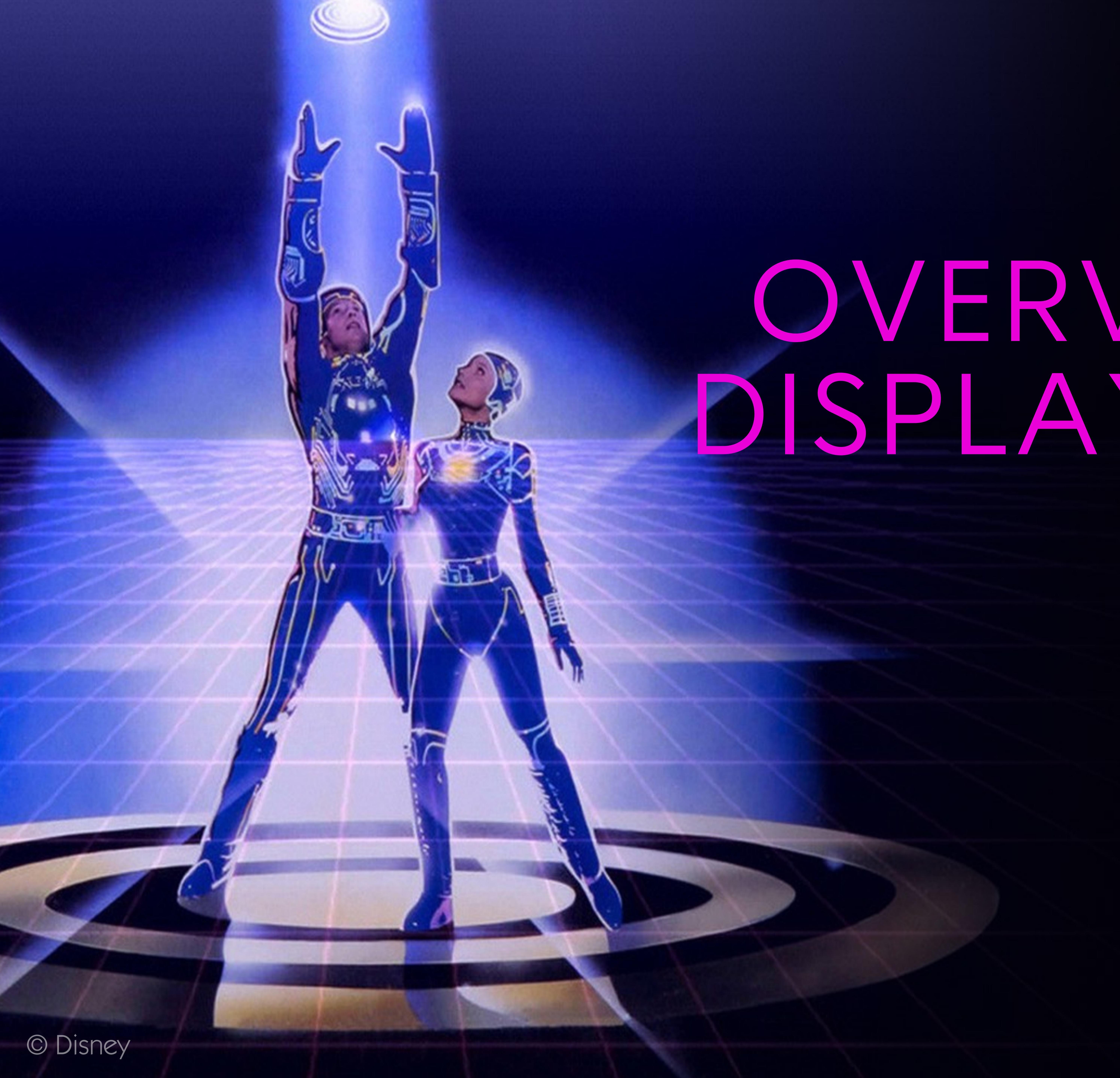


OVERVIEW OF THE DISPLAY PROPERTY & CSS GRID

10 September 2020

Brenda Storer

ME

@brendamarienyc

Independant Web Developer & Designer

I 💜 CSS

THE MOST COMMON DISPLAY PROPERTIES

```css
display: block;
display: inline;
display: inline-block;
display: none;
display: flex;
display: grid;
```

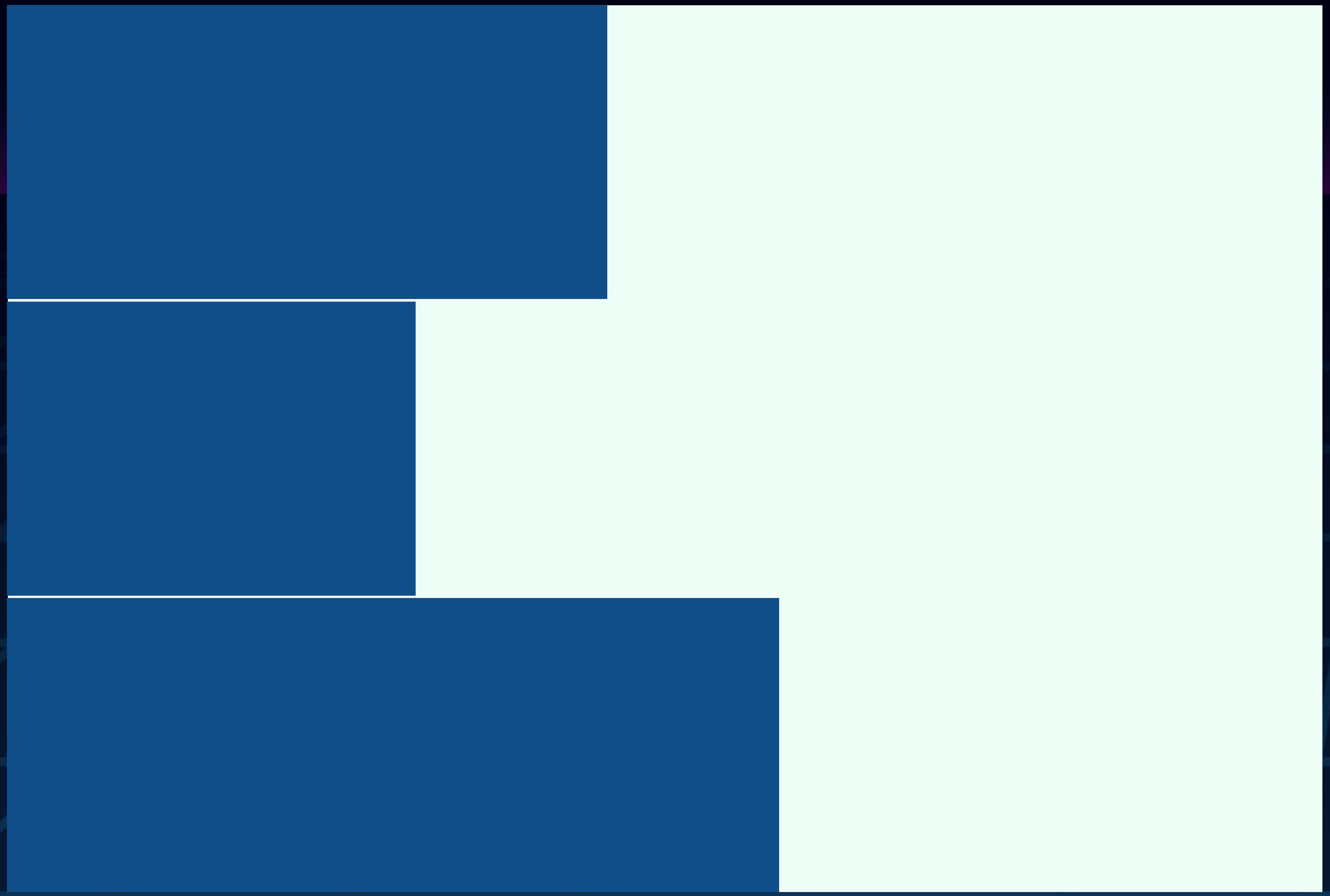
CHEAT SHEET

Understanding these will save you HOURS of bug squashing.

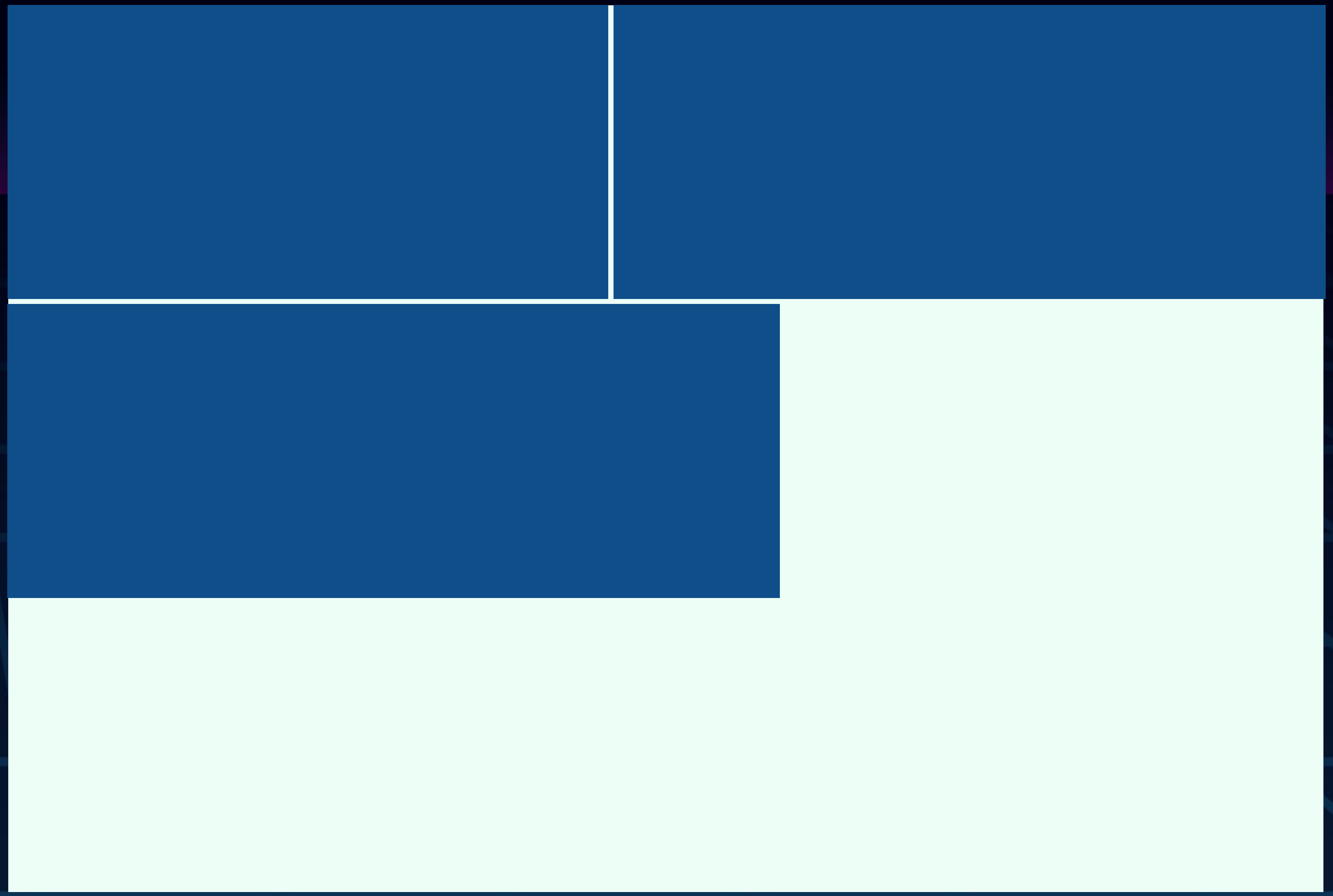
IN THE BEGINNING…

BLOCK vs INLINE

Layout vertically, one below the other



Display one after another in the direction that sentences run.



