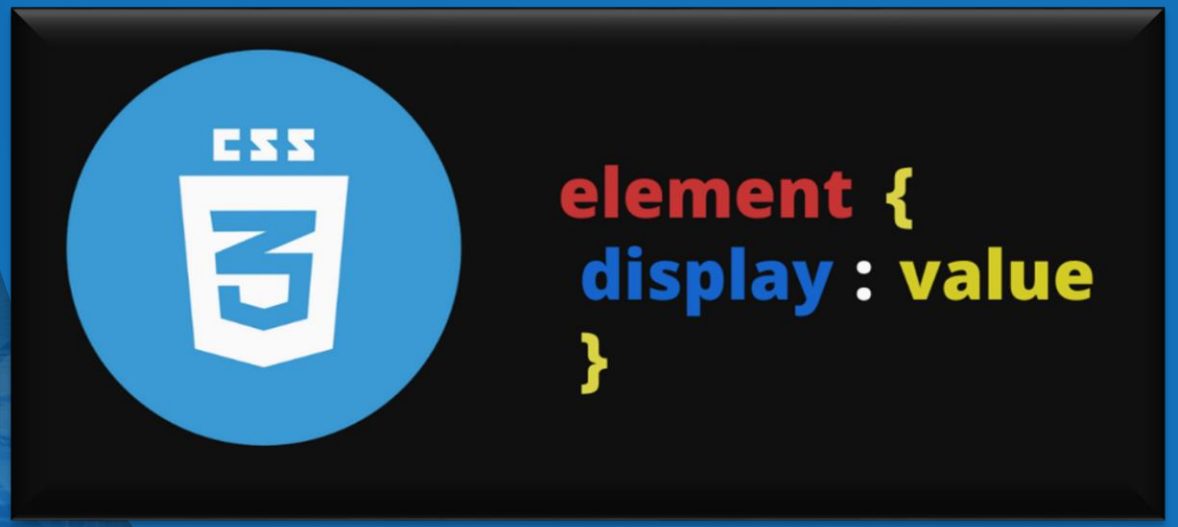
netlink
digital solutions

Display Properties



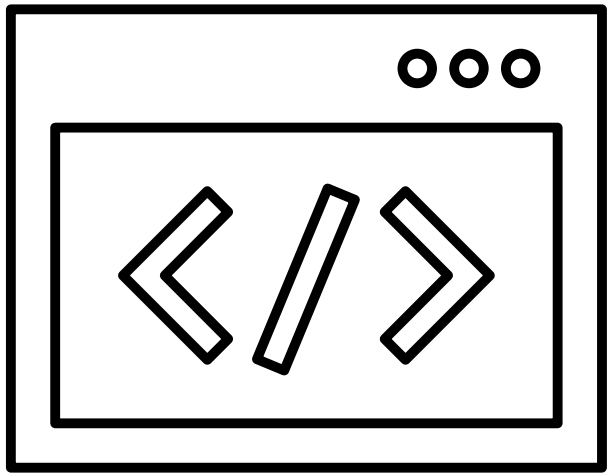
www.netlink.com

CSS Property : Display

- In CSS, the display property determines how an element looks.

- It is also a crucial part of the presentation of you HTML code as it has a **significant impact on layouts.**

- In fact, to use the **modern Flexbox and Grid models**, you need to use the display property before you get access to their various properties and values. This is one reason why the display property is so important in CSS.



