BRO Academy

CSS+

Lesson 6

Lesson Plan

1
Pseudo-classes

2
Especially User actions

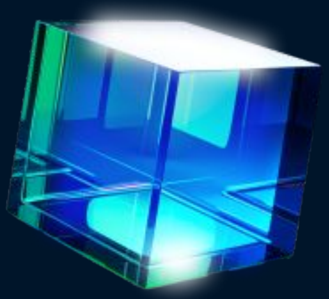
3
Pseudo-elements

4
Box model

Pseudo-classes

Help style specific attributes or states that are not reflected in the DOM.

- `User-action pseudo-classes`
- `The lang pseudo-class`
- `The negation pseudo-class`
- `Structural pseudo-classes`
- `User interface pseudo-class selectors`



Link Pseudo-classes for `<a>`

`:link` – styles unvisited links

`:visited` – styles visited links

```
a:link { color: blue; }

a:visited { color: purple; }
```

User-action pseudo-classes

`:active` – element is being clicked

`:focus` – element is in focus

`:hover` – mouse is hovering over the element

`a, button, input`

The `lang` pseudo-class

Applied to elements with the `lang` attribute

```
p:lang(fr) { font-style: italic; }


<p lang="fr">
   Adieu
</p>
<p lang="jw">
   Sugeng rawuh
</p>
```

The **negation** pseudo-class

Styles are applied to all elements except those matching the selector

`:not(p) { }` – all elements except paragraphs **tags**

`:not(.intro) { }` – all elements except those **with class** .intro

`:not(#news) { }` – all elements except those **with id** #news

`:not(:lang(fr)) { }` – all elements **except** those with the French **language**

`:not([disabled]) { }` – all elements except those **without** the disabled **attribute**

`p:not(.intro) { }` – all paragraphs **except** those with the class `.intro`

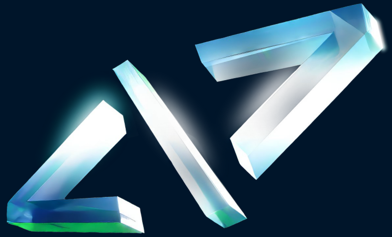
Structural pseudo-classes

Allow you to select elements based on their position in the document structure.

! If the document structure changes, the structural pseudo-class might apply to a different element or potentially to no element at all.

It can sometimes be difficult to determine exactly which element the styles will be applied to.

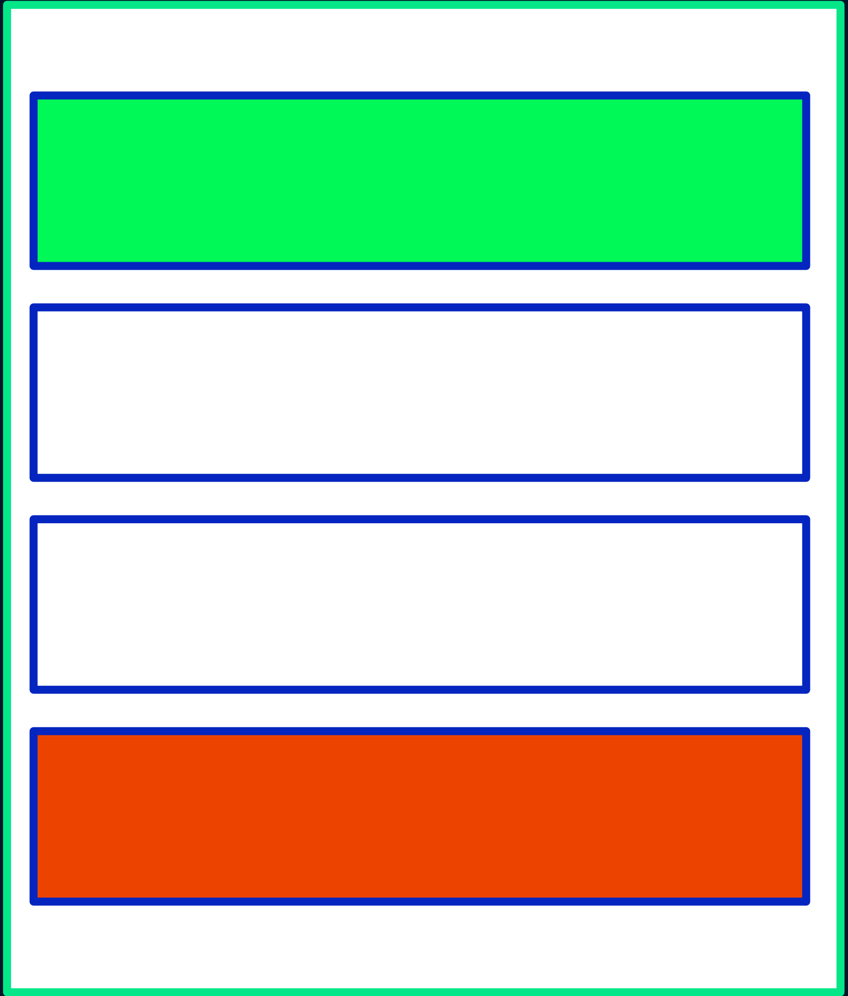
```
:first-child { }

:only-child { }

:nth-child(3n) { }
```