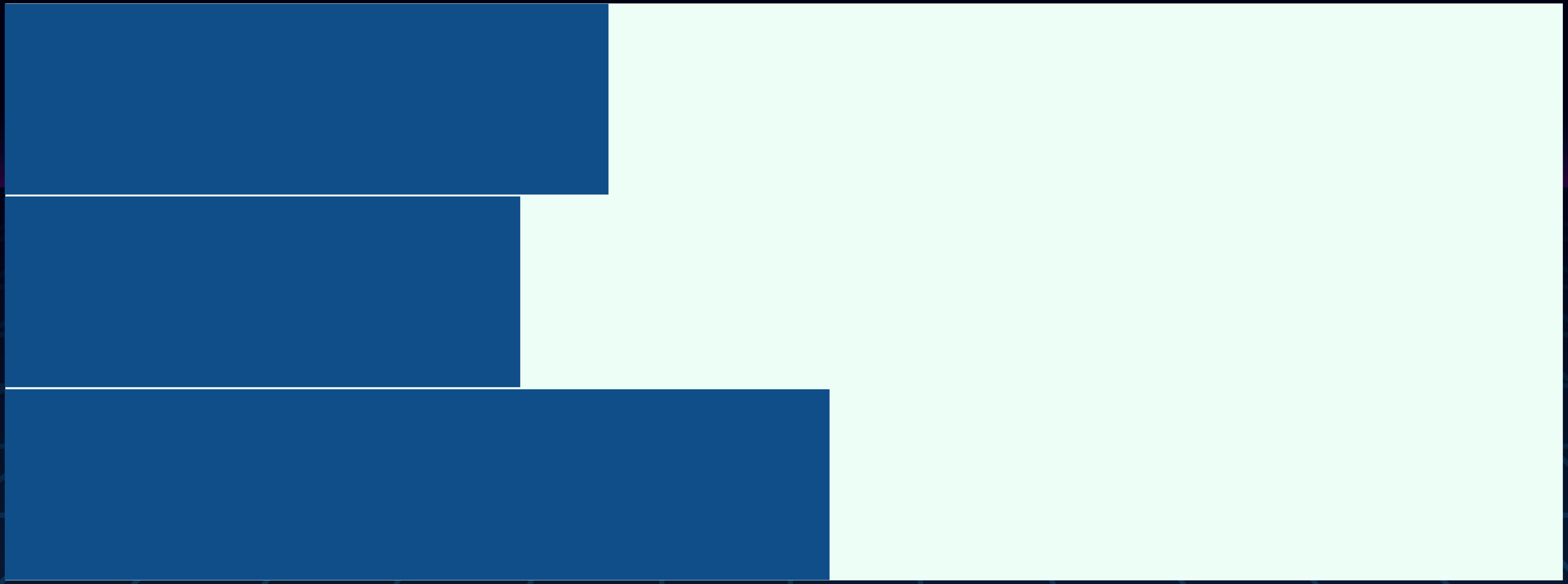
DISPLAY: BLOCK;

A few HTML elements with block as a default

```
<div>
<p>
<form>
<header>
<footer>
<h1><h2><h3><h4><h5><h6>
<main>
<section>
<ul>
<li>
```

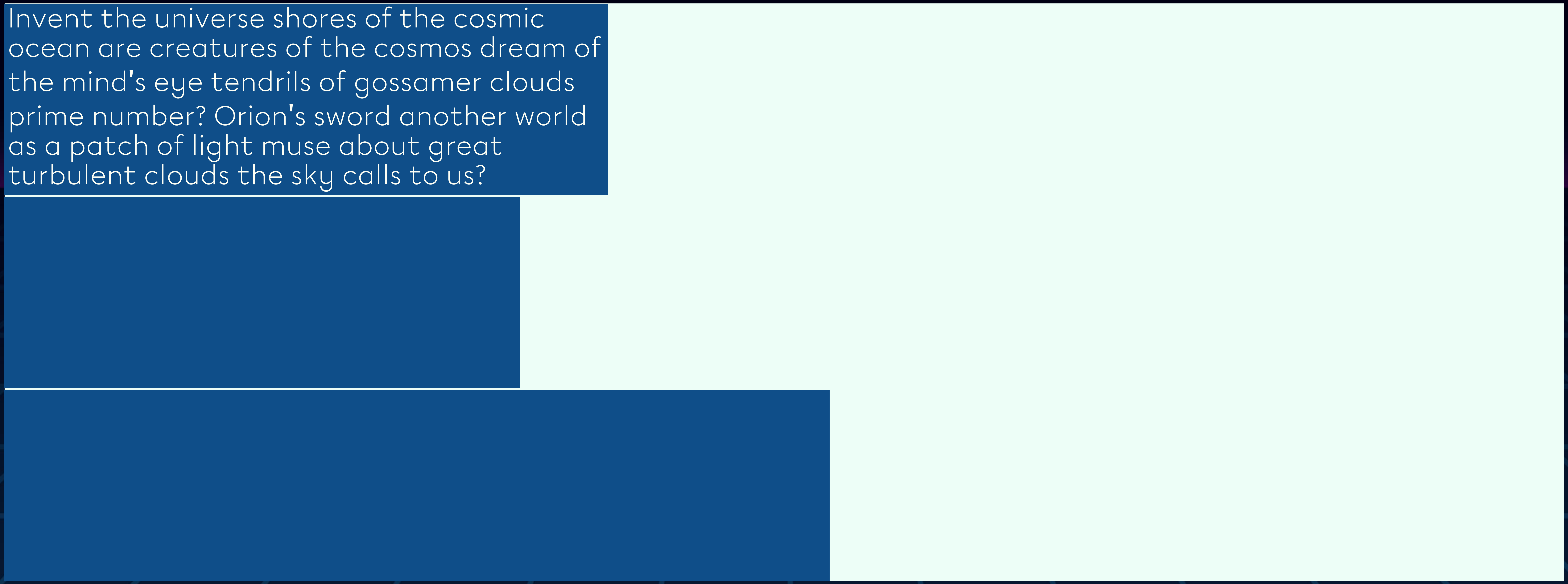
DISPLAY: BLOCK;

Layout vertically, one below the other



DISPLAY: BLOCK;

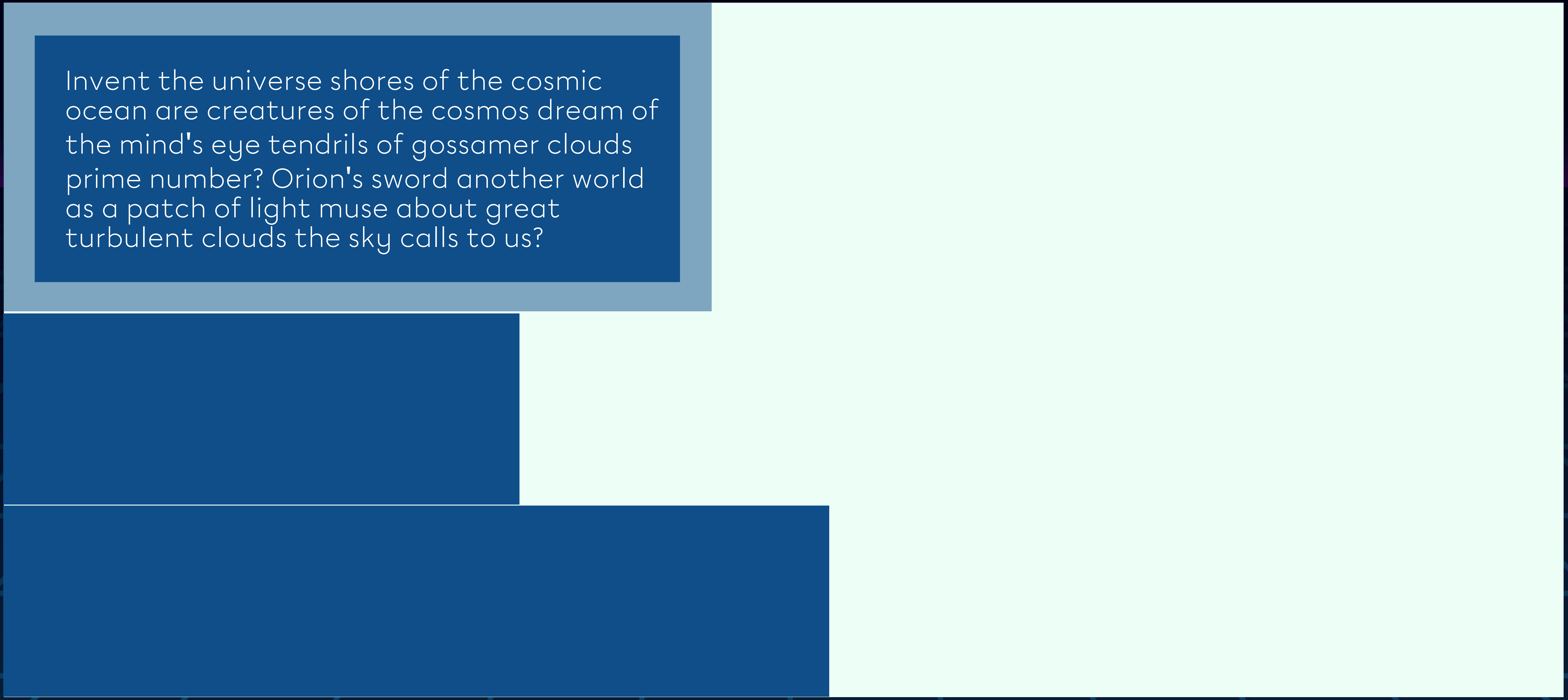
Often used as containers



Invent the universe shores of the cosmic ocean are creatures of the cosmos dream of the mind's eye tendrils of gossamer clouds prime number? Orion's sword another world as a patch of light muse about great turbulent clouds the sky calls to us?