Basic Display Property Syntax

```
element {

display: value;

}
```

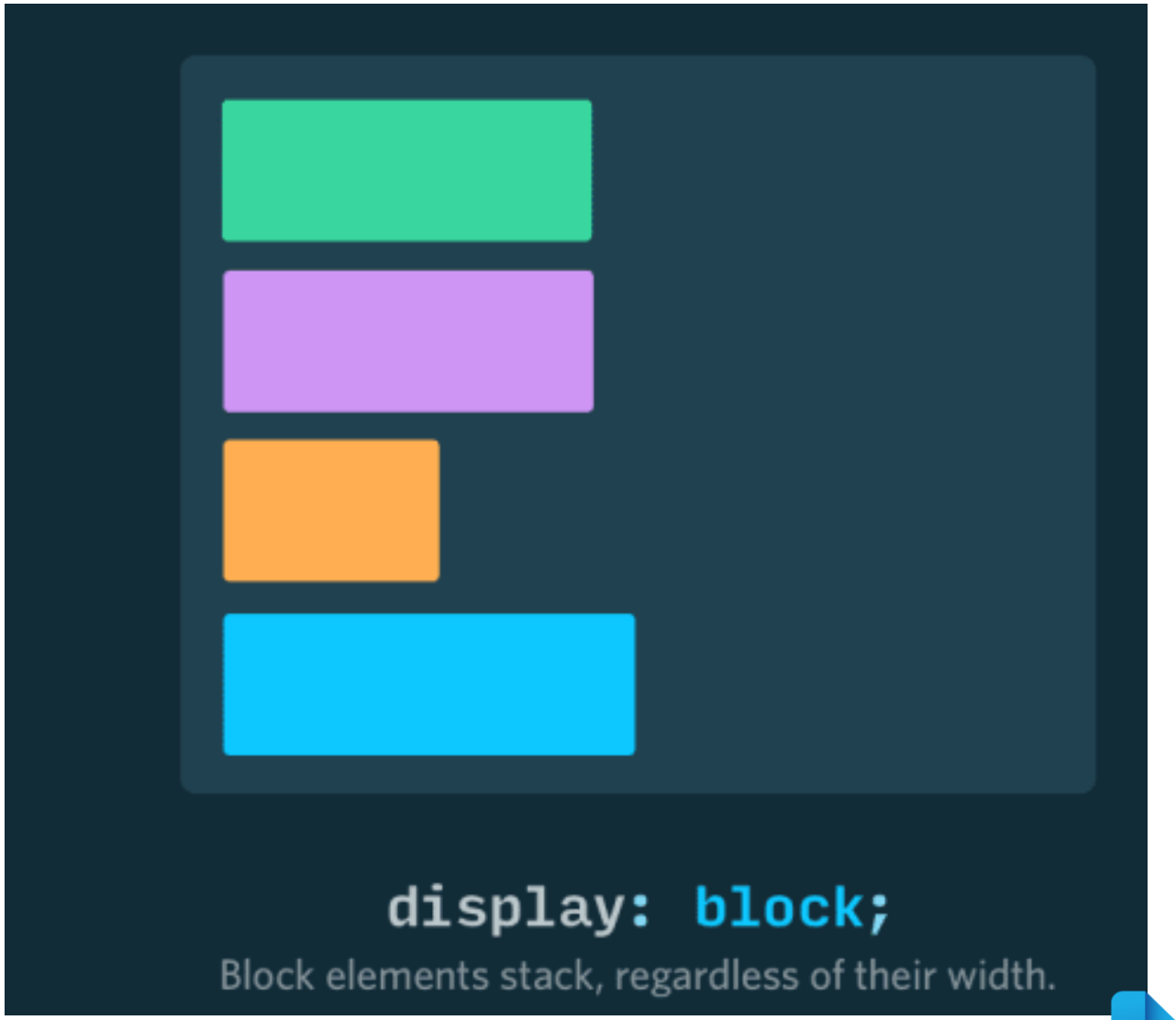
netlink

```css
div {
        display: block;
        display: inline;
        display: inline-block;
        display: flex;
        display: inline-flex;
        display: grid;
        display: inline-grid;
        display: inherit;
        display: none;
        display: initial;
        display: table;
        display: table-cell;
        display: table-row;
    }
```

The display property takes many different values such as inline, inline-block, block, table, and more, which all influence the layout and presentation of an element on the web page..

display : block

- display : block, by default, takes an entire width of the screen.
- It adds new line in the beginning and at end.



netlink

Block level Elements

Takes full width by default

Each starts with a new line

Width and height can be set

`<div>, <p>, <h1>, <footer>`

<address>	<article>	<aside>	<blockquote>	<canv
<dd>		<dl>	<dt>	<fiel
<figcaption>	<figure>	<footer>	<form>	<h1>-
<header>	<hr>		<main>	<nav>
<noscript>		<p>	<pre>	<sect
<table>	<tfoot>		<video>	

Some block elements (by default)

netlink

display : inline

An element with a display property set to inline will not start on a new line and **it will take up the remaining/available screen width.**