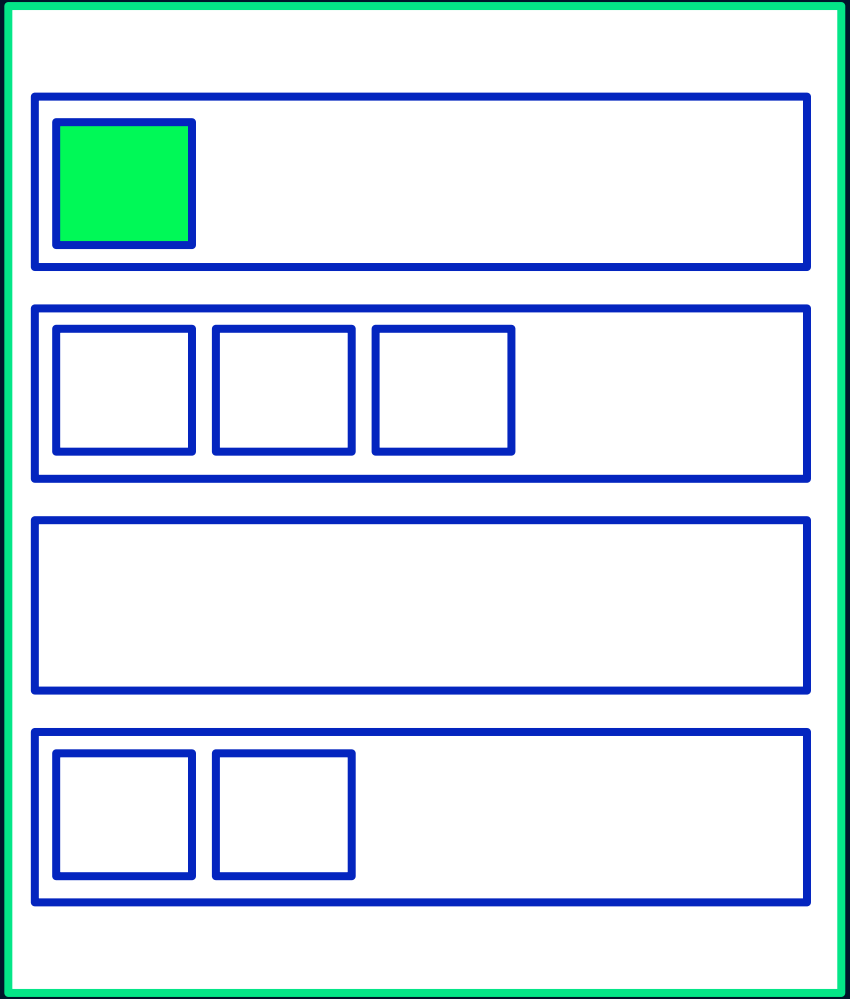


:first-child
:last-child

– Selects the element that is the first/last child of another element.

```
article:first-child { }
article:last-child { }

<section>
   <article> 1 </article>
   <article> 2 </article>
   <article> 3 </article>
   <article> 4 </article>
</section>
```



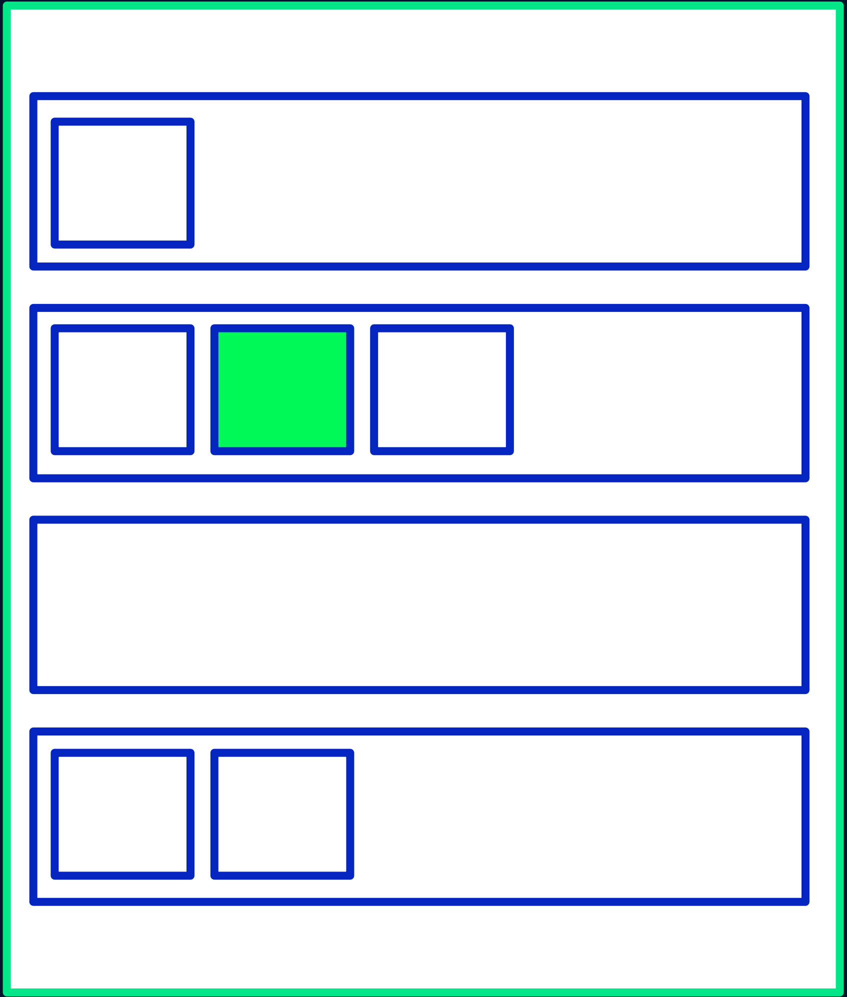


pseudo-class

:only-child

– Selects an element that is the only child of another element.

```
div:only-child { }

<article>
   <div> 1 </div>
</article>
```



pseudo-class

:only-of-type

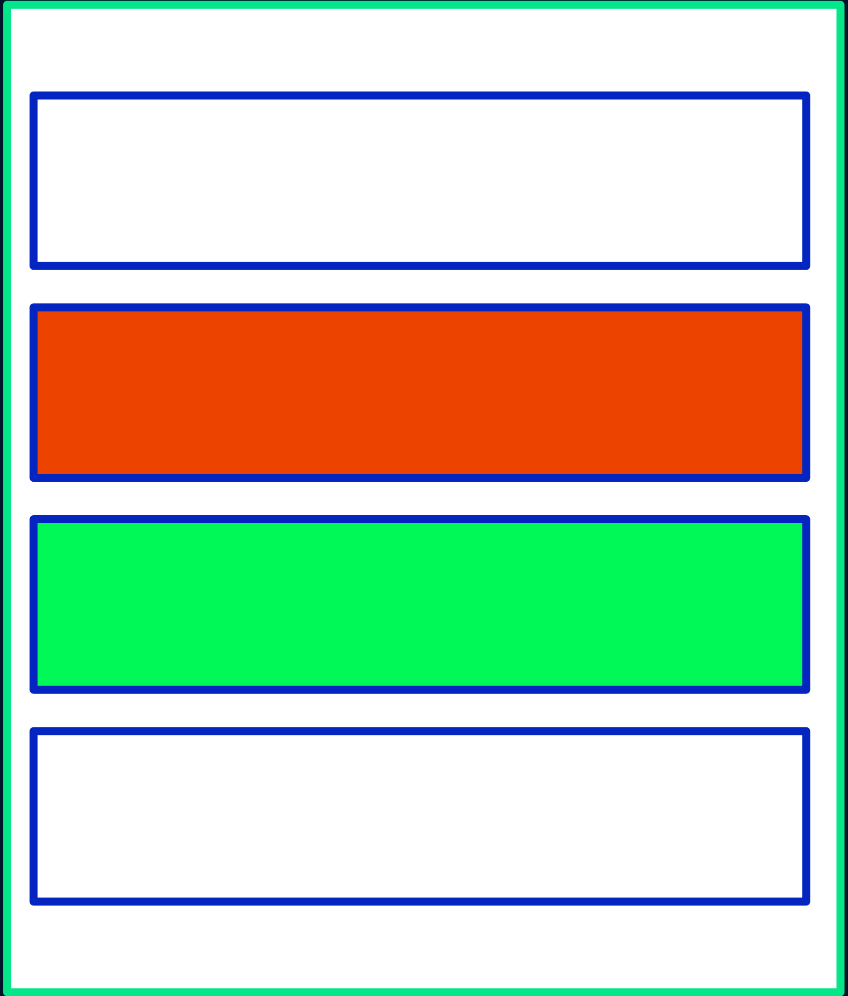
– Selects an element that is the **only** element of its **type** within its parent.

```
p:only-of-type { }

<article>
   <div> 1 </div>
   <p> 1 </p>
   <div> 1 </div>
</article>
```

