + .lab_choice::before,  
&:checked:disabled + .lab_choice::before{opacity: 0.4;}  
}  
.lab_choice{  
    display:block;position:relative;padding-left:50px;cursor: pointer;  
    .txt_switch{display:block;}  
    &:before{  
        position:absolute;top:0;left:0;width:40px;height:100%;border-radius:12px;background-color:$color2;content:"";  
        -webkit-transition:background-color .3s ease;-moz-transition:background-color .3s ease;-ms-transition:background-color .3s ease;-o-transition:background-color .3s ease;  
    }  
    &:after{  
        position:absolute;top:4px;left:4px;width:16px;height:16px;border-radius: 50%;background-color:$color3;content:"";  
        -webkit-transition:all .3s ease;-moz-transition:all .3s ease;-ms-transition:all .3s ease;;transition:all .3s ease;  
    }  
}  
&.type_alone .lab_choice{min-width:40px;min-height:24px;padding-left:0}  
&.type_reverse{  
    .inp_switch:checked + .lab_choice:after{left:auto;right:4px}  
    .lab_choice{  
        padding-left:0;padding-right:50px;  
        &:before{left:auto;right:0}  
        &:after{left:auto;right:20px}  
    }  
}  
}
```

Case8 : Footer

케이스 주제

Figma에서 제공되는 수치를 확인해서 디자인과 같은 반응형 UI를 마크업합니다.
아래의 요구사항을 참고해서 모든 브라우저에서 같은 디자인이 보이도록 해야합니다.

기능 요구사항

1. 디자인 가이드를 참고하여 브라우저 사이즈에 따라 적절한 순서로 위치할 수 있는 레이아웃 스타일을 작성해주세요.

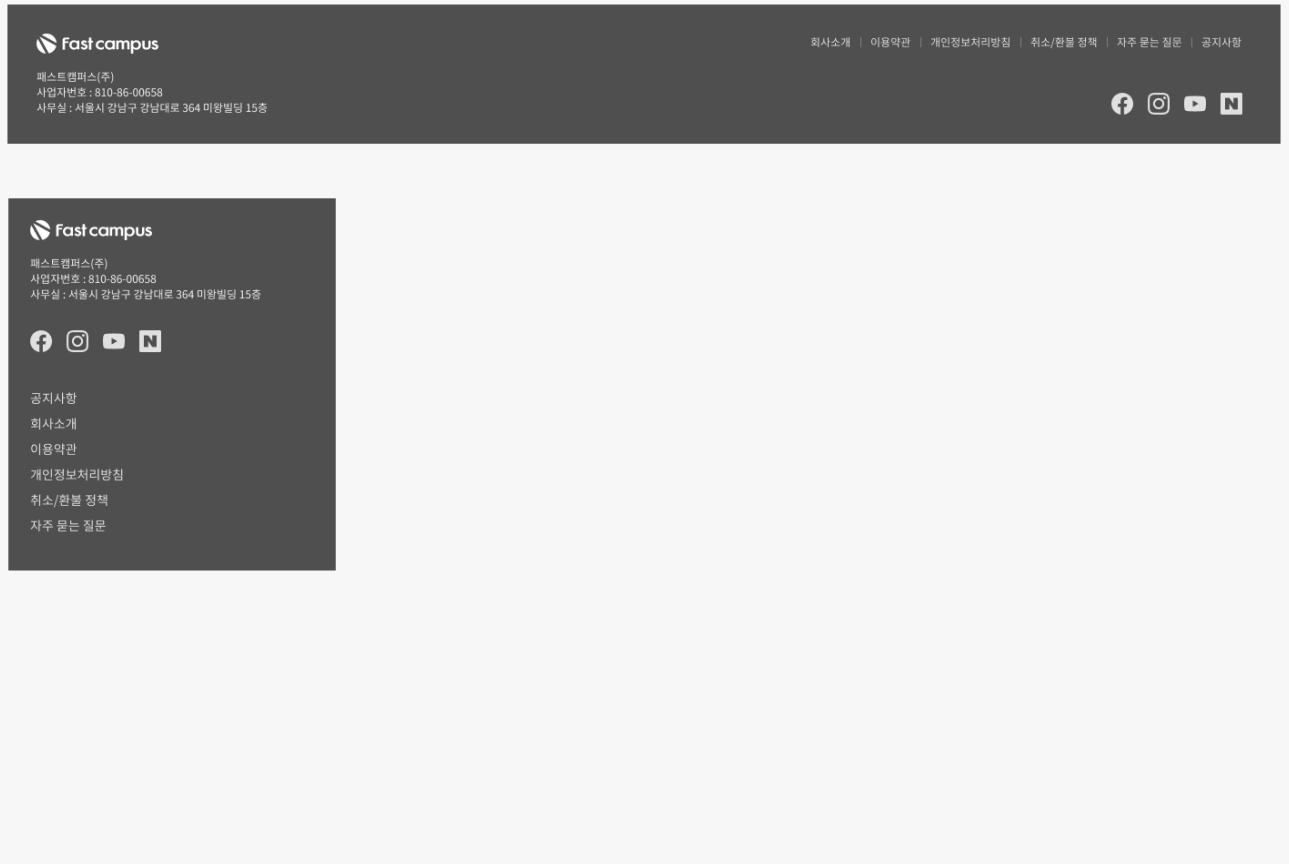
브라우저 사이즈에 따라 레이아웃 위치가 변경되는 것을 유의해주세요.
media query를 사용하여 반응형 디자인 스타일을 작성해주세요.

2. 디자인 가이드에 따라 컨텐츠를 채워주세요.
3. 디자인 가이드의 수치에 따라 스타일을 작성해주세요.

브라우저 사이즈에 따라 메뉴의 divider가 변경되는 것을 유의해주세요.

문제

Figma에서 확인하기



주요 학습 키워드

디자인 가이드를 참고하여 스타일 작성하기
반응형 스타일 작성하기

작성해주셔야 하는 question 파일경로

```
./question/question.html  
./question/question.css  
./question/question.scss
```

실행 방법

경로 ./question/question.html
index.html 열기

Case8 : Footer - 출제자 해설코드

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <title>02-footer (solution)</title>
    <link href="https://fonts.googleapis.com/css2?family=Noto+Sans+KR:wght@400;700&display=swap" rel="stylesheet" />
    <link rel="stylesheet" href="reset.css" />
    <link rel="stylesheet" href="solution.css" />
  </head>
  <body>
    <footer class="footer">
      <div class="footer-left-container">
        <div class="footer-logo">
          
        </div>
        <div class="footer-company-info">
          패스트캠퍼스(주)<br />
          사업자번호 : 810-86-00658<br />
          사무실 : 서울시 강남구 강남대로 364 미왕빌딩 15층
        </div>
      </div>

      <!-- 브라우저 사이즈에 따라 레이아웃 위치를 변경하기 위해 -->
      <div class="footer-right-container">
        <nav class="footer-menu-container">
          <ul class="footer-menu">
            <li class="footer-menu-item">
              <a href="#" class="footer-menu-item-link">회사소개</a>
            </li>
            <li class="footer-menu-item">
              <a href="#" class="footer-menu-item-link">이용약관</a>
            </li>
            <li class="footer-menu-item">
              <a href="#" class="footer-menu-item-link">개인정보처리방침</a>
            </li>
            <li class="footer-menu-item">
              <a href="#" class="footer-menu-item-link">취소/환불 정책</a>
            </li>
            <li class="footer-menu-item">
              <a href="#" class="footer-menu-item-link">자주 묻는 질문</a>
            </li>
          </ul>
        </nav>
      </div>
    </footer>
  </body>