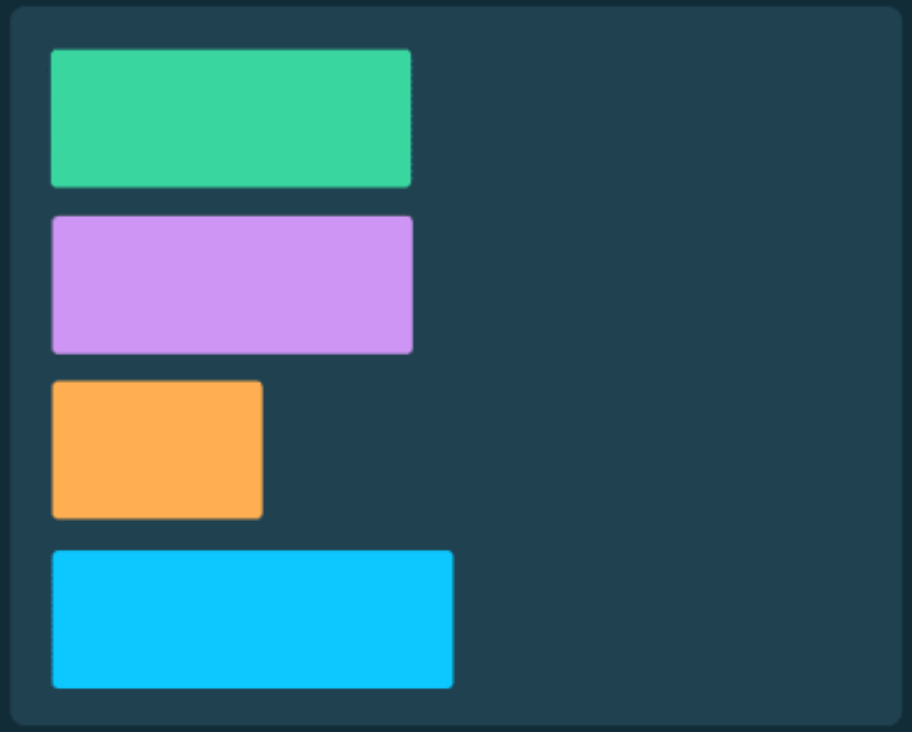
-> side by side

-> takes as much space as needed

->width and height can't be set!

Some elements are inline by default, like `<span>, <a>, <button>, <img>.`

BLOCK VS INLINE



`display: block;`

Block elements stack, regardless of their width.

`display: inline;`

Inline elements flow from one line to the next.

netlink

<a>	<abbr>	<acronym>	<b>	<bdo>
<big>	 	<button>	<cite>	<code>
<dfn>		<i>	<img>	<input>
<kbd>	<label>	<map>	<object>	<output>
<q>	<samp>	<script>	<select>	<small>
		<sub>	<sup>	<textarea>
<time>	<tt>	<var>		

Some inline elements (by default)

netlink

Display Property Values in CSS

(inline & block)

There are inline and block-level elements in CSS.

The difference between the two is that inline elements don't take up an entire space – that is, they don't start on a new line – but block elements do.

To implement the flex and grid layouts, you need to use the display property.

- What if we want to display inline-elements with height and width?