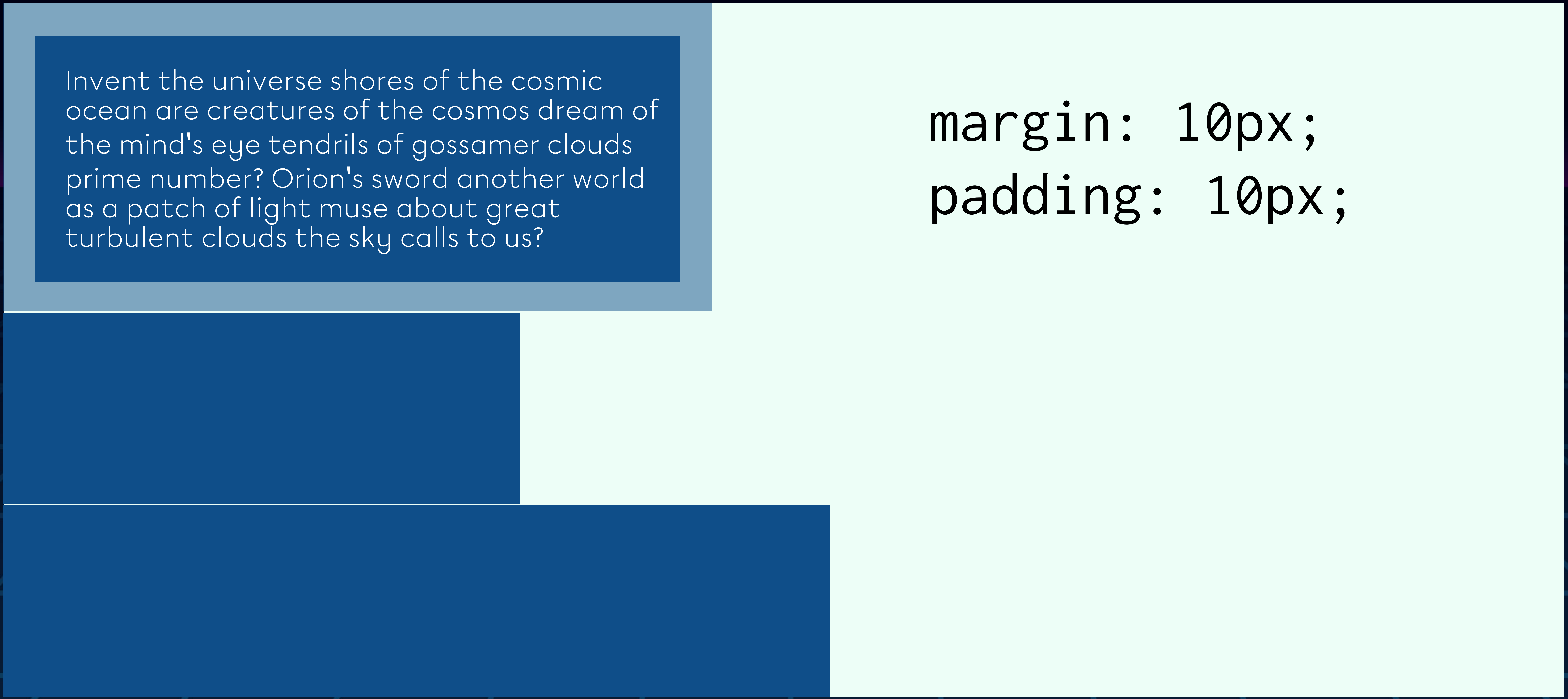
DISPLAY: BLOCK;

Takes margin & padding on all sides


Invent the universe shores of the cosmic ocean are creatures of the cosmos dream of the mind's eye tendrils of gossamer clouds prime number? Orion's sword another world as a patch of light muse about great turbulent clouds the sky calls to us?

DISPLAY: BLOCK;

Takes margin & padding on all sides

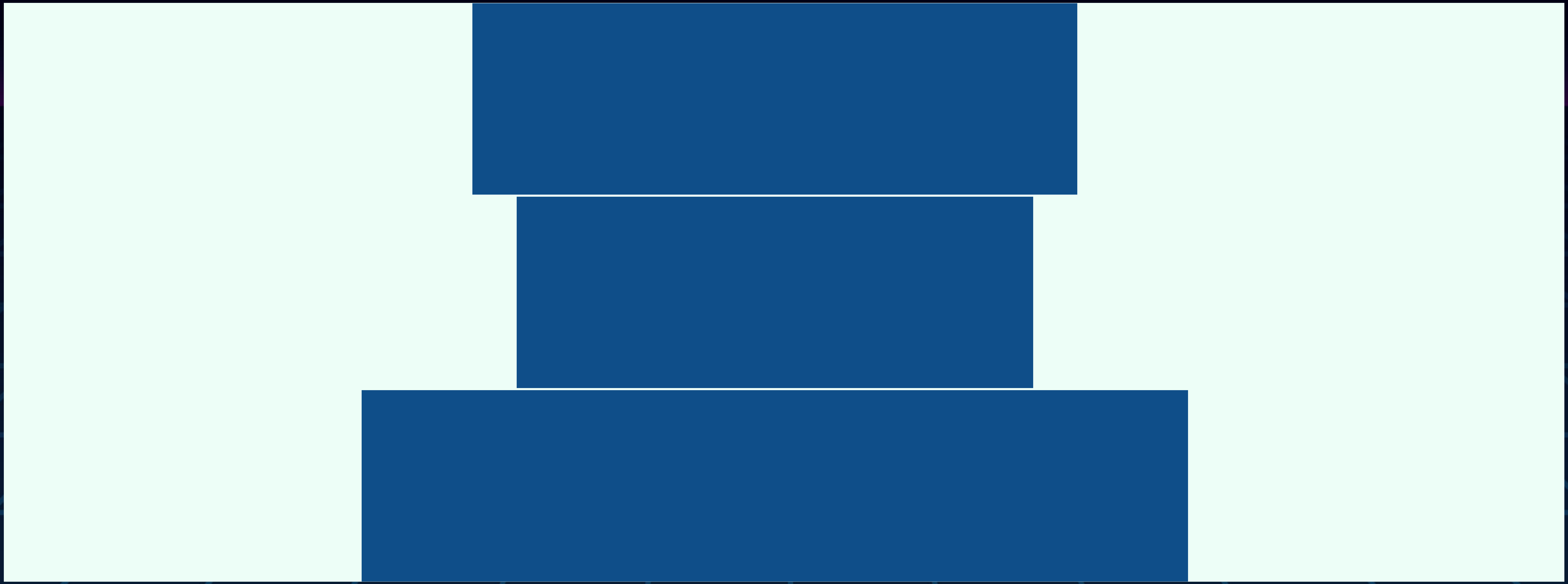

Invent the universe shores of the cosmic ocean are creatures of the cosmos dream of the mind's eye tendrils of gossamer clouds prime number? Orion's sword another world as a patch of light muse about great turbulent clouds the sky calls to us?

```
margin: 10px;
padding: 10px;
```

DISPLAY: BLOCK;

Center horizontally using auto margins

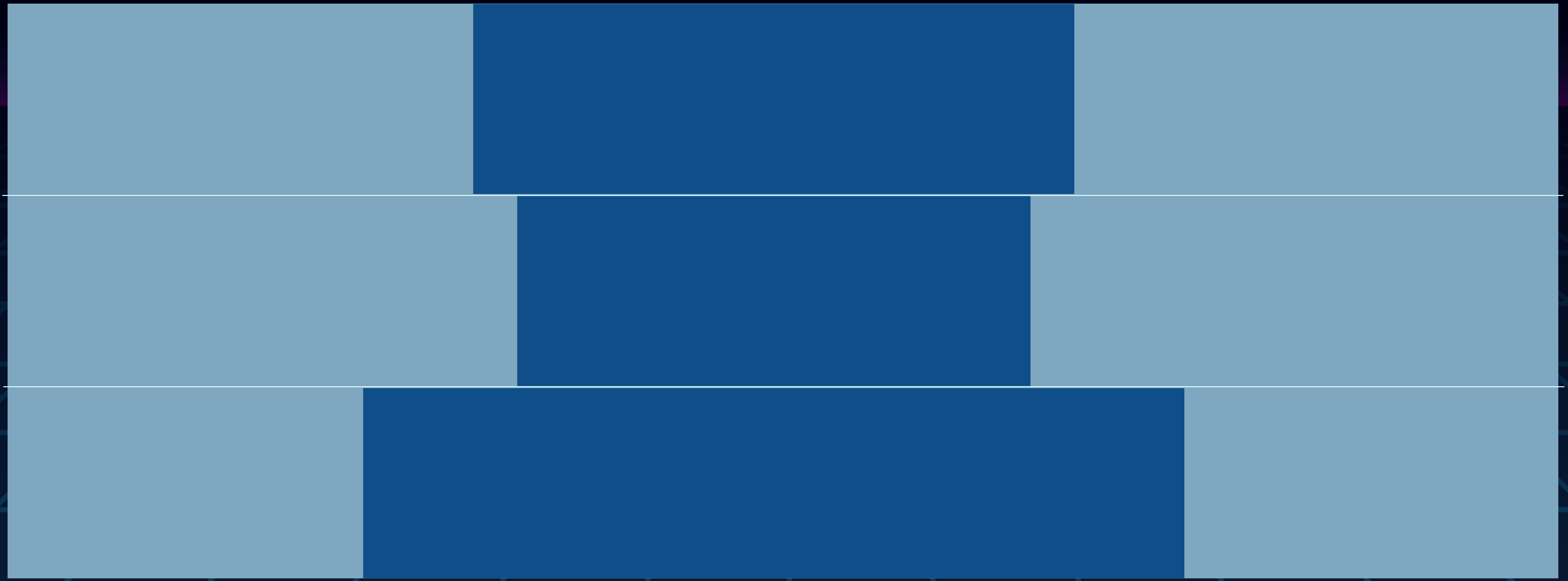
```
margin: 0 auto;
```



DISPLAY: BLOCK;