:first-of-type
:last-of-type

– Selects an element that is the first/last of its type within its parent element.

```
p:first-of-type { }
p:last-of-type { }

<section>
   <article> 1 </article>
   <p> 2 </p>
   <p> 3 </p>
   <article> 4 </article>
</section>
```

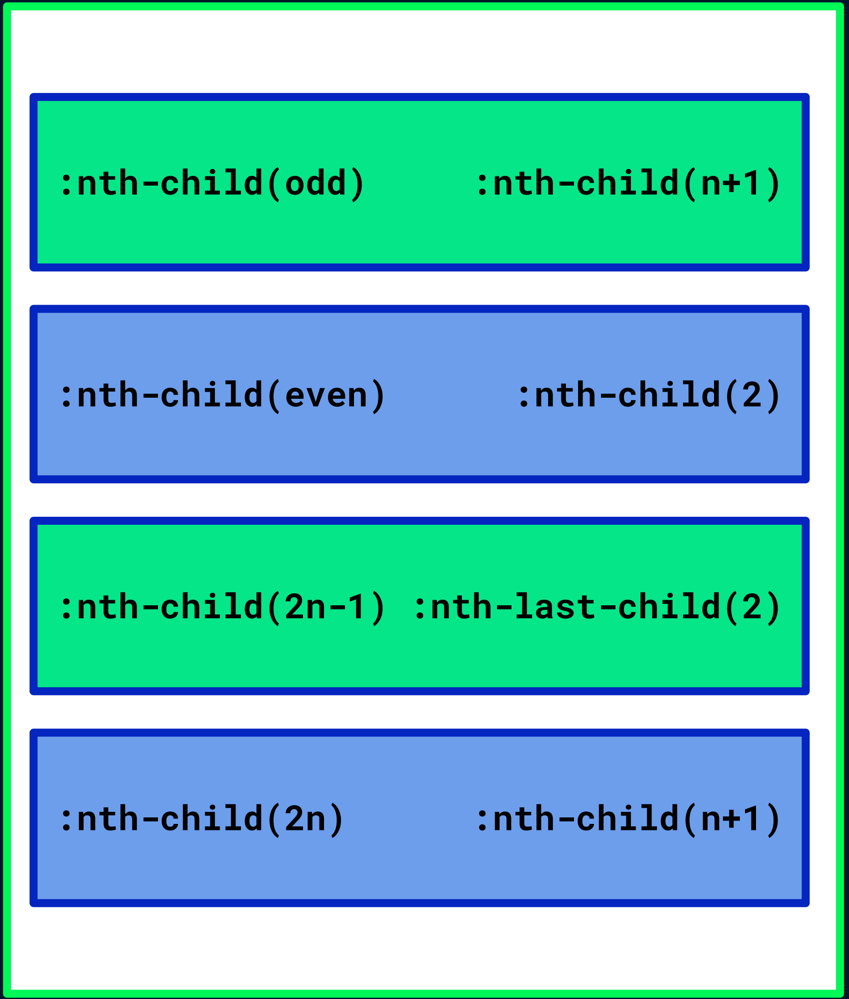


:nth-child(n)
:nth-last-child(n)

– Selects specific child elements in a parent element starting from the beginning or the end.

n:

- number
- number + n (selects every n-th element)
- expression with +/- (allows starting from an element other than the first)
- even (all even elements)
- odd (all odd elements)