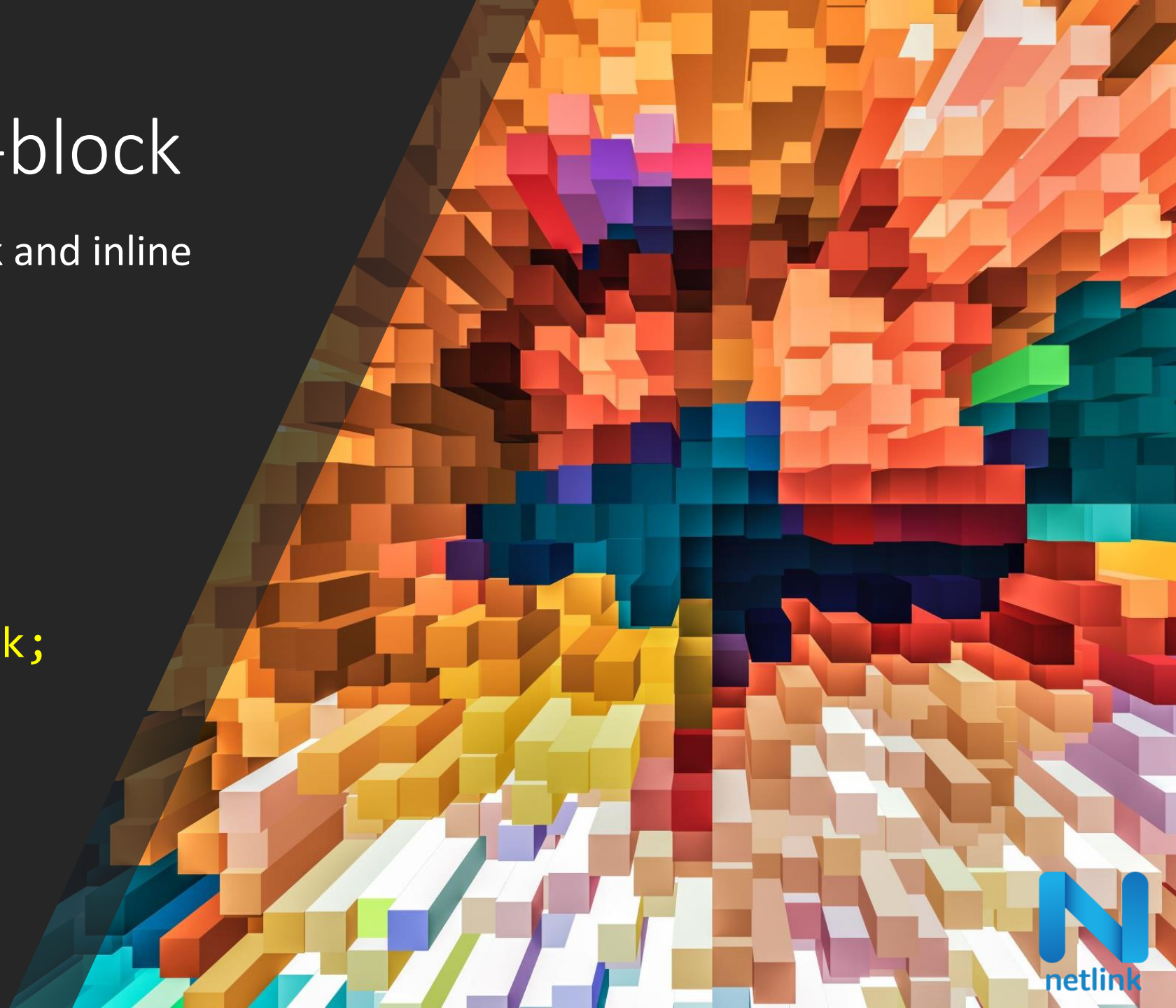
netlink

display : inline-block

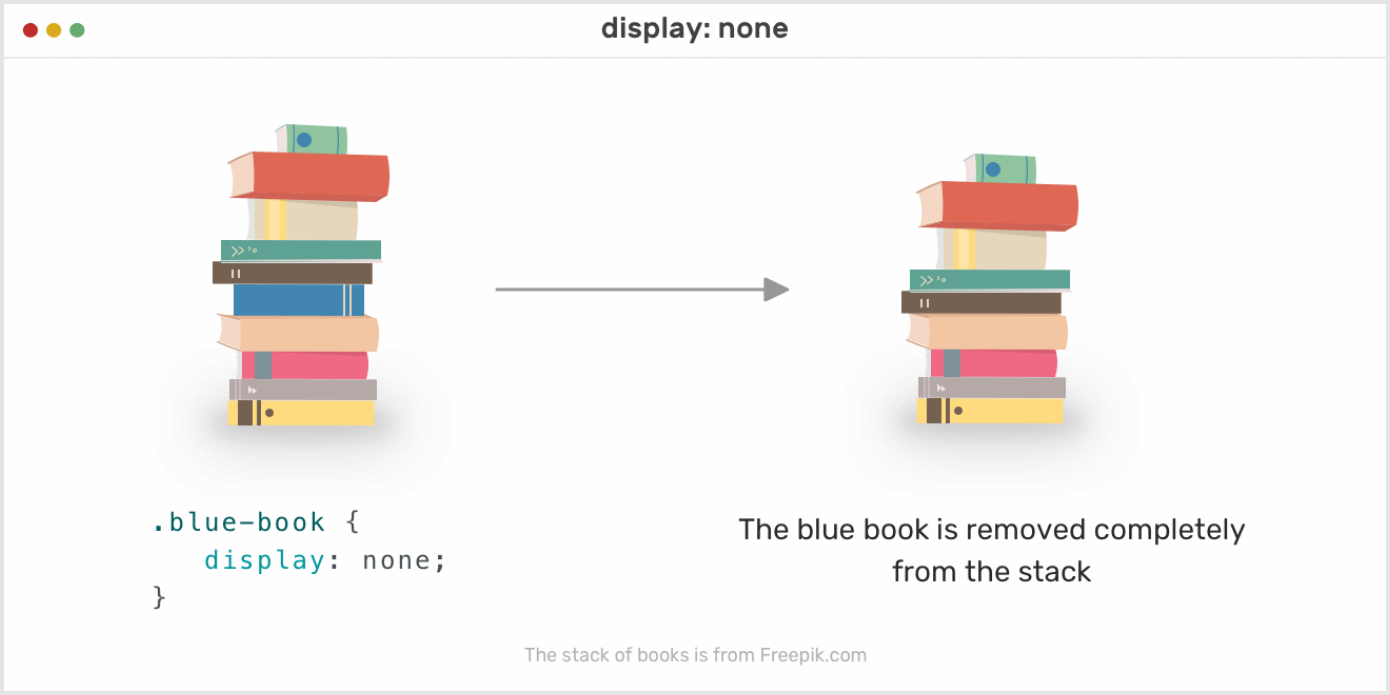
- It is a combination of block and inline block- elements.