```

```
<li class="footer-menu-item">
    <a href="#" class="footer-menu-item-link">공지사항</a>
</li>
</ul>
</nav>

<nav class="footer-channel-container">
    <ul class="footer-channel">
        <li class="footer-channel-item">
            <a class="footer-channel-item-link" href="#">
                
            </a>
        </li>
        <li class="footer-channel-item">
            <a class="footer-channel-item-link" href="#">
                
            </a>
        </li>
        <li class="footer-channel-item">
            <a class="footer-channel-item-link" href="#">
                
            </a>
        </li>
        <li class="footer-channel-item">
            <a class="footer-channel-item-link" href="#">
                
            </a>
        </li>
    </ul>
</nav>
</div>
</footer>
</body>
</html>
```

```
// figma : https://www.figma.com/file/9FXkniEMPgZKtJY4GwP60z/Input?node-id=0%3A3

// color palette
$gray1: #333333;
$gray2: #4f4f4f;
$gray3: #828282;
$gray4: #bdbdbd;
$gray5: #e0e0e0;
$gray6: #f2f2f2;
$gray7: #f9f9f9;
$primary: #ed234b;

// color variable rename
$background-color: $gray2;
$text-color: $gray6;
$menu-text-color: $gray5;
$divider-color: $gray3;

// default
html,
body {
    font-family: "Noto Sans KR", sans-serif;
    font-size: 14px;
    line-height: 20px;
    color: $text-color;
}

.footer {
    display: flex;
    justify-content: space-between;
    background: $background-color;
    padding: 32px;

    .footer-left-container {
        .footer-logo {
            margin-bottom: 16px;
        }
    }
}

.footer-right-container {
    display: flex;
    flex-direction: column;
    justify-content: space-between;

    .footer-menu-container {
        .footer-menu {
            display: flex;

            .footer-menu-item {
                display: flex;
                align-items: center;
            }
        }
    }
}

&:not(:last-child)::after {
```



```

        .footer-menu-item-link {
            font-size: 14px;
        }
    }

    .footer-channel-container {
        margin: 32px 0 40px;

        .footer-channel {
            justify-content: flex-start;
        }
    }

}

}

```

<Case8 : Footer> 문제와 같이 보면 좋은 팁

다른강사님께서 해당 출제문제를 본 후, 같이 확인하면 좋겠다고 주신 의견입니다.

HTML

푸터는 섹션 콘텐츠나 섹션 루트 예를 들어 `<body>`, `<section>`, `<article>` 등의 요소에서 사용할 수 있으며 작성자 정보나 저작권, 연락처 등의 정보를 담을 수 있습니다.

이때 저작권 정보는 `<small>` 요소, 연락처 정보는 `<address>` 요소의 사용이 적절합니다.

HTML 제작 시 콘텐츠의 의미를 명확하게 구분할 수 있는 요소를 고민하는 것이 좀 더 양질의 마크업 코드를 구현할 수 있는 지름길이 될 수 있습니다.

그리고 마크업 시에는 웹접근성 관점에서의 고민도 함께 필요합니다. 스크린리더 등에서 푸터 콘텐츠가 랜드마크 역할로 인식되도록 하려면 WAI-ARIA의 `role="contentinfo"`를 추가하시면 됩니다.

푸터에 관한 내용은 공식 레퍼런스나 MDN을 참고하시기 바랍니다.

 <https://html.spec.whatwg.org/multipage/sections.html#the-footer-element>

 <https://developer.mozilla.org/ko/docs/Web/HTML/Element/footer>

 https://developer.mozilla.org/en-US/docs/Web/Accessibility/ARIA/Roles/Contentinfo_role

CSS

많은 고객들이 CSS 작업 시 푸터 영역을 항상 브라우저 하단에 위치하도록 요구하는 경우가 있습니다. 웹을 표현할 수 있는 디바이스가 많아지면서 화면의 크기가 다양해지기 때문에 발생하는 요구사항인데요. 반응형 디자인 시 CSS의 플렉스 관련 속성을 사용하면 이를 구현할 수 있습니다. 해당 기능 구현은 위한 마크업 및 CSS 코드는 다음과 같습니다.

```

<div class="flex-container">
    <header class="header">헤더</header>
    <main class="main">메인콘텐츠</main>
    <footer class="footer">푸터</footer>
</div>

```

```
.flex-container {  
  display: flex;  
  flex-direction: column;  
}  
  
.footer {  
  margin-top: auto;  
}
```

위 코드의 핵심은 푸터를 화면 하단에 배치하기 위해 auto 마진을 활용했다는 것입니다. 그동안 auto 마진은 가운데 정렬 시 사용하는 트릭 정도로만 생각하고 계셨을텐데요 플렉스 아이템에 적용하여 다양한 형태의 레이아웃 구현이 가능합니다. 반응형 디자인에서 플렉스 관련 속성의 활용이 중요해 지고 있습니다. 플렉스 관련 속성에 대해 보다 심도있는 관심과 학습을 권장합니다. 보다 다양한 플렉스 레이아웃 방법이 궁금하시다면 Naver D2 사이트를 참고하시기 바랍니다.

📎 <https://d2.naver.com/helloworld/8540176#ch4>

Case9 : Login

케이스 주제

Figma에서 제공되는 수치를 확인해서 디자인과 같은 반응형 UI를 마크업합니다.
아래의 요구사항을 참고해서 모든 브라우저에서 같은 디자인이 보이도록 해야합니다.

기능 요구사항

1. 디자인 가이드를 참고하여 브라우저 사이즈에 따라 적절하게 위치할 수 있는 레이아웃 스타일을 작성해주세요.

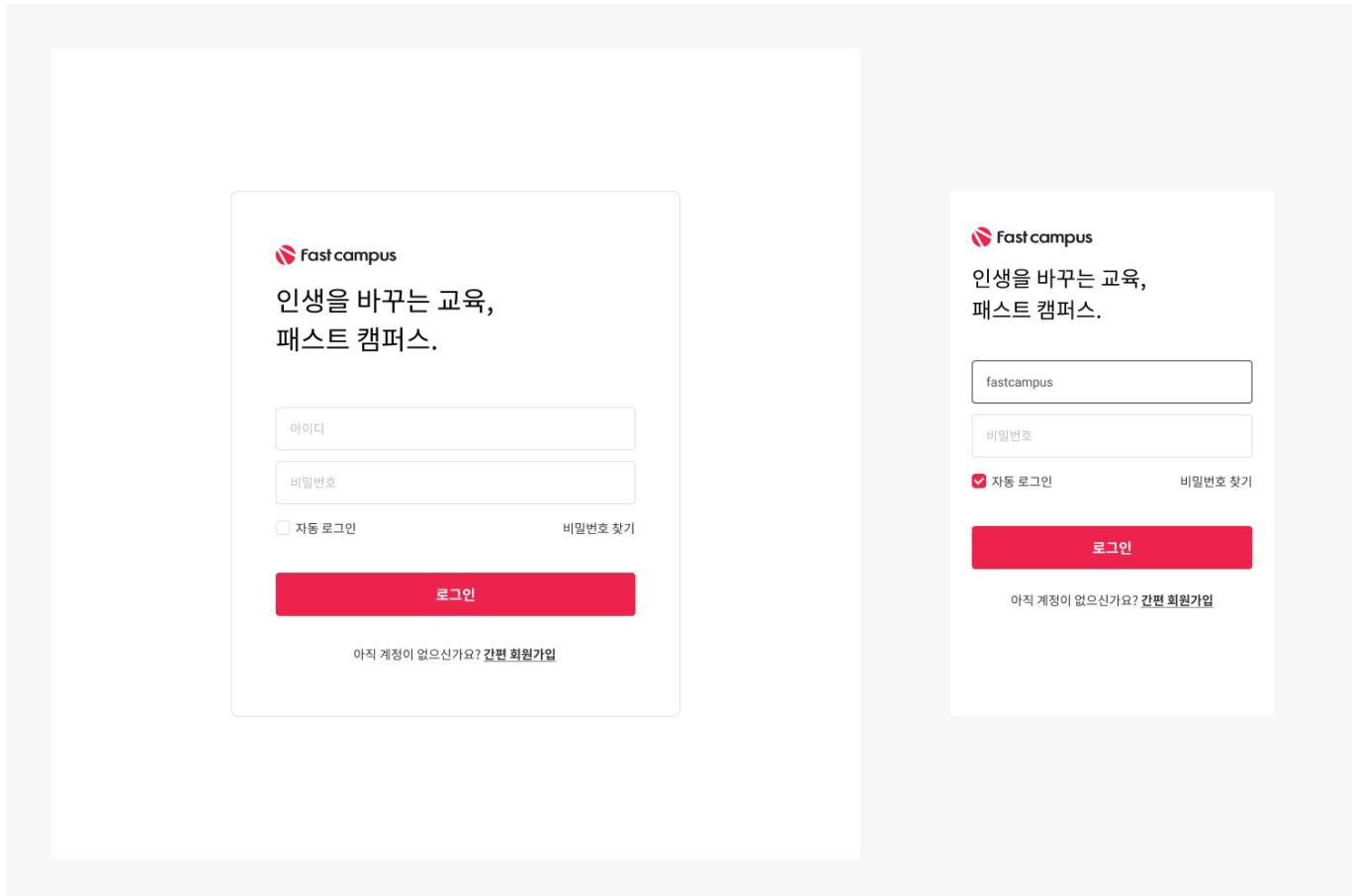
브라우저 사이즈에 따라 레이아웃이 변경되는 것을 유의해주세요.
media query를 사용하여 반응형 디자인 스타일을 작성해주세요.