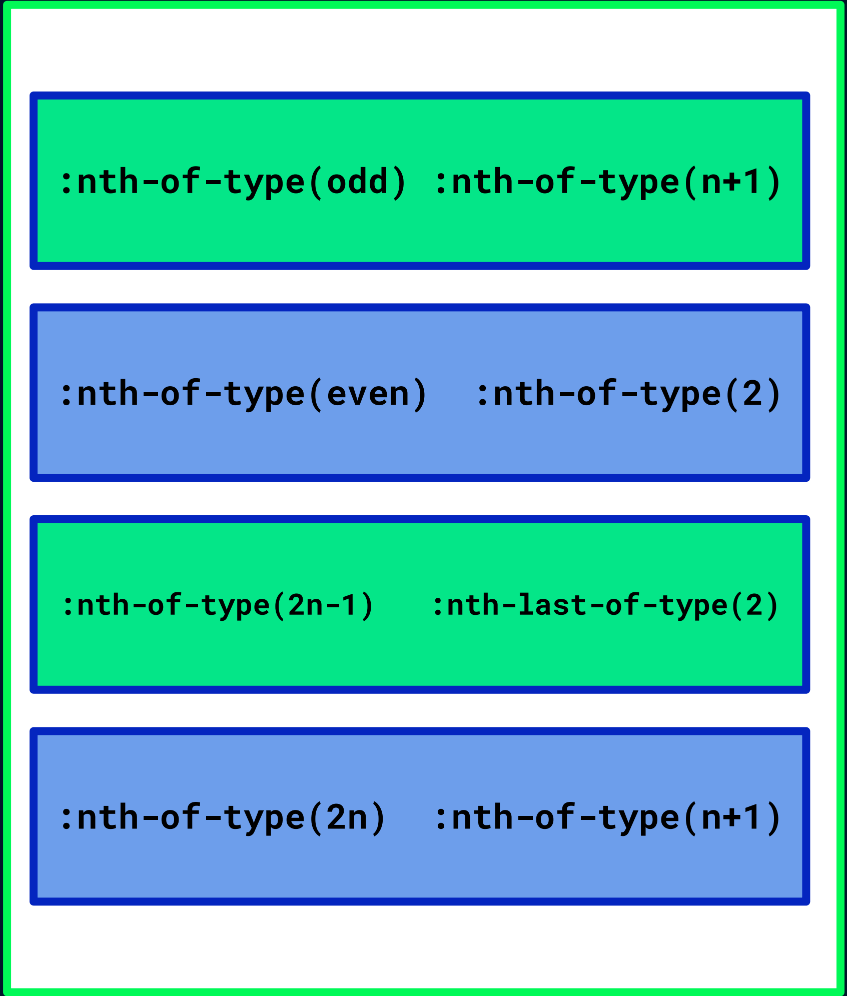


:nth-of-type(n)
:nth-last-of-type(n)

– Selects elements of a specific type in the parent element starting from the beginning or the end.

n:

- number
- number + n (selects every n-th element)
- expression with +/- (allows starting from an element other than the first)
- even (all even elements)
- odd (all odd elements)



:nth-of-type(odd) :nth-of-type(n+1)
:nth-of-type(even) :nth-of-type(2)
:nth-of-type(2n-1) :nth-last-of-type(2)
:nth-of-type(2n) :nth-of-type(n+1)

:root

– Selects the root element of the document (tag `<html>`).

:empty

– Selects an element that has no content or child elements (an empty element).

A space is already a character, so the tag is no longer considered empty.
It also applies to input elements where no value has been entered.

```
:root { }

<html>
   <head> 1 </head>
   <body> 1 </body>
</html>


p:empty { }

<article>
   <p> 1 </p>
   <p> </p>
   <p></p>
   <p><span></span></p>
</article>
```

PSEUDO-ELEMENTS

Pseudo-elements

– **(fake elements)** Allow styling elements that are not in the document tree.

`::-webkit-scrollbar` – styles the scrollbar

+ Other pseudo-elements of the form `::-webkit-scrollbar-*,` are used only with prefixes and only in `webkit` browsers

```
.invisible-scrollbar::-webkit-scrollbar {
  display: none;
}
```

Pseudo-elements for text

`::first-line` – styles the first line of text

`::first-letter` – styles the first letter of text

```
p::first-line { }
p:first-letter { }
<p>
   This is the first line
   of a paragraph of text
</p>
```

Pseudo-elements that Create a New Element

`::before` – A new element is created at the beginning of the element, before the content

`::after` – A new element is created at the end of the element, after the content

- An element can have only one `::before` and one `::after`
- These pseudo-elements are inserted into the document flow and occupy space
- They are visible in the inspector but cannot be accessed via JavaScript
- Any CSS properties can be applied to them.
- They can only be applied to elements with closing tags
- Often used to add decorative elements

content

– replaces content with a generated value.
This is a **required** property for the **before** and **after** pseudo-elements

```
p::before {
  content: “”;
}
```

Text	`“hello”;`
Image	`url(pic.png);`
Attribute (displays the value of an attribute as text)	`attr(cite);`
Counter	`counter(list-order);`
Nothing	`"  ";`
Special characters	`"\21E6";`
Emoji	`"🎮";`
Gradient	`linear-gradient(#e66465, #9198e5);`

content

In the **content** property, you can combine different values:

```
p:before {
  content: "class: " attr(class);
}
```

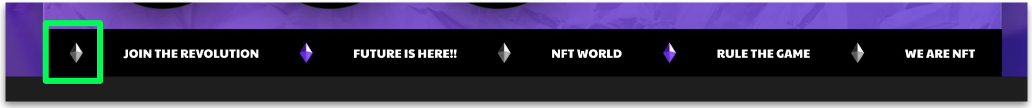
The **text "class: "** will be placed before the paragraph content, followed by the list of **all class attribute** values for that paragraph.

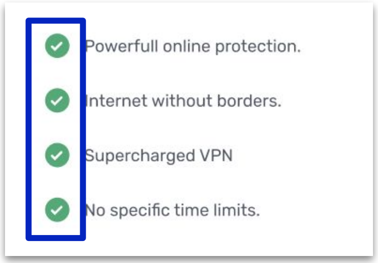
Cannot contain HTML tags:

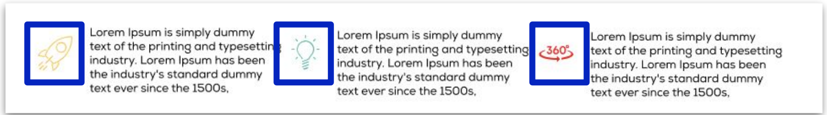
```
p:before {
  ~~content: <p>test<p>;~~
}
```

Examples of Pseudo-elements

They replace an empty tag that you might want to add for styling purposes.



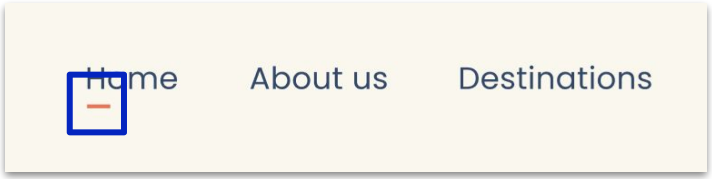


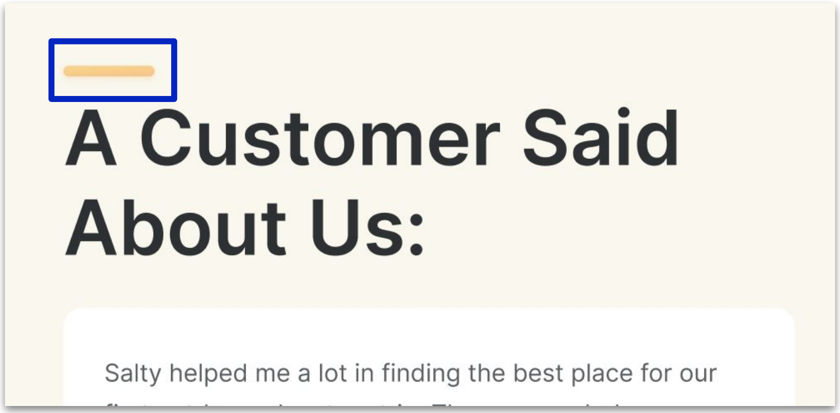


Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500s,

Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500s,

Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500s,





A Customer Said About Us:

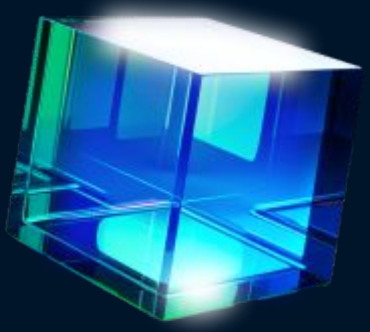
Salty helped me a lot in finding the best place for our

Pseudo-elements for Lists

- Usage of counters in lists
- Styling list markers

`::marker` – Styling list markers.