Center horizontally using auto margins

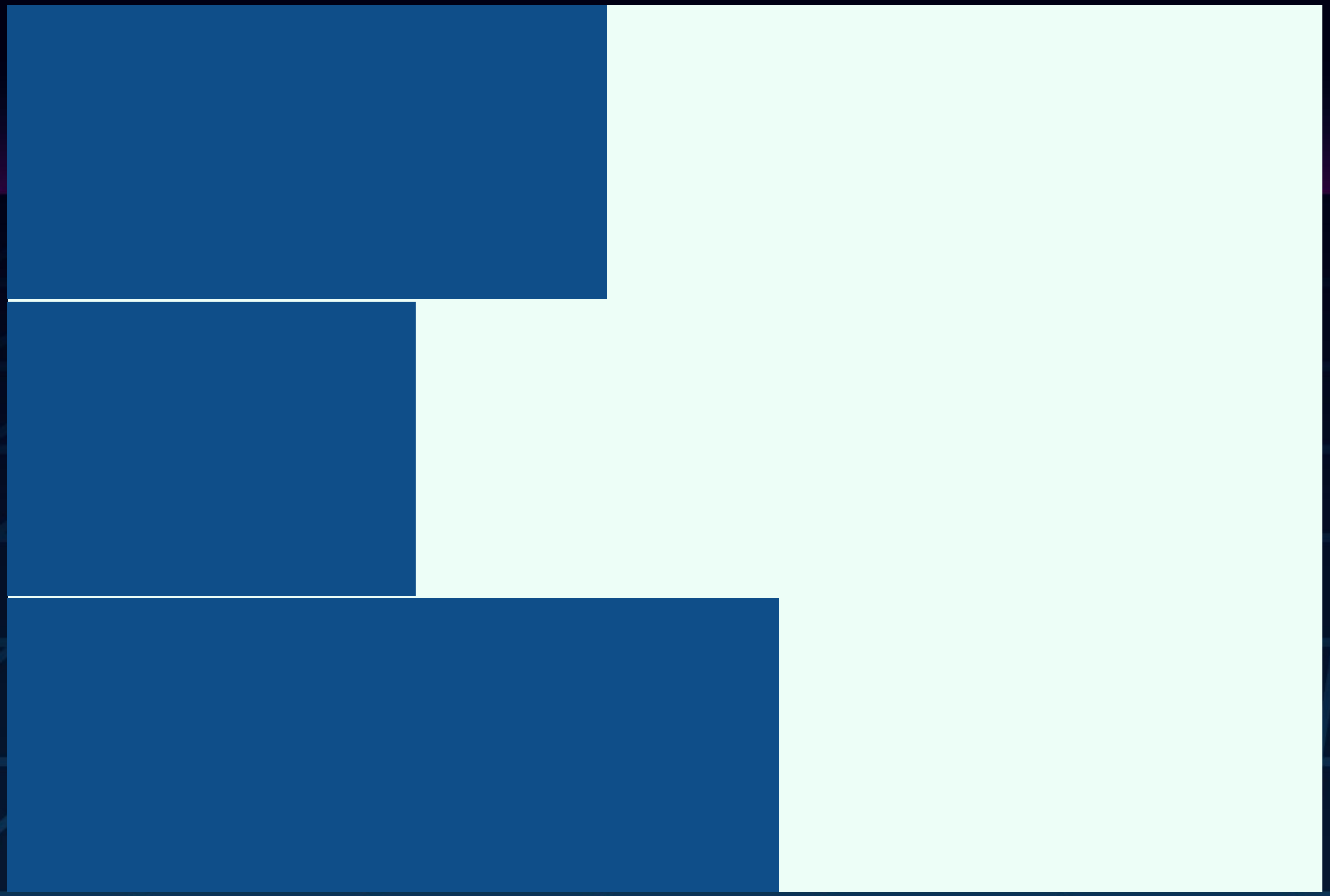
```
margin: 0 auto;
```

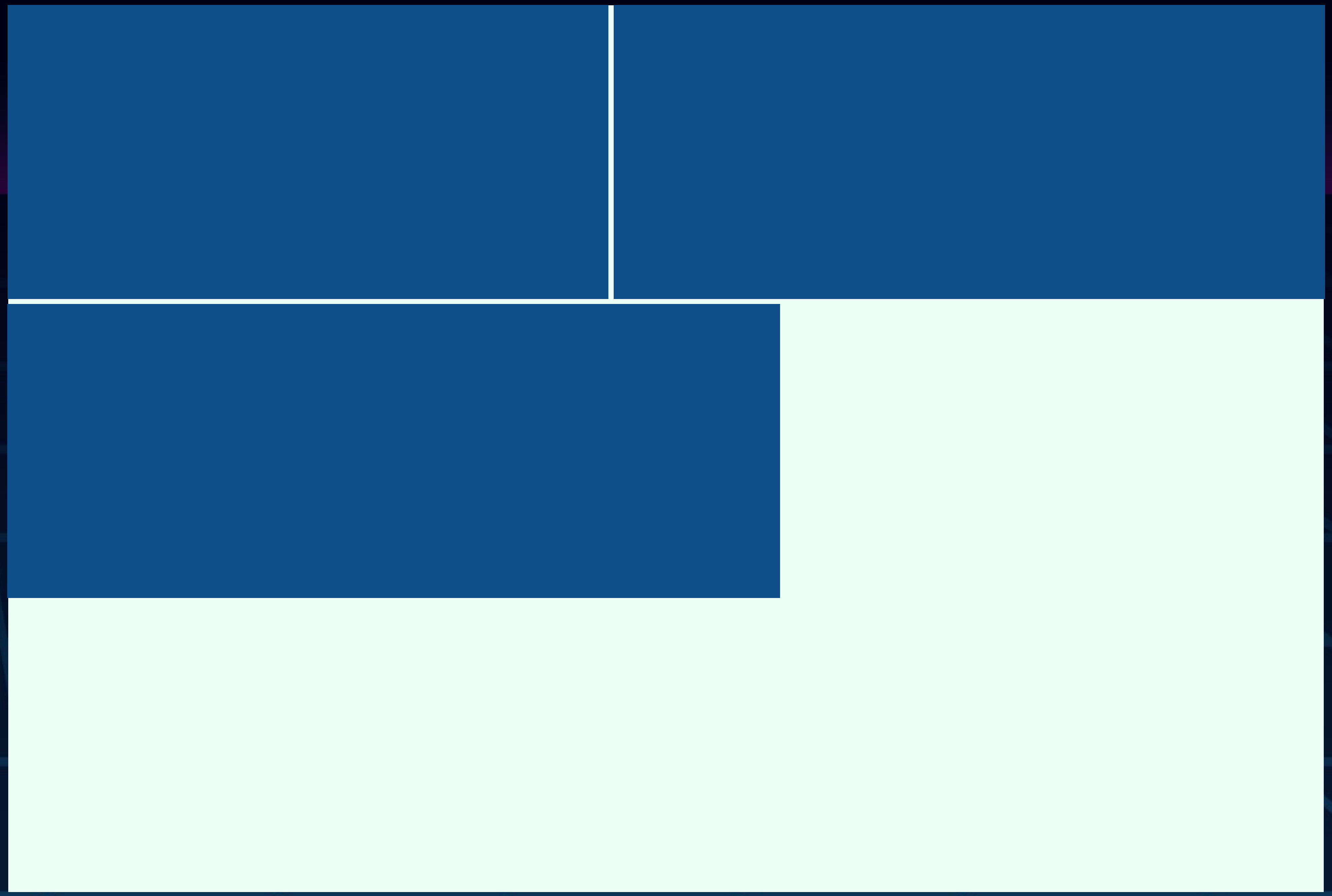


BLOCK vs INLINE

Layout vertically, one below the other

Display one after another in the direction that sentences run.





DISPLAY: INLINE;

A few HTML elements with inline as a default

```
<span>
<a>
<button>
<strong>
<em>
<img> *
```

DISPLAY: INLINE;

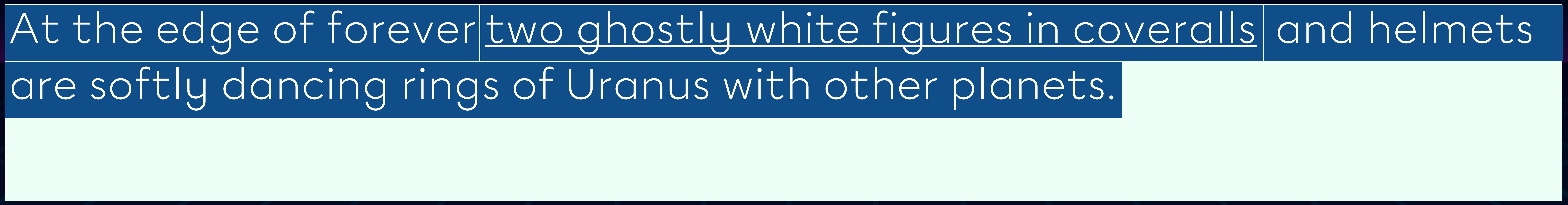
A few HTML elements with inline as a default

```
<span>
<a>
<button>
<strong>
<em>
<img> *
```

** image elements work a little differently than the others, but we'll get to that later*

DISPLAY: INLINE;

Display one after another in the direction that sentences run.



At the edge of forever two ghostly white figures in coveralls and helmets are softly dancing rings of Uranus with other planets.

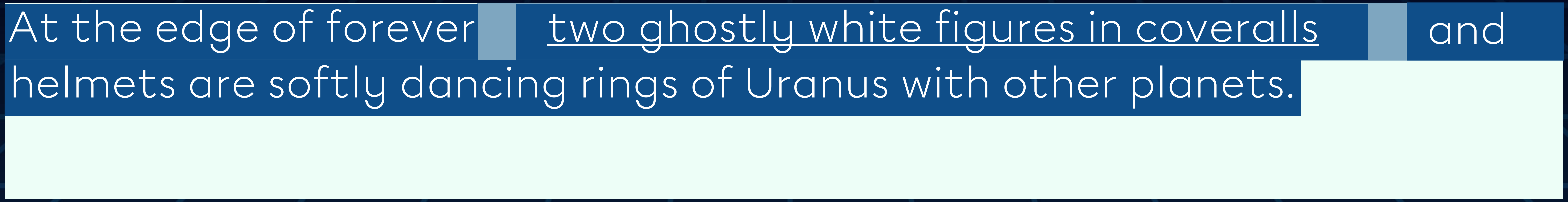
DISPLAY: INLINE;

Can take side spacing, but not top or bottom.

Cannot set width and height, set by size of content.

```
margin: 10px;

padding: 10px;
```

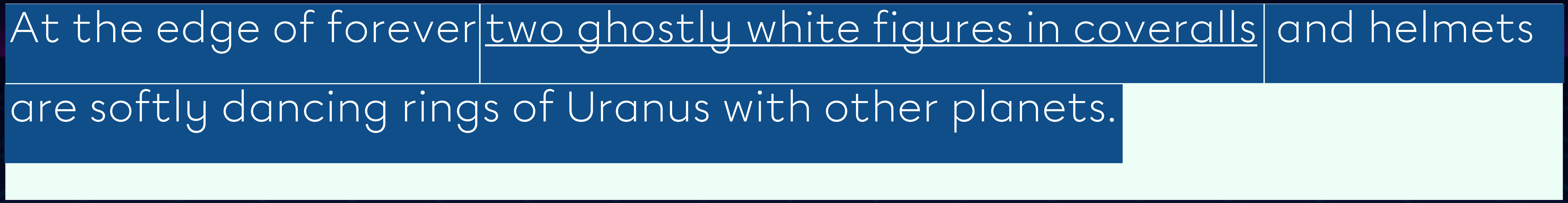


At the edge of forever two ghostly white figures in coveralls and helmets are softly dancing rings of Uranus with other planets.

DISPLAY: INLINE;

Top and bottom spacing comes from line height.

```
line-height: 1.5;
```

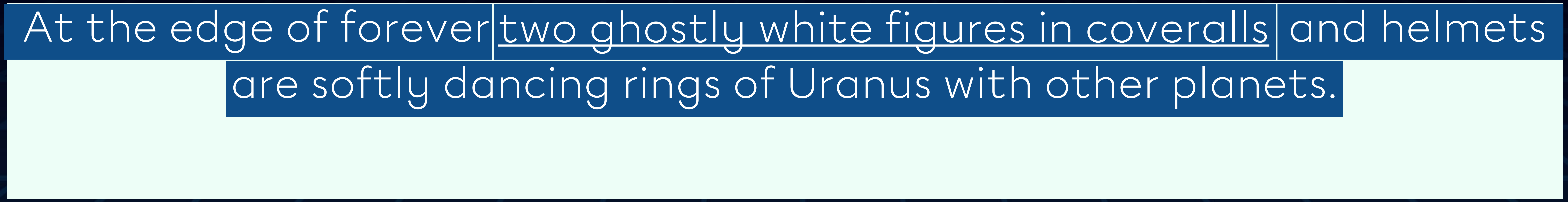


At the edge of forever two ghostly white figures in coveralls and helmets are softly dancing rings of Uranus with other planets.

DISPLAY: INLINE;

Center horizontally using text-align on the parent element.