2. 디자인 가이드에 따라 컨텐츠를 채워주세요.
3. 디자인 가이드의 수치에 따라 스타일을 작성해주세요.

로그인 컨테이너의 가로 길이가 브라우저 사이즈에 따라 적절하게 변경될 수 있도록 해주세요.
체크박스 input에서 이미지를 활용하는 것에 유의해주세요.

문제

Figma에서 확인하기



주요 학습 키워드

디자인 가이드를 참고하여 스타일 작성하기
반응형 스타일 작성하기

작성해주셔야 하는 question 파일경로

./question/question.html
./question/question.css
./question/question.scss

실행 방법

경로 ./question/question.html
index.html 열기

Case9 : Login - 출제자 해설코드

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <title>03-login (solution)</title>
    <link href="https://fonts.googleapis.com/css2?family=Noto+Sans+KR:wght@400;700&display=swap" rel="stylesheet" />
    <link rel="stylesheet" href="reset.css" />
    <link rel="stylesheet" href="solution.css" />
  </head>
  <body>
    <div class="login-container">
      <div class="login">
        <div class="login-logo">
          
        </div>
        <h1 class="login-title">
          인생을 바꾸는 교육,<br />
          패스트 캠퍼스.
        </h1>

        <form class="login-form-container">
          <div class="login-form-input-container">
            <input type="text" class="login-form-input" placeholder="0|0|0|0" />
            <input
              type="password"
              class="login-form-input"
              placeholder="비밀번호"
            />
          </div>

          <div class="login-option-container">
            <div class="login-form-checkbox-container">
              <input
                type="checkbox"
                class="login-form-checkbox"
                id="auto-login"
              />
              <label for="auto-login" class="login-form-checkbox-label">
                자동 로그인
              </label>
            </div>
          </div>
        </form>
      </div>
    </div>
  </body>

```

```

<div class="login-find-password-container">
    <a class="login-find-password-link" href="#">비밀번호 찾기</a>
</div>
</div>

<div class="login-button-container">
    <button class="login-button">로그인</button>
</div>

<div class="login-signup-container">
    아직 계정이 없으신가요?
    <a href="#" class="login-signup-link">간편 회원가입</a>
</div>
</form>
</div>
</div>
</body>
</html>

```

// figma : <https://www.figma.com/file/9FXkniEMPgZKtJY4GwP60z/Input?node-id=0%3A3>

```

// color palette
$gray1: #333333;
$gray2: #4f4f4f;
$gray3: #828282;
$gray4: #bdbdbd;
$gray5: #e0e0e0;
$gray6: #f2f2f2;
$gray7: #f9f9f9;
$primary: #ed234b;

// color variable rename
$text-color: $gray1;
$input-text-color: $gray2;
$placeholder-text-color: $gray4;
$border-color: $gray5;
$border-focus-color: $gray1;
$button-primary-color: $primary;

// default
html,
body {
    font-family: "Noto Sans KR", sans-serif;
    font-size: 14px;
    line-height: 20px;
    color: $text-color;
}

.login-container {
    display: flex;

```

```
justify-content: center;
align-items: center;
height: 100vh;

.login {
  border: 1px solid $border-color;
  border-radius: 8px;
  padding: 60px 50px;
  width: 100%;
  max-width: 500px;

  .login-title {
    font-size: 30px;
    color: black;
    line-height: 43px;
    margin: 16px 0 56px;
  }

  .login-form-container {
    display: flex;
    flex-direction: column;

    .login-form-input-container {
      display: flex;
      flex-direction: column;

      .login-form-input {
        border: 1px solid $border-color;
        border-radius: 4px;
        padding: 16px;
        color: $input-text-color;
        outline: none;

        &::placeholder {
          color: $placeholder-text-color;
        }

        &:focus {
          border: 1px solid $border-focus-color;
        }

        &:not(:last-child) {
          margin-bottom: 12px;
        }
      }
    }
  }
}

.login-option-container {
  display: flex;
  justify-content: space-between;
  margin: 18px 0 42px;

  // 기본 HTML 체크박스 대신 이미지 활용
  .login-form-checkbox-container {
    .login-form-checkbox {
```

```
        display: none;

    + label {
        background: url("./images/checkbox-default.png") no-repeat;
        height: 16px;
        display: inline-block;
        padding-left: 22px;
        line-height: 16px;
    }

    &:checked + label {
        background: url("./images/checkbox-checked.png") no-repeat;
    }
}

.login-find-password-container {
    .login-find-password-link {
        text-decoration: none;
        color: $text-color;
    }
}

.login-button-container {
    .login-button {
        width: 100%;
        height: 48px;
        line-height: 48px;
        color: white;
        border: none;
        border-radius: 4px;
        font-size: 16px;
        background: $button-primary-color;
        font-weight: bold;
        outline: none;
    }
}

.login-signup-container {
    margin-top: 32px;
    text-align: center;

    .login-signup-link {
        color: $text-color;
        font-weight: bold;
    }
}

@media screen and (max-width: 769px) {
    .login-container {
        align-items: flex-start;
    }
}
```

```
.login {
    border: none;
    padding: 40px 20px;

    .login-title {
        font-size: 24px;
        line-height: 35px;
        margin: 16px 0 40px;
    }
}

}

}

}
```

<Case9 : Login> 문제와 같이 보면 좋은 팁

다른강사님께서 해당 출제문제를 본 후, 같이 확인하면 좋겠다고 주신 의견입니다.

HTML

로그인과 같은 웹페이지에는 <input> 요소가 많이 사용되고 있습니다. 이때 아이디 또는 이메일 주소 그리고 비밀번호 등이 기본적으로 요구되는 값의 유형일 것입니다.

이런 유형의 경우 일반적인 아이디는 type="text", 이메일 유형의 아이디는 type="email", 비밀번호는 type="password"를 사용하는 것이 좋습니다.

특히 type="email" 유형의 경우 HTML5에 새롭게 등장한 값으로 입력 값이 이메일 형식이 아니면 오류 메시지를 보여줍니다.

<input> 요소나 type="email" 값의 경우 자세한 설명이나 사용 사례는 아래 MDN을 참고하시기 바랍니다.

🔗 <https://developer.mozilla.org/ko/docs/Web/HTML/Element/Input>

🔗 <https://developer.mozilla.org/en-US/docs/Web/HTML/Element/input/email>

또한 입력 값이 필수인지 아닌지 이해할 수 있도록 HTML5의 required 속성을 사용하는 것도 도움이 될 수 있습니다. 만약 required 속성을 사용하지 못하는 상황이라면 스크린리더 사용자들이 해당 입력 값이 필수임을 인지할 수 있도록 aria-required 속성을 사용하는 것이 도움이 됩니다.

대다수의 개발자들이 해당 정보가 필수임을 시각적으로만 표현하려 하지만 화면을 볼 수 없는 보조기기 등의 사용자를 고려하여 프로그래밍 적 해결책을 찾는 것 또한 중요할 수 있다는 점을 잊지 마시기 바랍니다.

🔗 <https://www.w3.org/TR/WCAG20-TECHS/ARIA2.html>

🔗 https://developer.mozilla.org/en-US/docs/Web/Accessibility/ARIA/ARIA_Techniques/Using_the_aria-required_attribute

CSS

체크박스와 같은 경우 대다수 커스텀 디자인을 선호합니다. 디자인 시안을 바탕으로 구현하기 위해서는 체크박스 커스텀이 필수일 수 있습니다.

다만 디자인에만 초점을 맞추다보면 키보드 접근이 불가한 경우를 종종 볼 수 있는데 이는 웹접근성 관점을 고려하지 않은 제작 방식이라 할 수 있습니다.

커스텀 체크박스 디자인의 핵심은 체크박스 입력서식인 <input> 요소에 display: none을 적용하지 않는 것입니다.

많은 개발자가 <input> 요소에 display: none;을 적용한 후 <label> 등의 요소에 커스텀 체크박스를 배경이미지로 사용합니다. 그러나 이렇게 개발할 경우 키보드 접근과 조작 자체가 불가능해 집니다.

그렇다면 어떤 방법을 사용하면 키보드로도 접근 및 조작 가능한 커스텀 체크박스를 만들 수 있을까요? `<input>` 요소에 `display: none` 대신 다음과 같은 트릭을 활용할 수 있습니다.