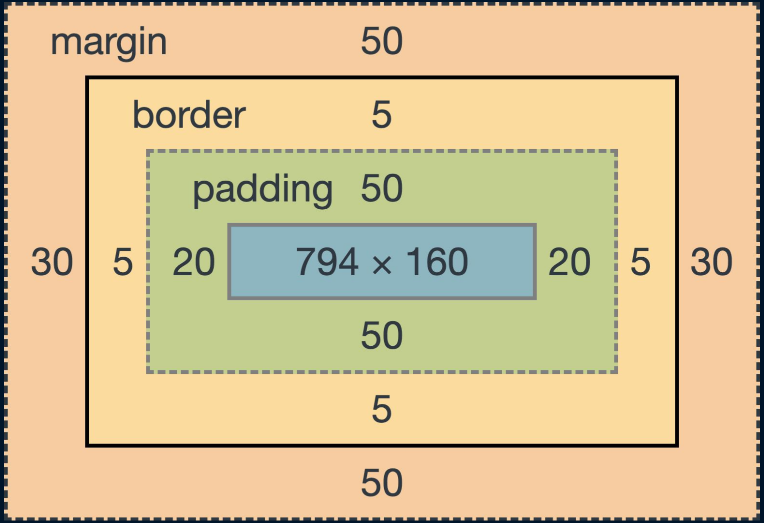
```
ol {
  counter-reset: section;
}
li::before {
  counter-increment: section;
  content: counter(section);
}
li::marker { }
```



BOX MODEL

Box model

- content
- padding – The space inside the element
- border
- margin – The space outside the element, pushing neighboring elements away.

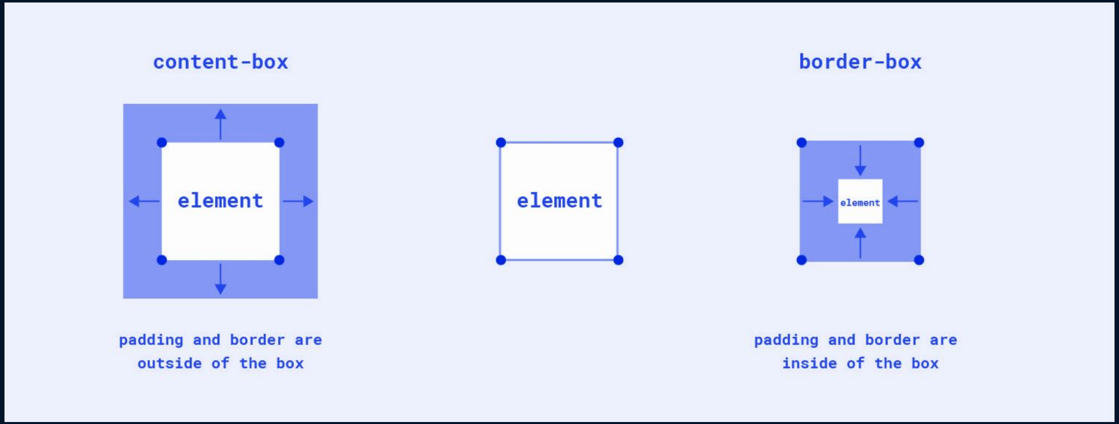


Demo

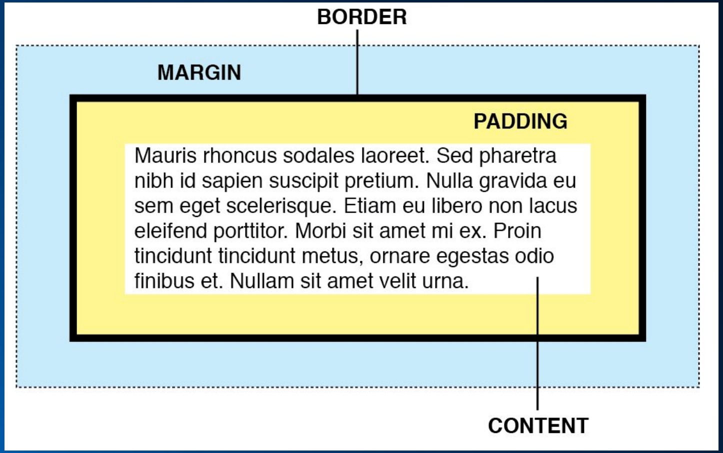
box-sizing property

Default:
`content-box` – width and height properties include the `content`, but **does not include** the `padding`, `border`, or `margin`.

`border-box` – width and height properties **include** the `content`, `padding`, and `border`, but **do not include** the `margin`.