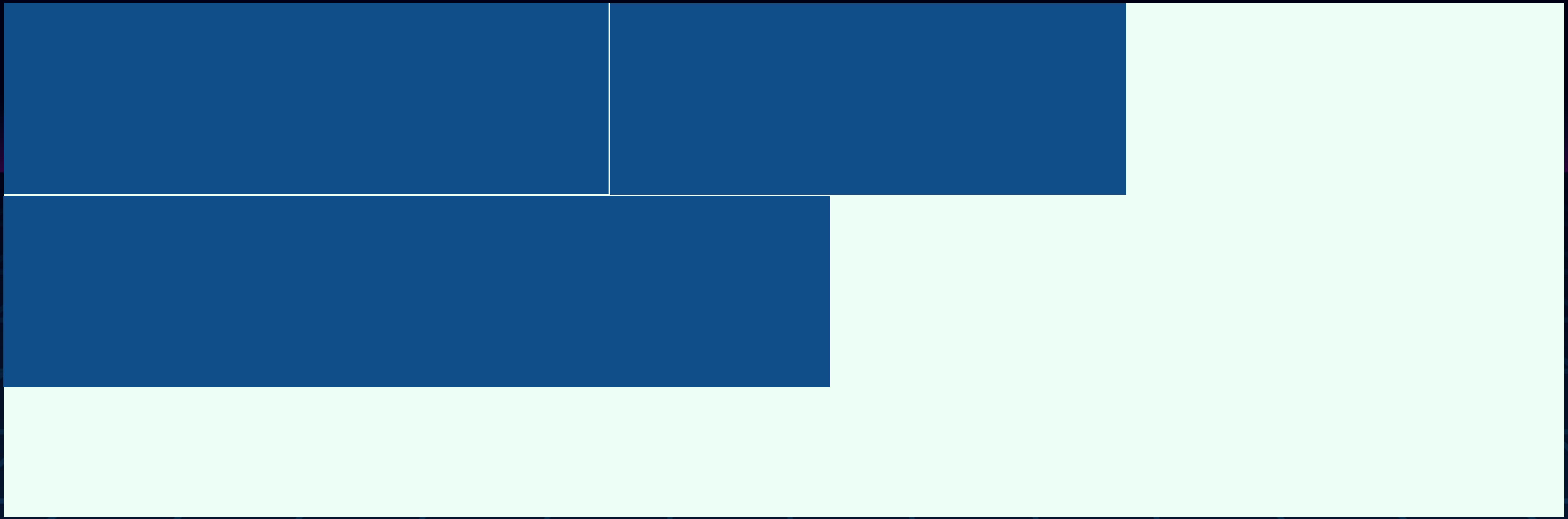
```
text-align: center;
```



At the edge of forever two ghostly white figures in coveralls and helmets are softly dancing rings of Uranus with other planets.

DISPLAY: INLINE-BLOCK;

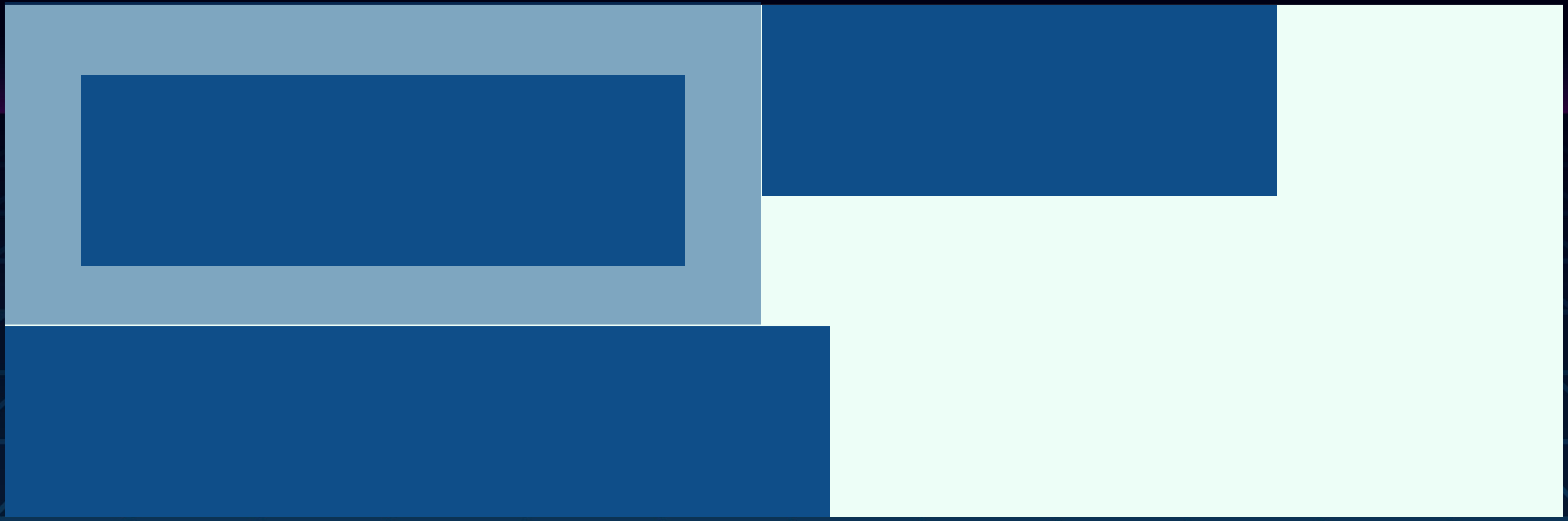
This is how images behave by default



DISPLAY: INLINE-BLOCK;

This is how images behave by default

```
margin: 20px;
```



DISPLAY: NONE;

Removes an element visually and the space it takes up in the layout so that the page renders as if that element does not exist.

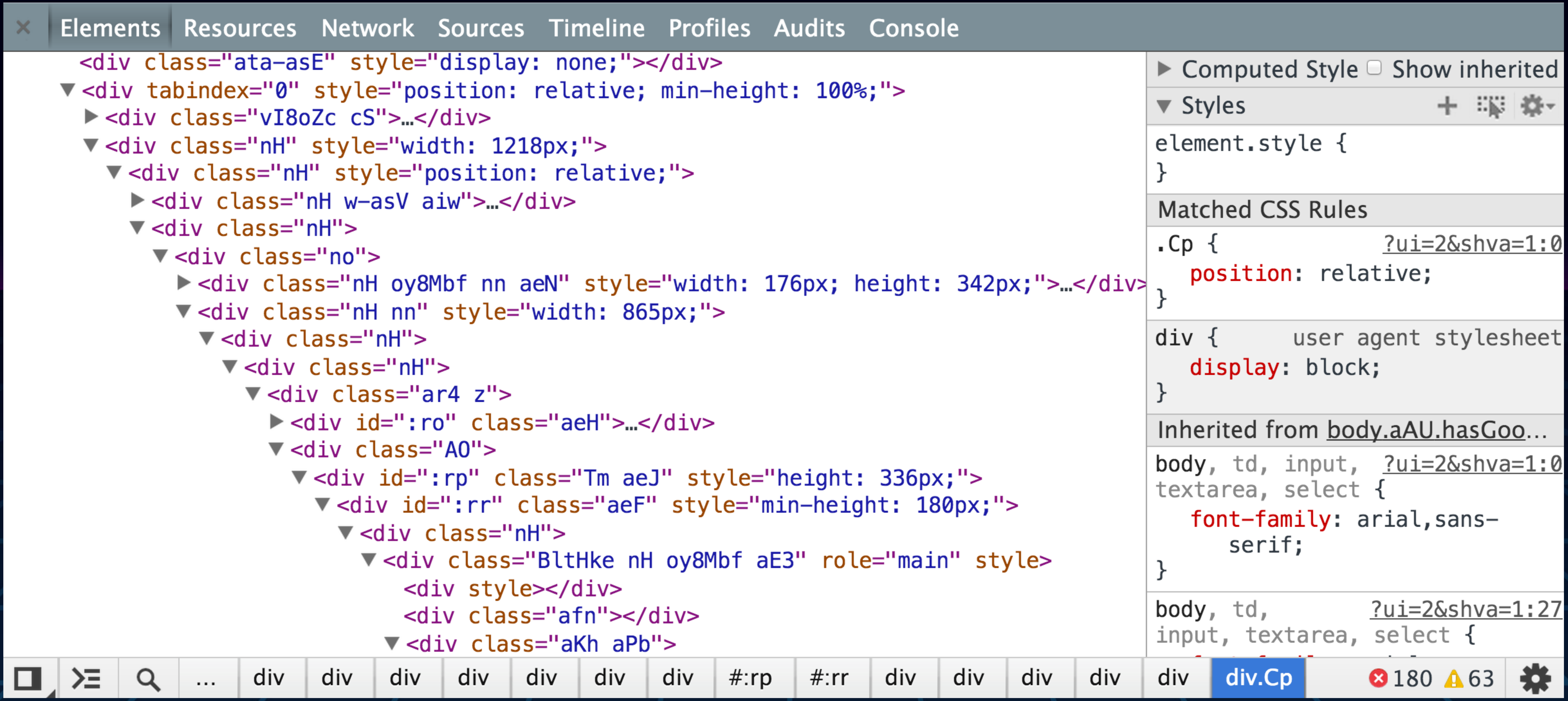
Removes the item from the accessibility tree, which means it will not be announced to screen readers. This includes all of its descendants.

	DISPLAY: NONE;	VISIBILITY: HIDDEN;	OPACITY: 0;
Removes the space it would take up so that the element has no effect on layout.	X		
Removes the item from the accessibility tree, which means it will not be announced to screen readers.	X	X	
Preserves the element's space in the layout.		X	X
Can be animated and transitioned with CSS			X

DISPLAY & ACCESSIBILITY

You can always change the display property to fit your layout needs. A button can become block element, a div can become inline element, etc. It's more important to use the proper semantic HTML element, and then change how it displays, rather than choose an element for its default display property.