```
.login-form-checkbox {
    opacity: 0;
    width: 16px;
    height: 16px;
    padding: 0;
    position: absolute;
    z-index: 100;
    background: transparent;
    cursor: pointer;
}
```

이제 키보드로 접근하여 스페이스 키를 눌러보면 `<label>` 요소에 미리 적용된 배경이미지가 변경되는 것을 확인할 수 있습니다. 그렇지만 여기서도 문제가 발생합니다.

아이디나 비밀번호 같은 입력서식과 달리 체크박스 입력서식은 `opacity: 0`을 지정했기 때문에 키보드가 접근하긴 하지만 포커스가 시각적으로 표시되지 않습니다.

이 문제를 해결하기 위해 `.login-form-checkbox:focus + label` 요소를 선택자로 활용하여 아웃라인 디자인을 추가할 수 있습니다.

다만 이 경우 레이블 전체에 아웃라인이 발생하기 때문에 체크박스 배경에만 적용하고자 할 경우 배경이미지를 `<label>` 요소 자체가 아닌 다른 요소에 적용하는 것이 도움이 됩니다.

다른 요소에 배경을 적용할 때 아래와 같은 형태의 마크업을 구성하여 문제를 해결해 보시기 바랍니다.

```
<div class="login-form-checkbox-container">
    <input type="checkbox" class="login-form-checkbox" id="auto-login">
    <span class="custom-checkbox"></span>
    <label for="auto-login" class="login-form-checkbox-label">자동로그인</label>
</div>
```

간혹 실무에서 커스텀 체크박스 제작을 위해 `<input>` 요소가 아닌 `` 등의 요소를 사용하는 경우가 있습니다.

이 경우 키보드 접근을 위해 `tabindex="0"`과 "WAI-ARIA의 `role="checkbox"`"를 활용하여 키보드 사용자나 보조기기 사용자가 이를 이해하고 조작하게 할 수도 있습니다.

그러나 `<input>`이 아닌 `` 등의 요소를 사용할 경우 입력값에 대한 처리나 마우스 및 키보드 관련 조작 기능을 모두 자바스크립트에서 처리를 해야한다는 것을 기억하시기 바랍니다.

`role="checkbox"`에 대한 참고사이트는 MDN을 살펴보시기 바랍니다.

🔗 https://developer.mozilla.org/en-US/docs/Web/Accessibility/ARIA/Roles/checkbox_role

Case10 : Signup

케이스 주제

Figma에서 제공되는 수치를 확인해서 디자인과 같은 반응형 UI를 마크업합니다.
아래의 요구사항을 참고해서 모든 브라우저에서 같은 디자인이 보이도록 해야합니다.

기능 요구사항

- 디자인 가이드를 참고하여 브라우저 사이즈에 따라 적절하게 위치할 수 있는 레이아웃 스타일을 작성해주세요.

브라우저 사이즈에 따라 레이아웃이 변경되는 것을 유의해주세요.
media query를 사용하여 반응형 디자인 스타일을 작성해주세요.

- 디자인 가이드에 따라 컨텐츠를 채워주세요.
- 디자인 가이드의 수치에 따라 스타일을 작성해주세요.

회원가입 컨테이너의 가로 길이가 브라우저 사이즈에 따라 적절하게 변경될 수 있도록 해주세요.
체크박스 input에서 이미지를 활용하는 것에 유의해주세요.

문제

 Figma에서 확인하기

 example

주요 학습 키워드

디자인 가이드를 참고하여 스타일 작성하기
반응형 스타일 작성하기

작성해주셔야 하는 question 파일경로

```
./question/question.html  
./question/question.css  
./question/question.scss
```

실행 방법

경로 ./question/question.html
index.html 열기

Case10 : Signup - 출제자 해설코드

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <title>04-signup (solution)</title>
    <link href="https://fonts.googleapis.com/css2?family=Noto+Sans+KR:wght@400;700&display=swap" rel="stylesheet" />
    <link rel="stylesheet" href="reset.css" />
    <link rel="stylesheet" href="solution.css" />
  </head>
  <body>
    <div class="signup-container">
      <div class="signup">
        <div class="signup-logo">
          
        </div>
        <h1 class="signup-title">
          인생을 바꾸는 교육,<br />
          패스트 캠퍼스.
        </h1>

        <form class="signup-form-container">
          <div class="signup-form-input-container">
            <input type="text" class="signup-form-input" placeholder="이름" />
            <div class="signup-form-email-row">
              <input
                type="email"
                class="signup-form-input"
                placeholder="이메일"
              />
              <button class="signup-form-send-validation-email-button">
                인증메일 발송
              </button>
            </div>
            <input
              type="tel"
              class="signup-form-input"
              placeholder="휴대폰 번호 (숫자만)"
            />
            <input
              type="password"
              class="signup-form-input"
            />
          </div>
        </form>
      </div>
    </div>
  </body>

```

```
        placeholder="비밀번호"
    />
<input
    type="password"
    class="signup-form-input"
    placeholder="비밀번호 확인"
/>
</div>

<div class="signup-option-container">
<div class="signup-option-row">
    <div class="signup-form-checkbox-container">
        <input
            type="checkbox"
            class="signup-form-checkbox"
            id="service-policy"
        />
        <label for="service-policy" class="signup-form-checkbox-label">
            서비스 이용약관 동의 (필수)
        </label>
    </div>
    <a href="#" class="signup-option-policy-link">보기</a>
</div>

<div class="signup-option-row">
    <div class="signup-form-checkbox-container">
        <input
            type="checkbox"
            class="signup-form-checkbox"
            id="service-policy"
        />
        <label for="service-policy" class="signup-form-checkbox-label">
            개인정보 처리방침 동의 (필수)
        </label>
    </div>
    <a href="#" class="signup-option-policy-link">보기</a>
</div>

<div class="signup-option-row">
    <div class="signup-form-checkbox-container">
        <input
            type="checkbox"
            class="signup-form-checkbox"
            id="service-policy"
        />
        <label for="service-policy" class="signup-form-checkbox-label">
            패스트캠퍼스의 소식과 다양한 안내 (선택)
        </label>
    </div>
    <a href="#" class="signup-option-policy-link">보기</a>
</div>
</div>

<div class="signup-button-container">
    <button class="signup-button">회원가입</button>
```

```
        </div>
    </form>
</div>
</div>
</body>
</html>
```

```
// figma : https://www.figma.com/file/9FXkniEMPgZKtJY4GwP60z/Input?node-id=0%3A3

// color palette
$gray1: #333333;
$gray2: #4f4f4f;
$gray3: #828282;
$gray4: #bdbdbd;
$gray5: #e0e0e0;
$gray6: #f2f2f2;
$gray7: #f9f9f9;
$primary: #ed234b;

// color variable rename
$text-color: $gray1;
$input-text-color: $gray2;
$text-link-color: $gray4;
$placeholder-text-color: $gray4;
$border-color: $gray5;
$border-focus-color: $gray1;
$button-primary-color: $primary;
$button-gray-color: $gray2;

// default
html,
body {
    font-family: "Noto Sans KR", sans-serif;
    font-size: 14px;
    line-height: 20px;
    color: $text-color;
}

.signup-container {
    display: flex;
    justify-content: center;
    align-items: center;
    height: 100vh;

    .signup {
        border: 1px solid $border-color;
        border-radius: 8px;
        padding: 60px 50px;
        width: 100%;
        max-width: 500px;
```

```
.signup-title {
  font-size: 30px;
  color: black;
  line-height: 43px;
  margin: 16px 0 56px;
}

.signup-form-container {
  display: flex;
  flex-direction: column;

  .signup-form-input-container {
    display: flex;
    flex-direction: column;

    .signup-form-input {
      border: 1px solid $border-color;
      border-radius: 4px;
      padding: 16px;
      color: $input-text-color;
      outline: none;
      width: 100%;

      &::placeholder {
        color: $placeholder-text-color;
      }

      &:focus {
        border: 1px solid $border-focus-color;
      }

      &:not(:last-child) {
        margin-bottom: 12px;
      }
    }
  }

  .signup-form-email-row {
    display: flex;

    .signup-form-send-validation-email-button {
      padding: 16px;
      background: $button-gray-color;
      border-radius: 4px;
      border: none;
      color: white;
      margin-left: 8px;
      min-width: max-content;
      height: 48px;
      outline: none;
    }
  }
}

.signup-option-container {
  display: flex;
```

```
flex-direction: column;
justify-content: space-between;
margin: 18px 0 42px;

.signup-option-row {
  width: 100%;
  display: flex;
  justify-content: space-between;
  align-items: center;

  &:not(:last-child) {
    margin-bottom: 12px;
  }

  .signup-option-policy-link {
    font-size: 12px;
    color: $text-link-color;
  }
}

// 기본 HTML 체크박스 대신 이미지 활용
.signup-form-checkbox-container {
  .signup-form-checkbox {
    display: none;

    + label {
      background: url("./images/checkbox-default.png") no-repeat;
      height: 16px;
      display: inline-block;
      padding-left: 22px;
      line-height: 16px;
    }

    &:checked + label {
      background: url("./images/checkbox-checked.png") no-repeat;
    }
  }
}

.signup-button-container {
  .signup-button {
    width: 100%;
    height: 48px;
    line-height: 48px;
    color: white;
    border: none;
    border-radius: 4px;
    font-size: 16px;
    background: $button-primary-color;
    font-weight: bold;
    outline: none;
  }
}
```

```
}

}

@media screen and (max-width: 769px) {
  .signup-container {
    align-items: flex-start;

    .signup {
      border: none;
      padding: 40px 20px;

      .signup-title {
        font-size: 24px;
        line-height: 35px;
        margin: 16px 0 40px;
      }
    }
  }
}
```