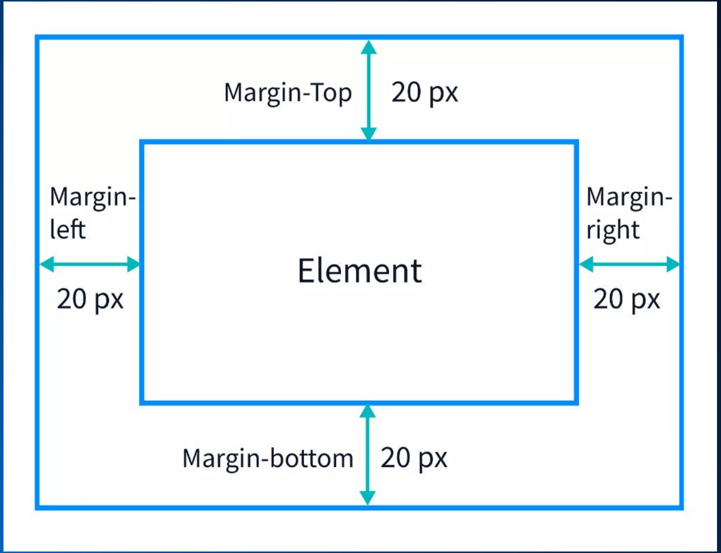

content-box
element
padding and border are outside of the box
element
border-box
element
padding and border are inside of the box

padding


BORDER
MARGIN
PADDING
Mauris rhoncus sodales laoreet. Sed pharetra nibh id sapien suscipit pretium. Nulla gravida eu sem eget scelerisque. Etiam eu libero non lacus eleifend porttitor. Morbi sit amet mi ex. Proin tincidunt tincidunt metus, ornare egestas odio finibus et. Nullam sit amet velit urna.
CONTENT

- is the internal space from the content to the edges of the element.
- Units of measurement:
  - **px**
  - **%**
- Cannot be negative.

margin

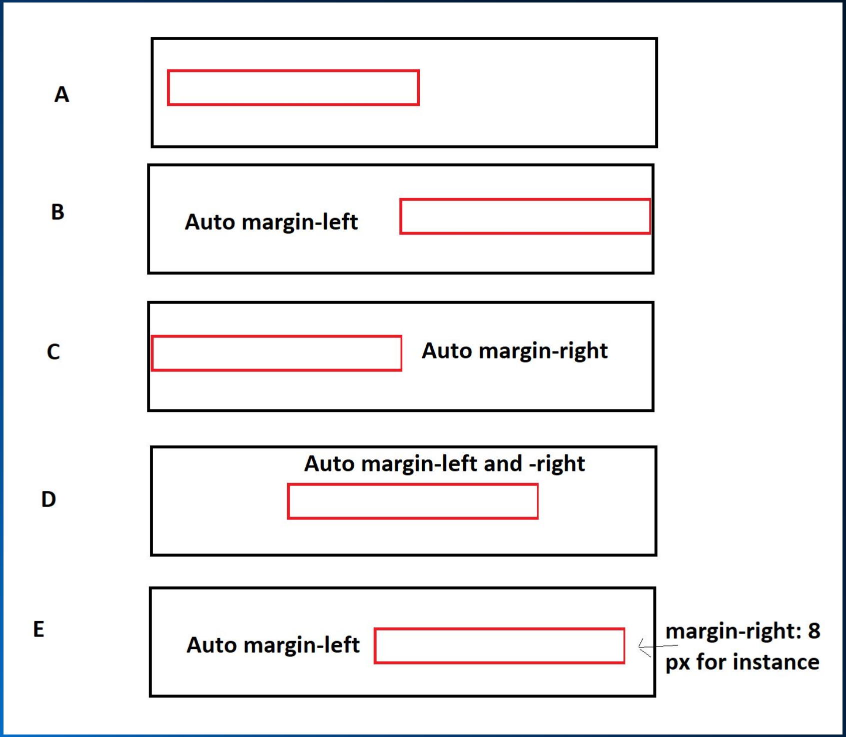


- The outer space that pushes neighboring elements away.
- Units of measurement:
  - **px**
  - **%**
  - **auto**

- Can have negative values:

  `margin-top: -10px;`
- `margin-top` and `margin-bottom` do not work in `inline-box`

margin: auto



```
margin: auto;
margin-right: auto:
margin: 0 auto;
```

The width of the margin is **automatically calculated** by the browser and occupies **all available space** between elements and margins within the container.