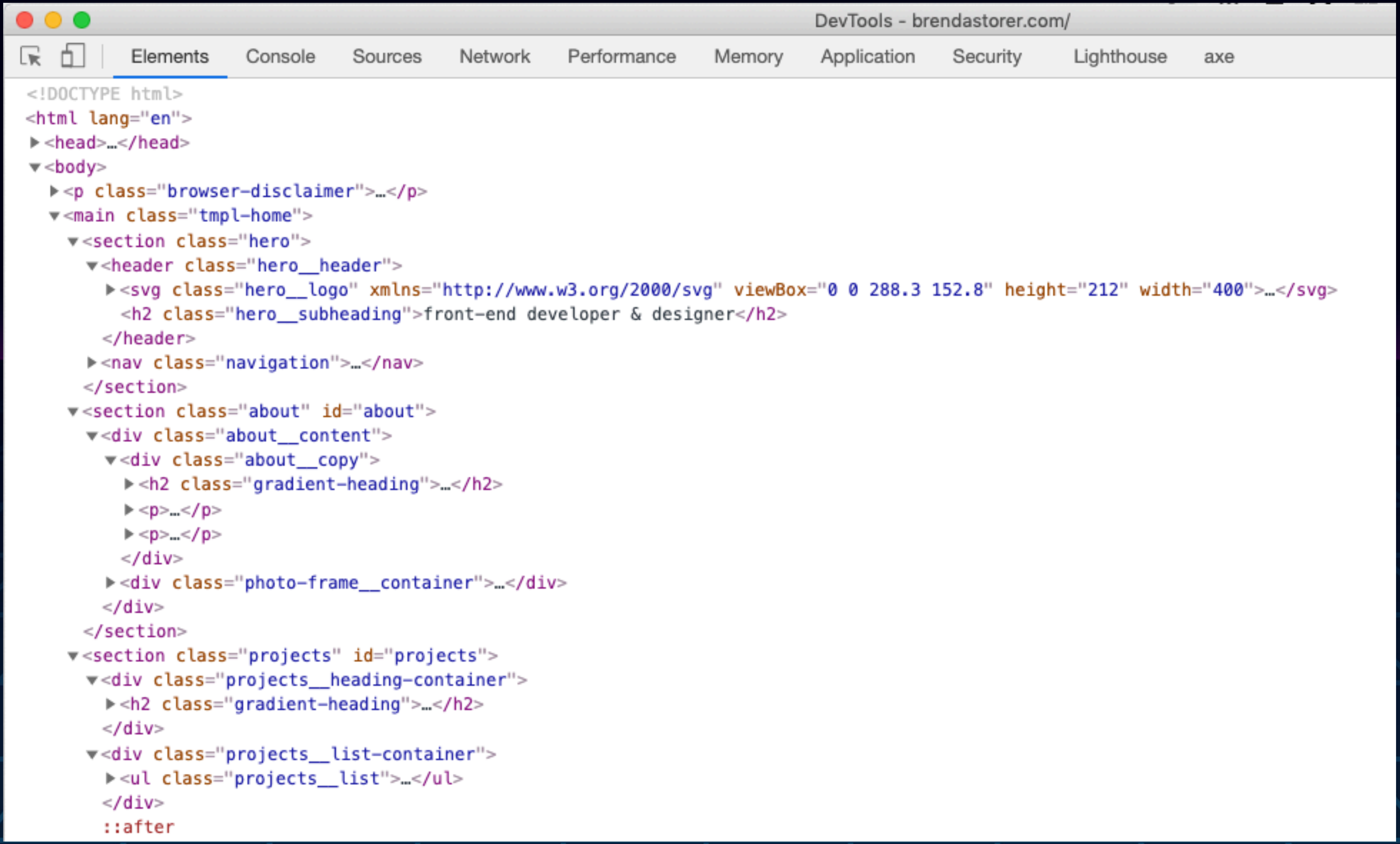
DIV SOUP 🤮



```
<div class="ata-asE" style="display: none;"></div>
▼<div tabindex="0" style="position: relative; min-height: 100%;">
 ▶<div class="vI8oZc cS">…</div>
 ▼<div class="nH" style="width: 1218px;">
  ▼<div class="nH" style="position: relative;">
   ▶<div class="nH w-asV aiw">…</div>
   ▼<div class="nH">
    ▼<div class="no">
     ▶<div class="nH oy8Mbf nn aeN" style="width: 176px; height: 342px;">…</div>
     ▼<div class="nH nn" style="width: 865px;">
      ▼<div class="nH">
       ▼<div class="nH">
        ▼<div class="ar4 z">
         ▶<div id=":ro" class="aeH">…</div>
         ▼<div class="AO">
          ▼<div id=":rp" class="Tm aeJ" style="height: 336px;">
           ▼<div id=":rr" class="aeF" style="min-height: 180px;">
            ▼<div class="nH">
             ▼<div class="BltHke nH oy8Mbf aE3" role="main" style>
               <div style></div>
               <div class="afn"></div>
              ▼<div class="aKh aPb">
```

```
element.style {
}
Matched CSS Rules
.Cp {                    ?ui=2&shva=1:0
  position: relative;
}
div {        user agent stylesheet
  display: block;
}
Inherited from body.aAU.hasGoo…
body, td, input,         ?ui=2&shva=1:0
textarea, select {
  font-family: arial,sans-
      serif;
}
body, td,               ?ui=2&shva=1:27
input, textarea, select {
```

MORE LIKE MINESTRONE

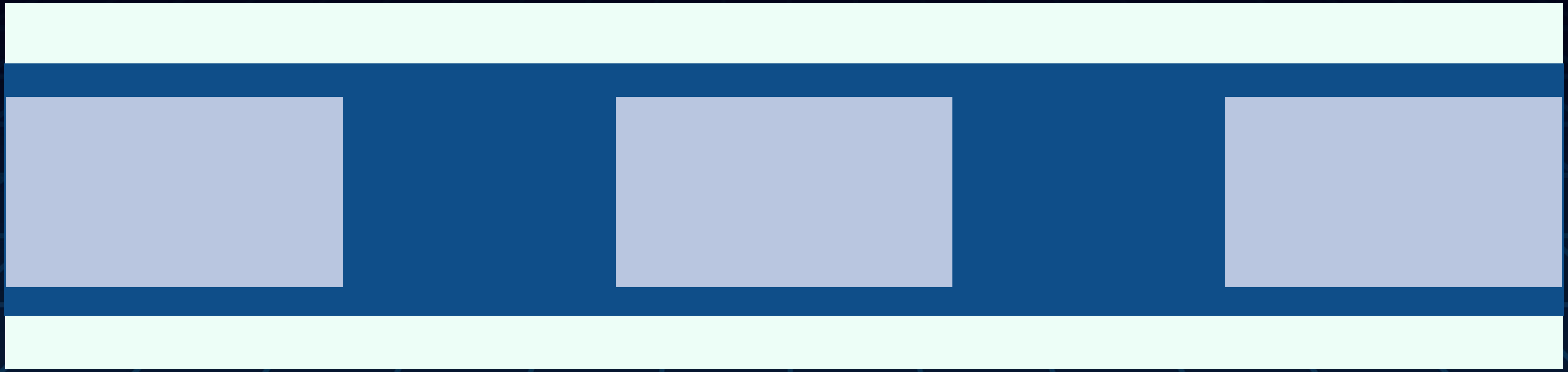


```
<!DOCTYPE html>
<html lang="en">
 ▶<head>…</head>
 ▼<body>
   ▶<p class="browser-disclaimer">…</p>
   ▼<main class="tmpl-home">
     ▼<section class="hero">
       ▼<header class="hero__header">
         ▶<svg class="hero__logo" xmlns="http://www.w3.org/2000/svg" viewBox="0 0 288.3 152.8" height="212" width="400">…</svg>
          <h2 class="hero__subheading">front-end developer & designer</h2>
        </header>
       ▶<nav class="navigation">…</nav>
      </section>
     ▼<section class="about" id="about">
       ▼<div class="about__content">
         ▼<div class="about__copy">
           ▶<h2 class="gradient-heading">…</h2>
           ▶<p>…</p>
           ▶<p>…</p>
          </div>
         ▶<div class="photo-frame__container">…</div>
        </div>
      </section>
     ▼<section class="projects" id="projects">
       ▼<div class="projects__heading-container">
         ▶<h2 class="gradient-heading">…</h2>
        </div>
       ▼<div class="projects__list-container">
         ▶<ul class="projects__list">…</ul>
        </div>
        ::after
```

DISPLAY: FLEX;

Used on a parent container, to affect its children.

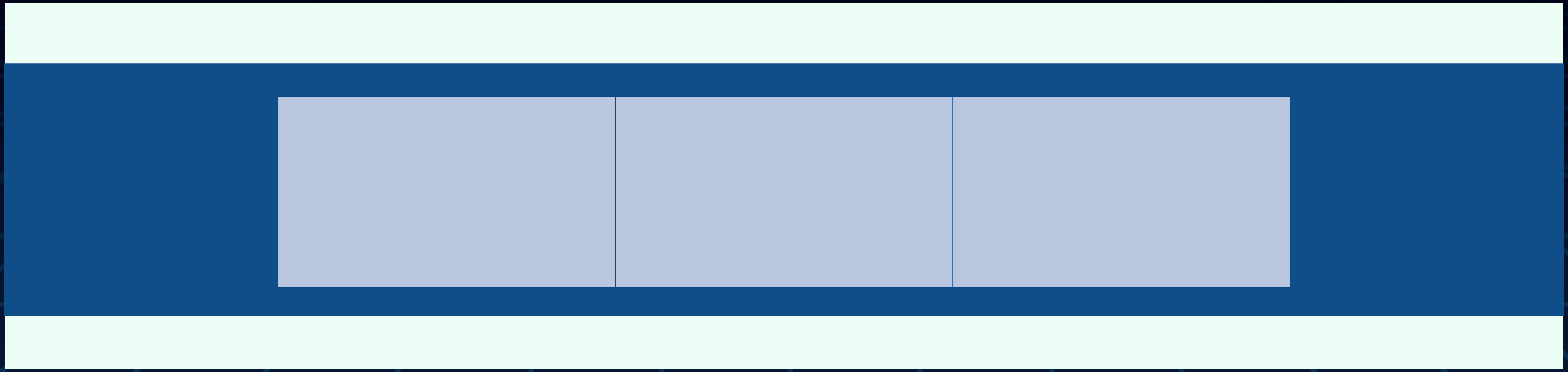
```
display: flex;
align-items: center;
justify-content: space-between;
```



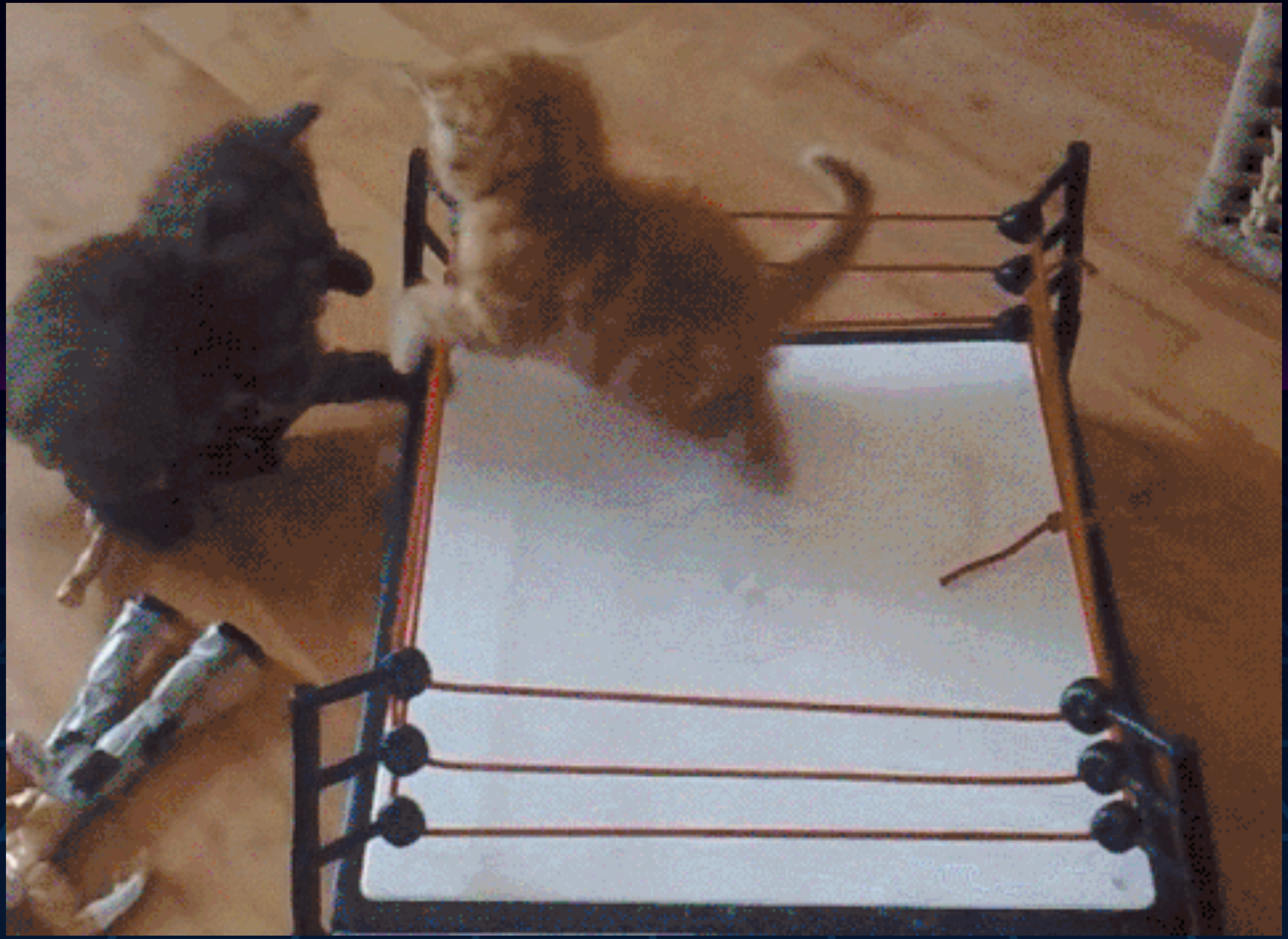
DISPLAY: FLEX;

Vertical centering made easy!