<Case10 : Sign up> 문제와 같이 보면 좋은 팁

다른강사님께서 해당 출제문제를 본 후, 같이 확인하면 좋겠다고 주신 의견입니다.

HTML

회원가입과 같은 웹페이지에는 `<input>` 요소가 많이 사용되고 있습니다. 이때 아이디 또는 이메일 주소 그리고 비밀번호 등이 기본적으로 요구되는 값의 유형일 것입니다.

이런 유형의 경우 일반적인 아이디는 `type="text"`, 이메일 유형의 아이디는 `type="email"`, 비밀번호는 `type="password"`를 사용하는 것이 좋습니다.

특히 `type="email"` 유형의 경우 HTML5에 새롭게 등장한 값으로 입력 값이 이메일 형식이 아니면 오류 메시지를 보여줍니다.

`<input>` 요소나 `type="email"` 값의 경우 자세한 설명이나 사용 사례는 아래 MDN을 참고하시기 바랍니다.

🔗 <https://developer.mozilla.org/ko/docs/Web/HTML/Element/Input>

🔗 <https://developer.mozilla.org/en-US/docs/Web/HTML/Element/input/email>

또한 입력 값이 필수인지 아닌지 이해할 수 있도록 HTML5의 `required` 속성을 사용하는 것도 도움이 될 수 있습니다. 만약 `required` 속성을 사용하지 못하는 상황이라면 스크린리더 사용자들이 해당 입력 값이 필수임을 인지할 수 있도록 `aria-required` 속성을 사용하는 것 이 도움이 됩니다.

대다수의 개발자들이 해당 정보가 필수임을 시각적으로만 표현하려 하지만 화면을 볼 수 없는 보조기기 등의 사용자를 고려하여 프로그래밍 적 해결책을 찾는 것 또한 중요할 수 있다는 점을 잊지 마시기 바랍니다.

🔗 <https://www.w3.org/TR/WCAG20-TECHS/ARIA2.html>

🔗 https://developer.mozilla.org/en-US/docs/Web/Accessibility/ARIA/ARIA_Techniques/Using_the_aria-required_attribute

HTML에 문법 오류가 발견되고 있습니다. `id` 사용의 경우 문서에서 중복된 이름을 사용할 수 없는데 현재 `signup.html`의 약관 관련한 체크박스에 `service-policy`라는 동일한 아이디를 사용하고 있어 문제가 있습니다. 이 경우 사용자는 개의 체크박스 중 하나만 선택할 수 있게 됩니다.

이런 오류를 문서단위로 검증하고자 할 경우

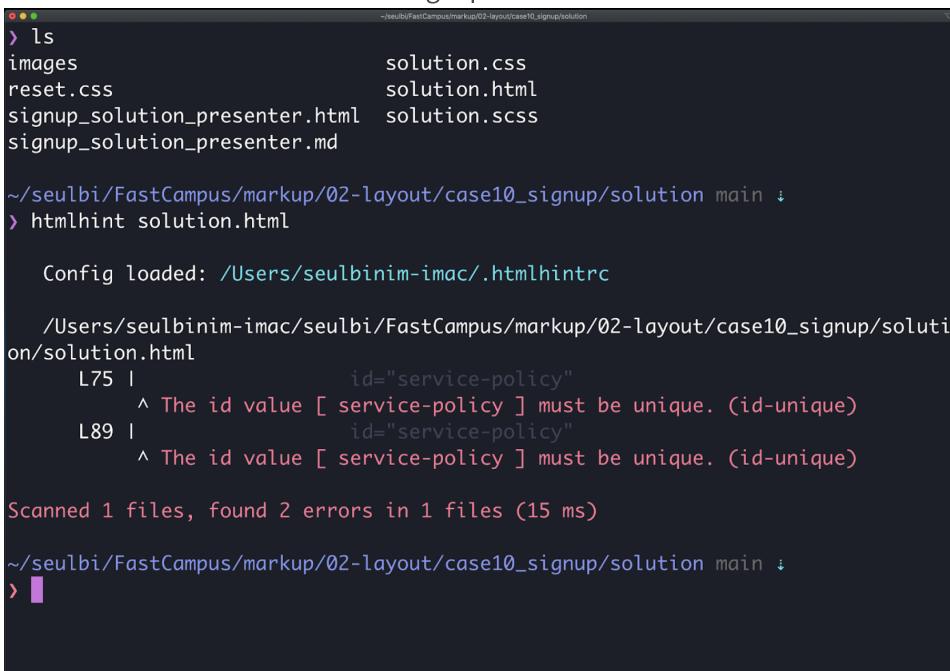
 <https://validator.w3.org/>

를 활용하시거나 htmlhint라는 node의 패키지를 설치해서 검증해 보는 것이 필요합니다.

htmlhint의 경우 특정 폴더 내 모든 HTML파일에 대한 일괄 검수도 가능합니다. 일괄 검수 시 명령어는 다음과 같습니다.

```
htmlhint **/*.html
```

아래는 CLI 환경에서 htmlhint로 검사해 본 signup.html 파일의 결과입니다.



```
> ls
images                      solution.css
reset.css                   solution.html
signup_solution_presenter.html  solution.scss
signup_solution_presenter.md

~/seulbi/FastCampus/markup/02-layout/case10_signup/solution main ↓
> htmlhint solution.html

Config loaded: /Users/seulbinim-imac/.htmlhintrc

/Users/seulbinim-imac/seulbi/FastCampus/markup/02-layout/case10_signup/soluti
on/solution.html
L75 |           id="service-policy"
     ^ The id value [ service-policy ] must be unique. (id-unique)
L89 |           id="service-policy"
     ^ The id value [ service-policy ] must be unique. (id-unique)

Scanned 1 files, found 2 errors in 1 files (15 ms)

~/seulbi/FastCampus/markup/02-layout/case10_signup/solution main ↓
> █
```

CSS

체크박스와 같은 경우 대다수 커스텀 디자인을 선호합니다. 디자인 시안을 바탕으로 구현하기 위해서는 체크박스 커스텀이 필수일 수 있습니다.