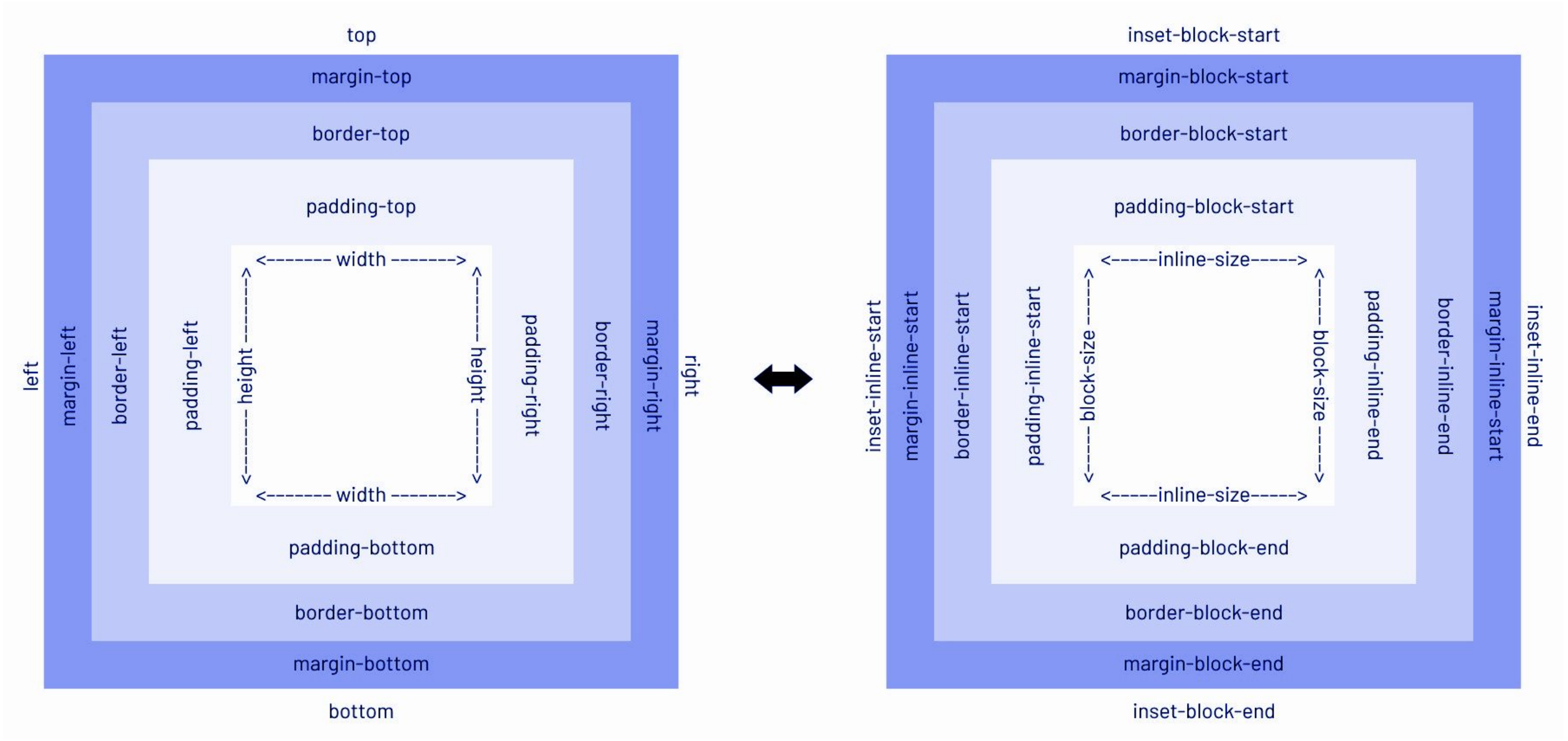
CSS Logical Properties

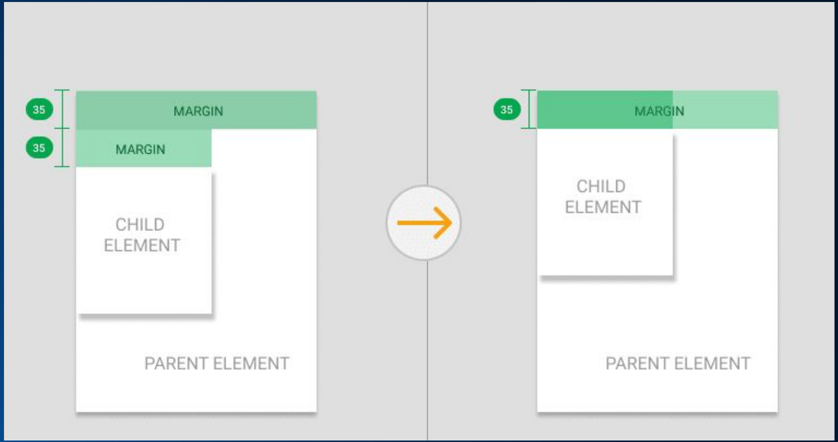
Margins/padding could be applied consistently **according to the document flow**:

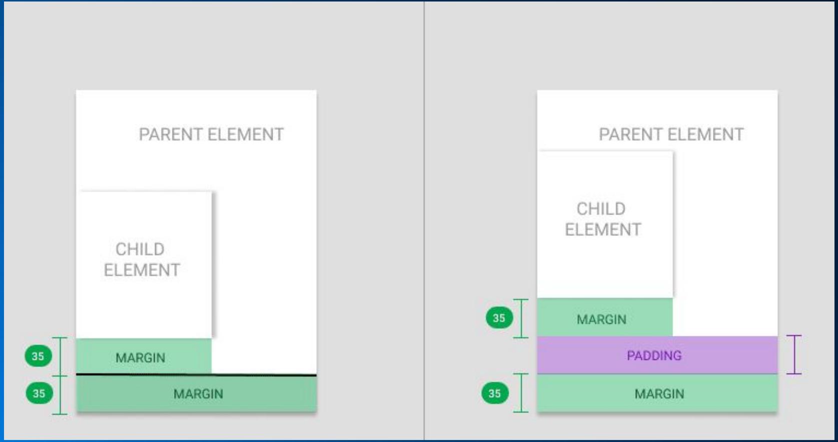
- use **-block** for **vertical spacing**
- use **-inline** as for **horizontal spacing**

```
margin-block (margin-top + margin-bottom)
margin-inline (margin-left + margin-right)

padding-block (padding-top + padding-bottom)
padding-inline (padding-left + padding-right)
```

physical / logical

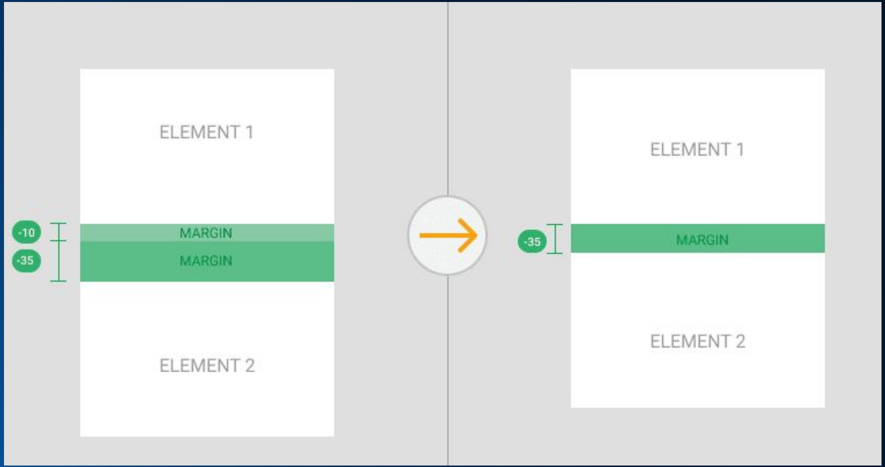


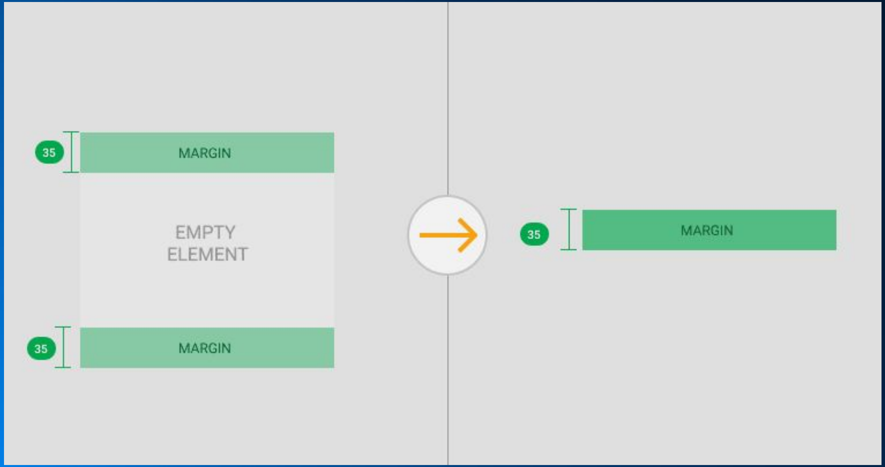




Collapsing Margins

- Works for the **`margin-top`** of the first child inside a parent and for the **`margin-bottom`** of the last child inside a parent.
- The child's **`margin`** becomes the parent's **`margin`** or the larger of the two applies.
- Does not work if the parent has **`padding`**
- In elements with **`position: absolute`** `and` **`position: fixed, display: flex, display: grid; float`** margins do not collapse or cause margin drop.