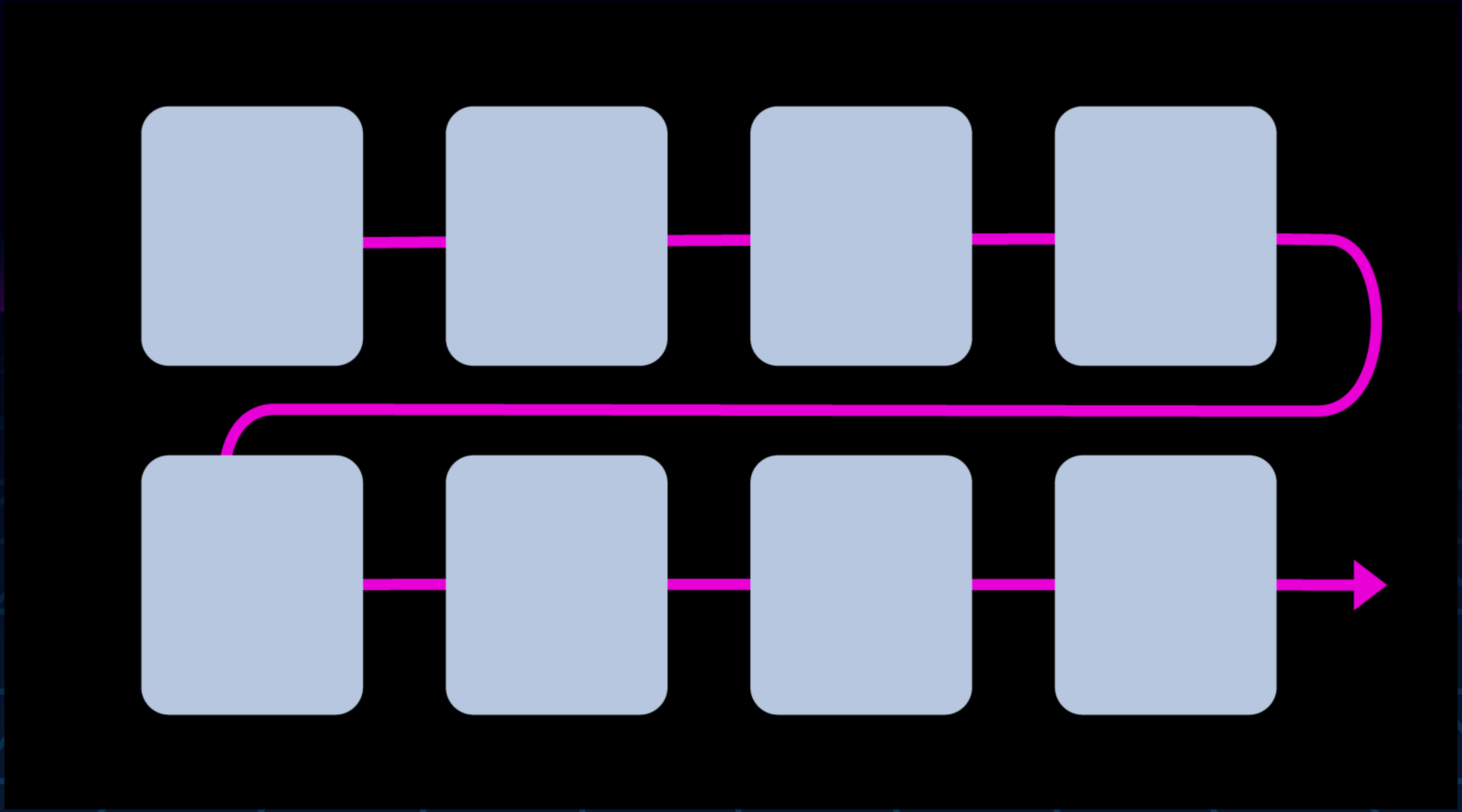
```
display: flex;
align-items: center;
justify-content: center;
```



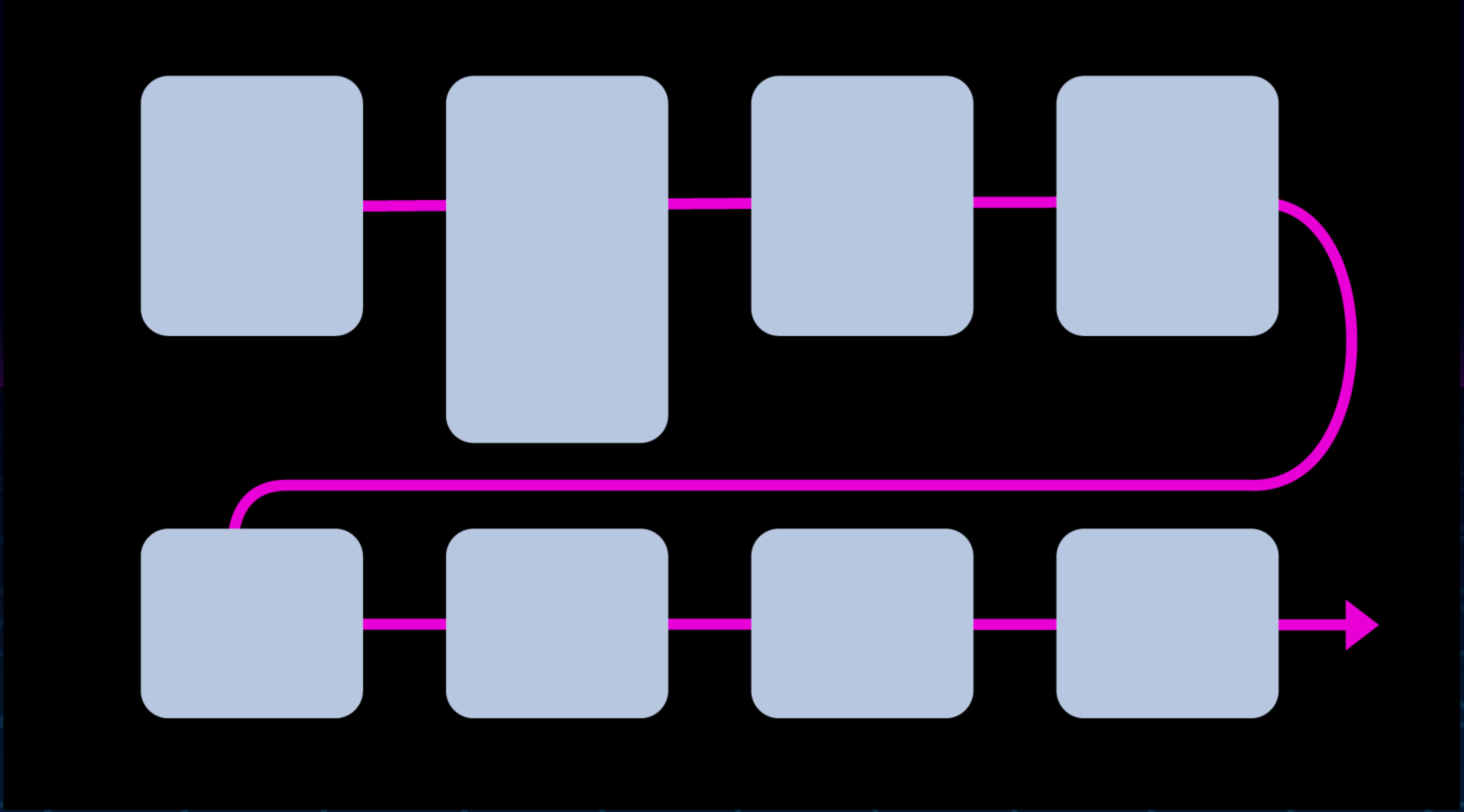
FLEXBOX VS. CSS GRID



FLEXBOX & GRID FRAMEWORKS ARE ONLY ONE DIMENSIONAL



UH OH.



GRID FRAMEWORKS

To achieve a grid layout in the past required complex hacks using the only CSS that was available at the time, never meant to do more than layout basic documents. These frameworks were once standard and really useful.

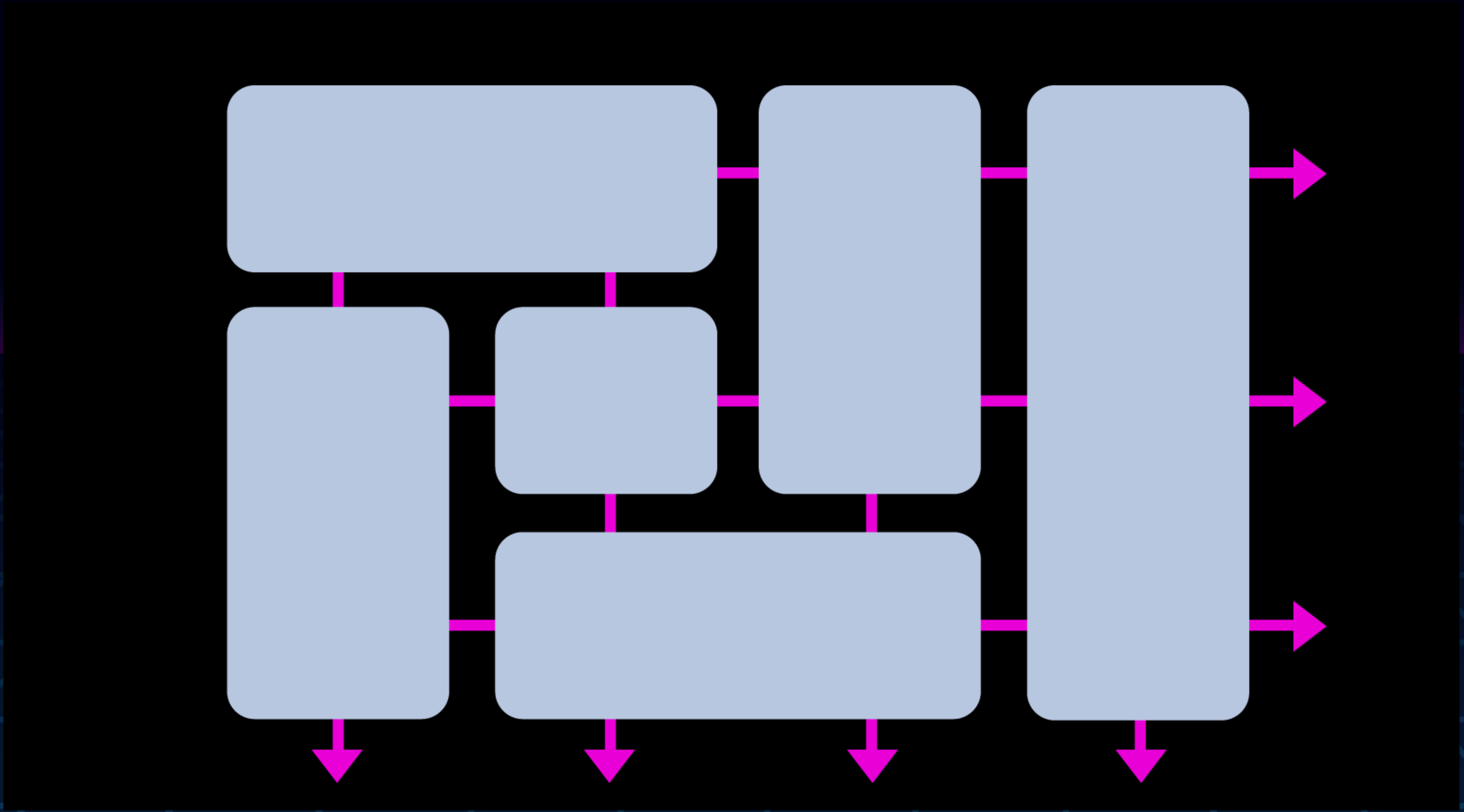
Bootstrap	Skeleton
Foundation	Neat
Bulma	Suzy
Materialize	Simple Grid