다만 디자인에만 초점을 맞추다보면 키보드 접근이 불가한 경우를 종종 볼 수 있는데 이는 웹접근성 관점을 고려하지 않은 제작 방식이라 할 수 있습니다.

커스텀 체크박스 디자인의 핵심은 체크박스 입력서식인 `<input>` 요소에 `display: none`을 적용하지 않는 것입니다.

많은 개발자가 `<input>` 요소에 `display: none`;을 적용한 후 `<label>` 등의 요소에 커스텀 체크박스를 배경이미지로 사용합니다. 그러나 이렇게 개발할 경우 키보드 접근과 조작 자체가 불가능해집니다.

그렇다면 어떤 방법을 사용하면 키보드로도 접근 및 조작 가능한 커스텀 체크박스를 만들 수 있을까요? `<input>` 요소에 `display: none` 대신 다음과 같은 트릭을 활용할 수 있습니다.

```
.signup-form-checkbox {
    opacity: 0;
    width: 16px;
    height: 16px;
    padding: 0;
    position: absolute;
    z-index: 100;
    background: transparent;
    cursor: pointer;
}
```

이제 키보드로 접근하여 스페이스 키를 눌러보면 `<label>` 요소에 미리 적용된 배경이미지가 변경되는 것을 확인할 수 있습니다. 그렇지만 여기서도 문제가 발생합니다.

아이디나 비밀번호 같은 입력서식과 달리 체크박스 입력서식은 opacity: 0을 지정했기 때문에 키보드가 접근하기 하지만 포커스가 시각적으로 표시되지 않습니다.

이 문제를 해결하기 위해 .signup-form-checkbox:focus + label 요소를 선택자로 활용하여 아웃라인 디자인을 추가할 수 있습니다.

다만 이 경우 레이어 전체에 아웃라인이 발생하기 때문에 체크박스 배경에만 적용하고자 할 경우 배경이미지를 <label> 요소 자체가 아닌 다른 요소에 적용하는 것이 도움이 됩니다.

다른 요소에 배경을 적용할 때 아래와 같은 형태의 마크업을 구성하여 문제를 해결해 보시기 바랍니다.

```
<div class="signup-form-checkbox-container">
  <input
    type="checkbox"
    class="signup-form-checkbox"
    id="service-policy"
  />
  <span class="custom-checkbox"></span>
  <label for="service-policy" class="signup-form-checkbox-label">
    개인정보 처리방침 동의 (필수)
  </label>
</div>
```

간혹 실무에서 커스텀 체크박스 제작을 위해 <input> 요소가 아닌 등의 요소를 사용하는 경우가 있습니다.

이 경우 키보드 접근을 위해 tabindex="0"과 "WAI-ARIA의 role="checkbox"를 활용하여 키보드 사용자나 보조기기 사용자가 이를 이해하고 조작하게 할 수도 있습니다.

그러나 <input>이 아닌 등의 요소를 사용할 경우 입력값에 대한 처리나 마우스 및 키보드 관련 조작 기능을 모두 자바스크립트에서 처리를 해야한다는 것을 기억하시기 바랍니다.

role="checkbox"에 대한 참고사이트는 MDN을 살펴보시기 바랍니다.

☞ https://developer.mozilla.org/en-US/docs/Web/Accessibility/ARIA/Roles/checkbox_role

CSS 빌드 결과물의 선택자가 너무 복잡한 것 같습니다. 이렇게 복잡한 선택자를 사용하면 추후 CSS를 재정의 해야하는 상황에서 문제를 일으킬 소지가 있습니다.

Sass는 이를 방지하기 위해 효율적으로 중첩패턴을 사용할 수 있는 & 지시자가 있기 때문에 이를 잘 활용하여 재사용이나 확장 가능한 코드를 짜는 것이 필요합니다.

예를 들어 다음과 같은 방식으로 수정하면 좋을 것 같습니다.

```
// 중첩 패턴 BAD Case
.signup{
  .signup-container{
    ...
  }
  .signup-form-input{
    ...
  }
}

// 중첩 패턴 GOOD Case
.signup{
  &-container{
    ...
  }
  &-form-input{
```

```
...  
}  
}  
}
```

그리고 미디어쿼리를 사용할 경우 지금처럼 모든 디바이스에 적용될 코드를 선언한 후 모바일 디바이스에서 재정의하는 방식은 모바일 최적화 관점에서는 좋지 않을 수 있습니다. 이미 모바일에서 모든 코드를 적용한 후 다시 또 재정의 한 코드를 해석해서 다시 적용해야 하니까요. 이럴 땐, 아래와 같이 하나의 선택자 내부에서 미디어쿼리 믹스인을 호출하는 방식도 사용해볼 수 있습니다.

```
.signup-container {  
    // 모든 디바이스 공통  
    display: flex;  
    justify-content: center;  
    height: 100vh;  
  
    // 모바일 디바이스  
    @include mobile{  
        align-items: flex-start  
    }  
  
    // 데스크탑 디바이스  
    @include desktop{  
        align-items: center;  
    }  
}
```

Case19 : Position

케이스 주제

Figma에서 제공되는 화면을 확인하여 position 레이아웃을 만듭니다.

아래의 요구사항을 참고해서 모든 브라우저에서 같은 디자인이 보이도록 해야합니다.

기능 요구사항

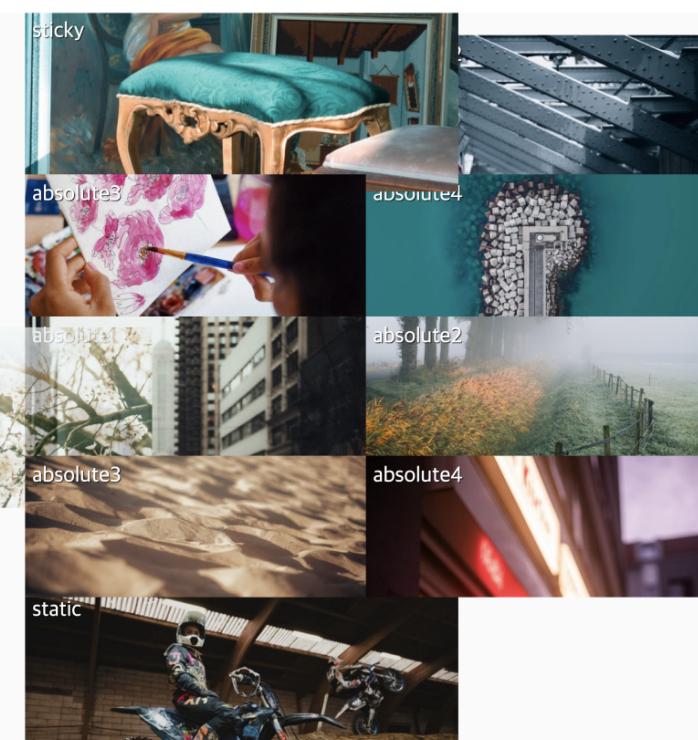
1. example.png를 확인하여 레이아웃을 Position으로 구성합니다.
2. relative 엘리먼트의 41.25%는 화면이 줄어도 비율이 일정하게 하기 위함입니다.
3. 박스의 순서는 figma를 확인해주세요.
4. 스크롤을 내릴 시에 sticky 및 fixed 엘리먼트가 absolute3번 보다는 아래에 있어야 합니다.
5. HTML파일 내에 div 태그에 적절히 position과 관련된 클래스를 부여하여 CSS로 제어해보세요.

문제

Figma에서 확인하기

Position

스크롤을 내릴 시에 sticky 및 fixed 엘리먼트가 absolute3번 보다는 아래에 있어야 합니다.



The grid contains 12 images arranged in a 4x3 grid. The columns are labeled from left to right: 'sticky', 'absolute3', 'absolute4', 'absolute2', 'absolute3', 'absolute4', 'static', 'fixed', 'absolute3', 'absolute4', 'absolute2', and 'absolute3'. The images show various examples of how these positions affect element placement relative to the scroll position.