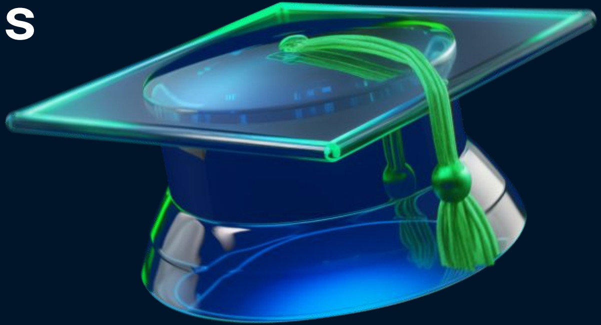


Collapsing Margins

- Works only vertically.
- Instead of adding margins together, the larger margin is applied.
- In elements with `position: absolute` and `position: fixed, display: flex, display: grid; float,` margins do not collapse or cause margin drop.
- If an element has no content inside, the margins still remain in the document flow.

Summary

1. Pseudo-elements types
   - before & after
2. Pseudo-classes
   - `User-action pseudo-classes`
3. Box model
   - padding
   - margin



Homework

1. Apply box model

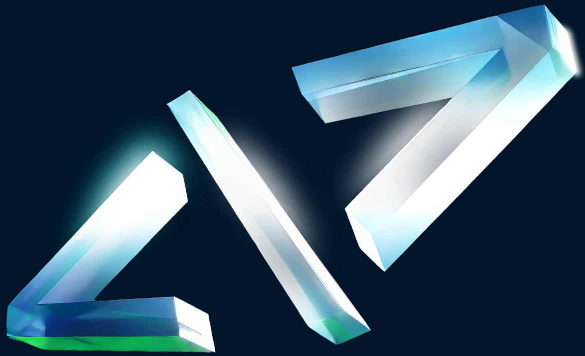
Implement inner (padding) and outer (margin) spacing for the elements.

2. Apply all states for links and buttons:

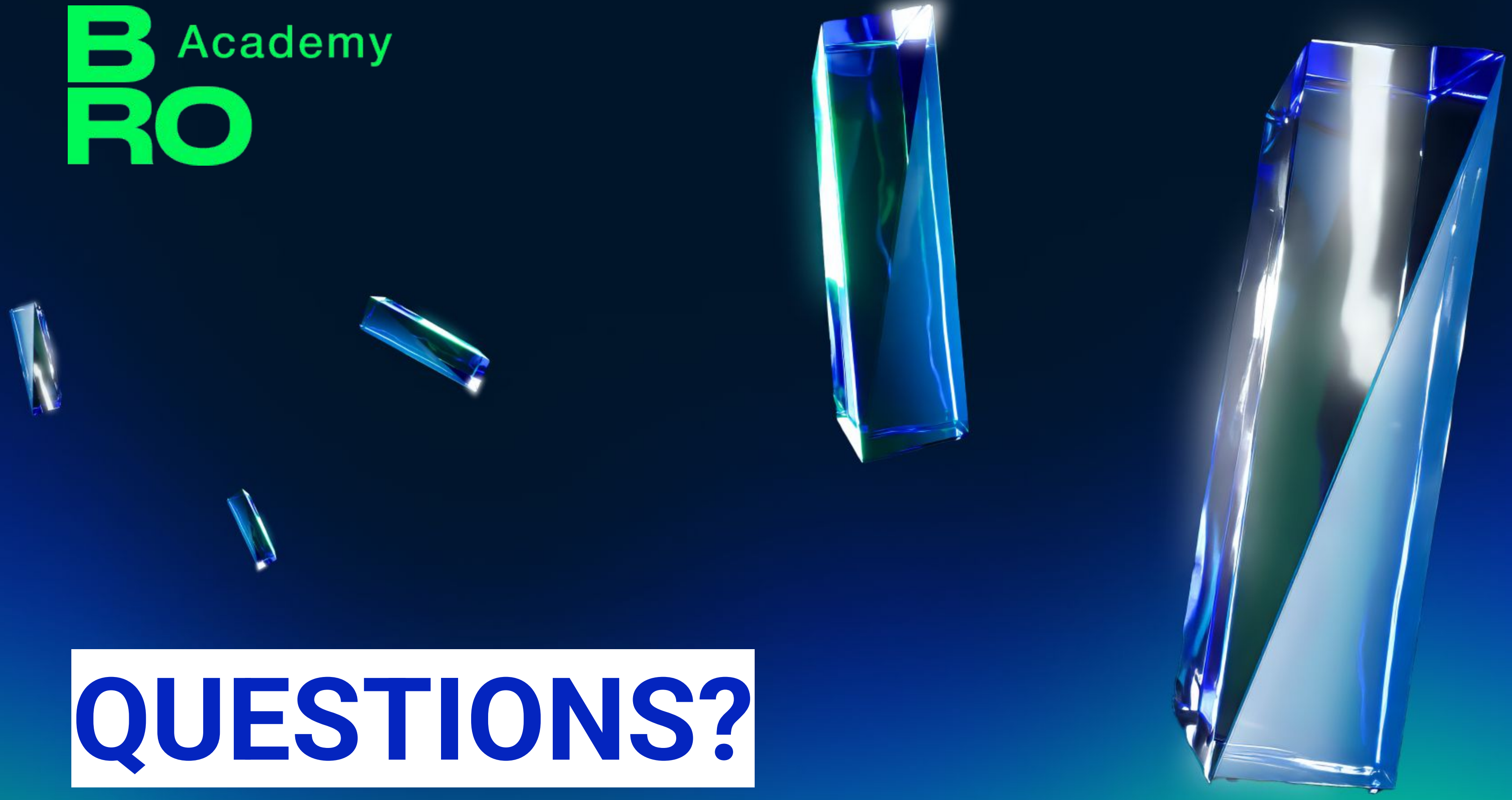
- hover
- active
- focus

according to the UI kit.

Use pseudo-elements where necessary



BRO Academy



QUESTIONS?

Please fill out the feedback form

It’s very important for us

THANK YOU!

Have a good evening!