GRID FRAMEWORKS

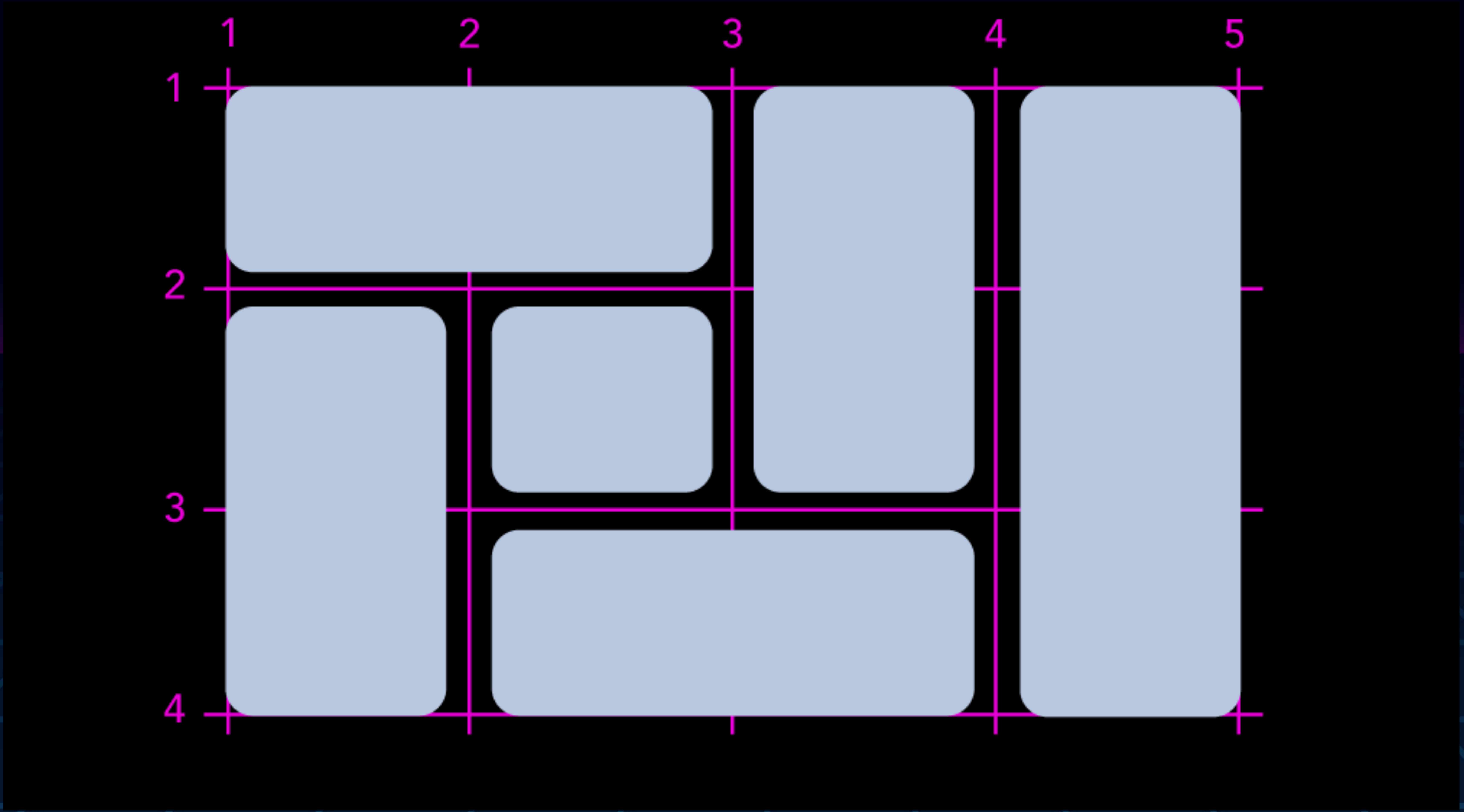
Grid Frameworks rely on specific markup structure and class names to achieve layout. At first they used floats or inline-block. Later they used Flexbox. Still, none of these CSS properties were meant to lay out content in this way.

CSS Grid was built to solve that problem.

CSS GRID ALLOWS YOU TO LAY OUT ELEMENTS IN 2D, ROWS & COLUMNS



PLACING ITEMS ON THE GRID



https://codepen.io/brendamarienyc/pres/QGKrmw

PRACTICE

CSS Grid Garden

REMEMBER

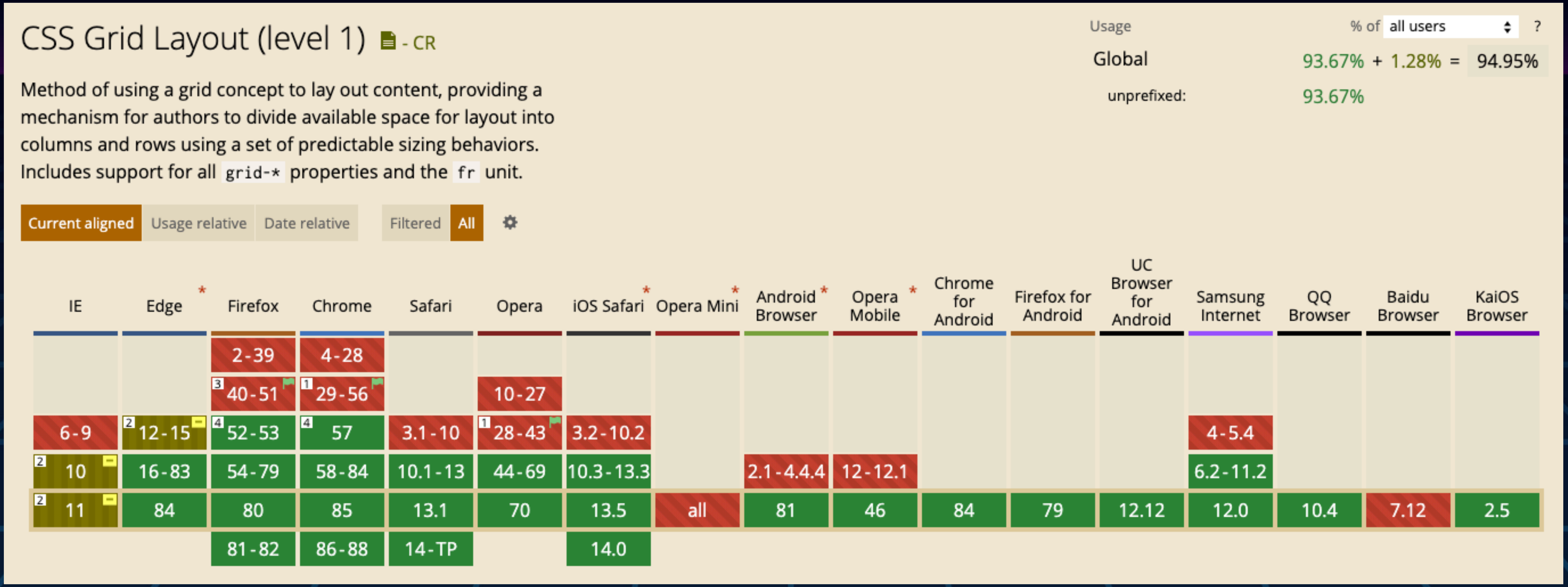
Flexbox and CSS Grid are display classes that you apply to container elements to affect the layout of their direct descendents.

Troubleshooting with the Firefox Grid and Flexbox inspectors is life saving.

There is more than one way to achieve the same goal.

CSS GRID IS THE NEW STANDARD

If anyone tells you it doesn’t have enough browser support, you can kindly refer them to caniuse.com



CSS Grid Layout (level 1) - CR

Usage | % of all users | ?

Global 93.67% + 1.28% = 94.95%

unprefixed: 93.67%

Method of using a grid concept to lay out content, providing a mechanism for authors to divide available space for layout into columns and rows using a set of predictable sizing behaviors. Includes support for all `grid-*` properties and the `fr` unit.

Current aligned | Usage relative | Date relative | Filtered | All

IE	Edge *	Firefox	Chrome	Safari	Opera	iOS Safari *	Opera Mini *	Android Browser *	Opera Mobile *	Chrome for Android	Firefox for Android	UC Browser for Android	Samsung Internet	QQ Browser	Baidu Browser	KaiOS Browser
		2-39	4-28													
		[3] 40-51	[1] 29-56		10-27											
6-9	[2] 12-15	[4] 52-53	[4] 57	3.1-10	[1] 28-43	3.2-10.2							4-5.4			
[2] 10	16-83	54-79	58-84	10.1-13	44-69	10.3-13.3		2.1-4.4.4	12-12.1				6.2-11.2			
[2] 11	84	80	85	13.1	70	13.5	all	81	46	84	79	12.12	12.0	10.4	7.12	2.5
		81-82	86-88	14-TP		14.0										

RESOURCES

GRID

[CSS Grid Garden]

[CSS Tricks Complete Guide to Grid]

[Layout Land - Jen Simmons Basics of CSS Grid: The Big Picture]

[Grid By Example - Rachel Andrew]

DISPLAY PROPERTIES

MDN Block and Inline layout flow

MDN Display none accessibility concerns

Deeper explanation into hiding content accessibly

Joni Trythall’s Flexbox Cheatsheet

Centering using Flexbox from Digital Ocean

Fairly balanced perspective on using Grid

Frameworks vs CSS Grid

RESOURCES FROM ME

Codepen Collection of CSS Grid Examples

Grub Gallery (responsive grid example)

THANK YOU!

@brendamarienyc

http://brendastorer.com