주요 학습 키워드

position으로 비율이 일정한 레이아웃 만들기
static, relative, absolute, sticky 사용해 보기
z-index를 적절히 사용하여 레이아웃이 쌓이는 기준점 파악하기

작성해주셔야 하는 question 파일경로

```
./question/question.html  
./question/question.css  
./question/question.scss
```

실행 방법

경로 [./question/question.html](#)

[index.html 열기](#)

Case19 : Position - 출제자 해설코드

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>4-2 Position (solution)</title>
    <link rel="stylesheet" href="solution.css">
</head>
<body>
    <h1>position</h1>
    <div class="position_wrap">
        <div class="item_relative">
            <span class="item_absolute"></span>
            <span class="item_absolute"></span>
            <span class="item_absolute"></span>
            <span class="item_absolute"></span>
        </div>
        <div class="item_sticky">
            
        </div>
        <div class="item_relative">
            <span class="item_absolute"></span>
            <span class="item_absolute"></span>
            <span class="item_absolute"></span>
            <span class="item_absolute"></span>
        </div>
        <div class="item_relative">
            <span class="item_absolute"></span>
            <span class="item_absolute"></span>
            <span class="item_absolute"></span>
            <span class="item_absolute"></span>
        </div>
        <div class="item_fixed">
            
        </div>
        <div class="item_static">
            
        </div>
    </div>
</body>
</html>
```

```
@charset "utf-8";\n\n/* variables */\n\n$w100: 100%;\n$half: 50%;\n$basis_space: 40px;\n\n/* reset */\n\n*,\n*:before,\n*:after {\n    box-sizing: border-box;\n}\n\n*:focus {\n    outline: none;\n}\n\nhtml,\nbody {\n    position: relative;\n    width: $w100;\n    height: $w100;\n    padding: 0;\n    margin: 0;\n}\n\nh1 {\n    text-align: center;\n    margin: 35px auto 0;\n}\n\nimg {\n    display: block;\n    max-width: 100%;\n}\n\n.position_wrap {\n    display: block;\n    position: relative;\n    width: $w100;\n    height: 1950px;\n    max-width: 1024px;\n    margin: $basis_space auto;\n    padding: $basis_space;\n    background-color: #fafafa;\n    *:before {\n        content: 'static';\n    }\n}
```

```
display: block;
position: absolute;
color: #fff;
font-size: 30px;
padding: 10px;
text-shadow: 1px 1px 1px rgba(0, 0, 0, .9);
}

.item_relative {
    padding-bottom: 41.25%;
    position: relative;
    top: 10px;
    &:before {
        content: 'relative';
    }
}

.item_absolute {
    position: absolute;
    left: 0;
    top: 0;
    width: 50%;
    &:before {
        content: 'absolute';
    }
    &:nth-of-type(2) {
        left: auto;
        right: 0;
        top: 0;
        bottom: auto;
        &:before {
            content: 'absolute2';
        }
    }
    &:nth-of-type(3) {
        left: 0;
        right: auto;
        bottom: 0;
        top: auto;
        z-index: 1;
        &:before {
            content: 'absolute3';
        }
    }
    &:nth-of-type(4) {
        top: auto;
        right: 0;
        bottom: 0;
        left: auto;
        &:before {
            content: 'absolute4';
        }
    }
}
```

```
.item_static {
  position: static;
}

.item_fixed {
  position: fixed;
  left: 0;
  top: 50%;
  height: 150px;
  margin-top: -75px;
  opacity: .7;
  img {
    height: 100%;
  }
  &:before {
    content: 'fixed';
  }
}

.item_sticky {
  position: sticky;
  top: 0;
  left: 0;
  z-index: 1;
  &:before {
    content: 'sticky';
  }
}

}
```

Case20 : Float

케이스 주제

Figma에서 제공되는 화면을 확인하여 float 레이아웃 갤러리를 만듭니다.

아래의 요구사항을 참고해서 모든 브라우저에서 같은 디자인이 보이도록 해야합니다.

기능 요구사항

1. example.png를 확인하여 레이아웃을 Float으로 구성합니다.
2. 각 박스의 높이는 300px(큰 박스), 150px(작은 박스)가 적절히 섞이도록 구성해야 합니다.
3. 박스의 순서는 figma를 확인해주세요.

문제

Float



주요 학습 키워드

float으로 갤러리 레이아웃 만들기
float 해제 및 필요한 곳에만 HTML 및 CSS 사용하기
float으로 다양한 브라우저에서도 같은 레이아웃 만들기

작성해주셔야 하는 question 파일경로

./question/question.html
./question/question.css
./question/question.scss

실행 방법

경로 ./question/question.html

[index.html 열기](#)

Case20 : Float - 출제자 해설코드

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>4-1 Float (solution)</title>
    <link rel="stylesheet" href="solution.css">
</head>
<body>
    <h1>float</h1>
    <div class="float_wrap">
        <div class="item_wrap">
            <!-- 각 레이아웃이 나눠지는 만큼 큰 덩어리 단위로 우선 구분지어줍니다. -->
            <div class="item large"></div>
            <!-- 큰 덩어리 단위로 감싸야 하는 이유가 충분하다면 감싸야합니다. -->
            <div class="item_child_wrap">
                <div class="item small"></div>
                <div class="item small"></div>
            </div>
            <div class="item large"></div>
        </div>
        <div class="item_wrap">
            <div class="item_child_wrap">
                <div class="item half small"></div>
                <div class="item half small"></div>
                <div class="item small w-100"></div>
            </div>
            <div class="item large"></div>
        </div>
    </div>
</body>
</html>
```

```
@charset "utf-8";

/* variables */

$w100: 100%;
$half: 50%;
$third: 33.33%;
```

```
$large_box: 300px;
$small_box: 150px;
$basis_space: 40px;
$padding: 15px;
$box-shadow: 14px 17px 34px -20px rgba(0, 0, 0, 0.75);

/* reset */

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

*:focus {
    outline: none;
}

html,
body {
    position: relative;
    width: $w100;
    height: $w100;
    padding: 0;
    margin: 0;
}

h1 {
    text-align: center;
    margin: 35px auto 0;
}

.float_wrap {
    display: block;
    width: $w100;
    max-width: 1024px;
    margin: $basis_space auto;
    padding: $basis_space;
    background-color: #fafafa;
    counter-reset: items;
    .item_wrap {
        display: block;
        /* float을 해제하는 방법들은 다양합니다. */
        &:after {
            content: '';
            display: block;
            clear: both;
        }
        .item {
            float: left;
            width: $third;
            position: relative;
        }
    }
}
```

```
height: $small_box;
padding: $padding;
box-sizing: border-box;
overflow: hidden;
&.large {
    height: $large_box;
}
&.small {
    height: $small_box;
}
img {
    position: absolute;
    left: $half;
    top: $half;
    height: 100%;
    -webkit-transform: translate3d(-$half, -$half, 0);
    transform: translate3d(-$half, -$half, 0);
}
&:before {
    counter-increment: items;
    content: counter(items);
    position: relative;
    z-index: 1;
    color: #fff;
}
.item_child_wrap {
    display: block;
    float: left;
    width: $third;
    .item {
        float: none;
        width: 100%;
        &.half {
            width: $half;
        }
    }
}
&:nth-of-type(2) {
    .item_child_wrap {
        width: 66.66%;
        .item {
            float: left;
        }
        .w-100 img {
            width: 100%;
        }
    }
    .item:last-of-type {
        float: right;
    }
}
```

```
    }  
}
```

Case21 : Flex

케이스 주제

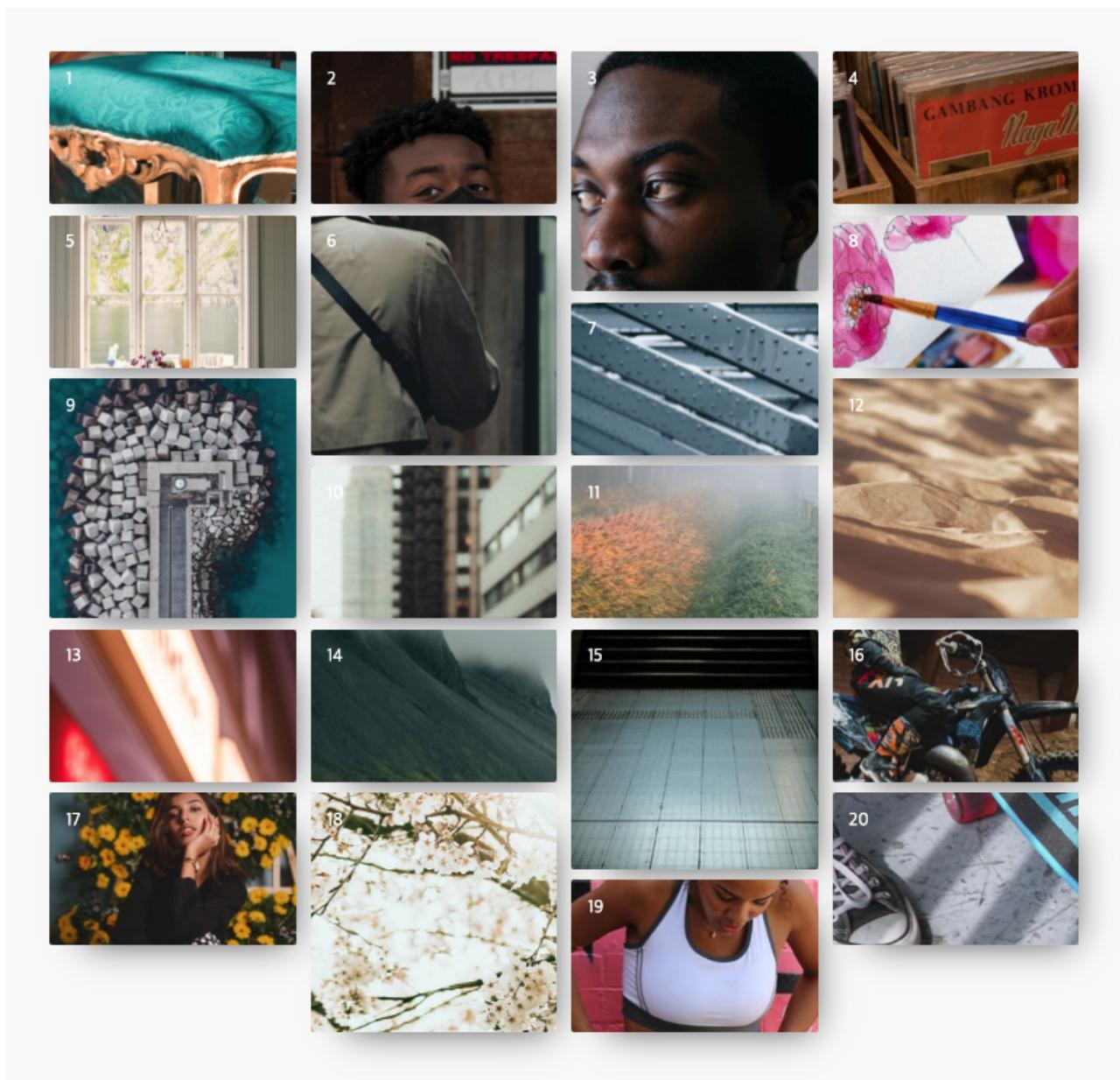
Figma에서 제공되는 화면을 확인하여 Masonry 갤러리를 만듭니다.

아래의 요구사항을 참고해서 모든 브라우저에서 같은 디자인이 보이도록 해야합니다.

기능 요구사항

1. 지정된 브라우저 해상도(table(1024px), pablet(767px), mobile(480px))에 따라서 Masonry 레이아웃을 플렉스로 구성합니다.
2. 각 박스의 높이는 220px(큰 박스), 140px(작은 박스)가 적절히 섞이도록 구성해야 합니다.
3. flex를 사용하여 적절히 높이를 조절하되 횡방향으로 구성합니다.
4. 박스의 순서는 figma를 확인해주세요.
5. Vertical(세로형태)로 먼저 작업해보시고 Horizontal(가로형태)를 추가로 작성해보세요.

문제



주요 학습 키워드

flex로 다양한 레이아웃 만들기

flex로 갤러리 만들기

flex로 masonry 만들기

작성해주셔야 하는 question 파일경로

./question/question.css

./question/question.scss

실행 방법

경로 ./question/question.html

index.html 열기

Case21: Flex - 출제자 해설코드

```
@charset "utf-8";

/* variables */

$w100: 100%;
$half: 50%;
$large_box: 220px;
$small_box: 140px;
$basis_space: 40px;
$item_space: 10px;
$padding: 15px;
$box-shadow: 14px 17px 34px -20px rgba(0, 0, 0, 0.75);
$platform_tablet: 1024px;
$platform_pablet: 767px;
$platform_mobile: 480px;

/* reset */

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

*:focus {
    outline: none;
}

html,
body {
    position: relative;
    width: $w100;
    height: $w100;
    padding: 0;
    margin: 0;
}

h1 {
    text-align: center;
    margin: 35px auto 0;
}

/* mixin */

// 큰 박스 개수, 작은 박스 개수, 아이템 너비, 세로로 가장 긴 라인의 아이템 개수, 컬럼 숫자, 아이템의 높이
@mixin responsiveMixin($largeNum, $smallNum, $itemWidth, $itemNum, $colNum, $nthNum) {
    // 가장 긴 라인의 큰 박스 개수 + 작은 박스 + 기준 어백 위아래 2곳 + 각 아이템 별 하단 어백 * 아이템 개수
    height: calc(#{$large_box} * #{$largeNum} + #{$small_box} * #{$smallNum} + #{$basis_space} * 2 + #{$item_space} * #{$itemNum});
    .item {
        width: calc(#{$itemWidth} - #{$item_space});
        // 기본 박스 높이는 nthNum을 사용합니다.
        &:nth-of-type(#{$nthNum}n) {
            height: $small_box;
        }
        &:nth-of-type(#{$nthNum} + 1)n) {
            height: $large_box;
        }
    }
    // 수평 레이아웃을 위한 세팅입니다.
    &.horizontal {
        // 순서 세팅은 colNum을 사용합니다.
        @for $i from 1 through $colNum - 1 {
            &:nth-of-type(#{$colNum}n + #{$i}) {
                order: #{$i};
            }
        }
        &:nth-of-type(#{$colNum}n) {
    
```

```

        order: #{$colNum};
    }
}
}

.flex_wrap {
    /* 나름의 css 선택자의 순서 규칙을 가지고 작성하면 다른 사람들이 유지보수할 때 패턴을 파악하기 용이합니다.
     * 저는 다음과 같은 규칙을 가지고 작성합니다.
     * display 속성 -> 포지션에 대한 속성 -> 너비 높이 -> 여백 -> 백그라운드 컬러 등 기타 속성 -> 텍스트 속성
     */
    display: flex;
    flex-flow: column wrap;
    align-content: space-between;
    width: $w100;
    max-width: $platform_tablet;
    margin: $basis_space auto;
    padding: $basis_space;
    background-color: #fafafa;
    counter-reset: items;
    @include responsiveMixin(2, 3, 25%, 5, 4, 2);
}

.item {
    display: flex;
    position: relative;
    height: $small_box;
    margin-bottom: $item_space;
    padding: $padding;
    -webkit-box-shadow: $box-shadow;
    box-shadow: $box-shadow;
    color: #000;
    box-sizing: border-box;
    overflow: hidden;
    background-color: rgba(0, 0, 0, .3);
    img {
        position: absolute;
        left: $half;
        top: $half;
        -webkit-transform: translate3d(-$half, -$half, 0);
        transform: translate3d(-$half, -$half, 0);
    }
    &:before {
        counter-increment: items;
        content: counter(items);
        position: relative;
        z-index: 1;
        color: #fff;
    }
}
@media (max-width: $platform_tablet) {
    @include responsiveMixin(2, 5, 33.33%, 7, 3, 3);
}
@media (max-width: $platform_pablet) {
    @include responsiveMixin(3, 7, $half, 10, 2, 2);
}
@media (max-width: $platform_mobile) {
    @include responsiveMixin(0, 0, $w100, 0, 1, 1);
    height: auto;
    // mobile class는 css 선택자의 우선순위를 높이기 위함입니다. (important 사용 지양을 위함)
    .item.mobile {
        width: $w100;
        height: $large_box;
    }
}
}
}

```