- Syntax :

```
display : inline-block;
```



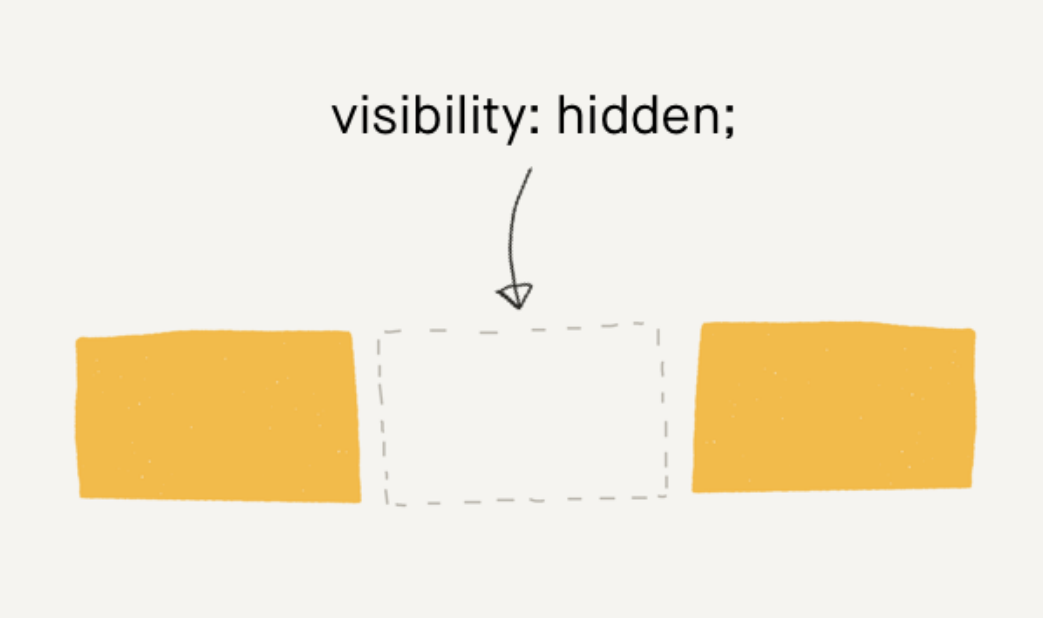
display : none

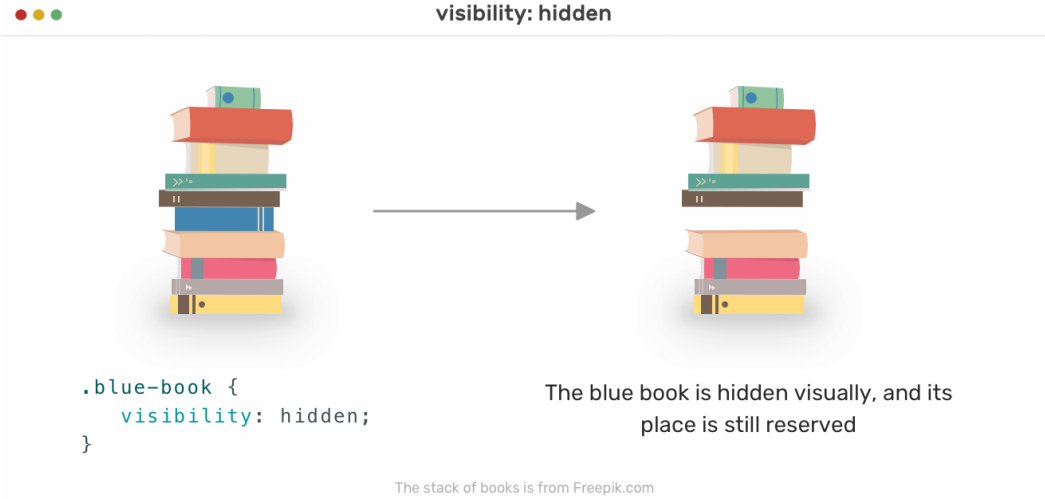


- `display: none;` is commonly used with JavaScript to hide and show elements without deleting and recreating them

netlink

visibility : hidden





netlink

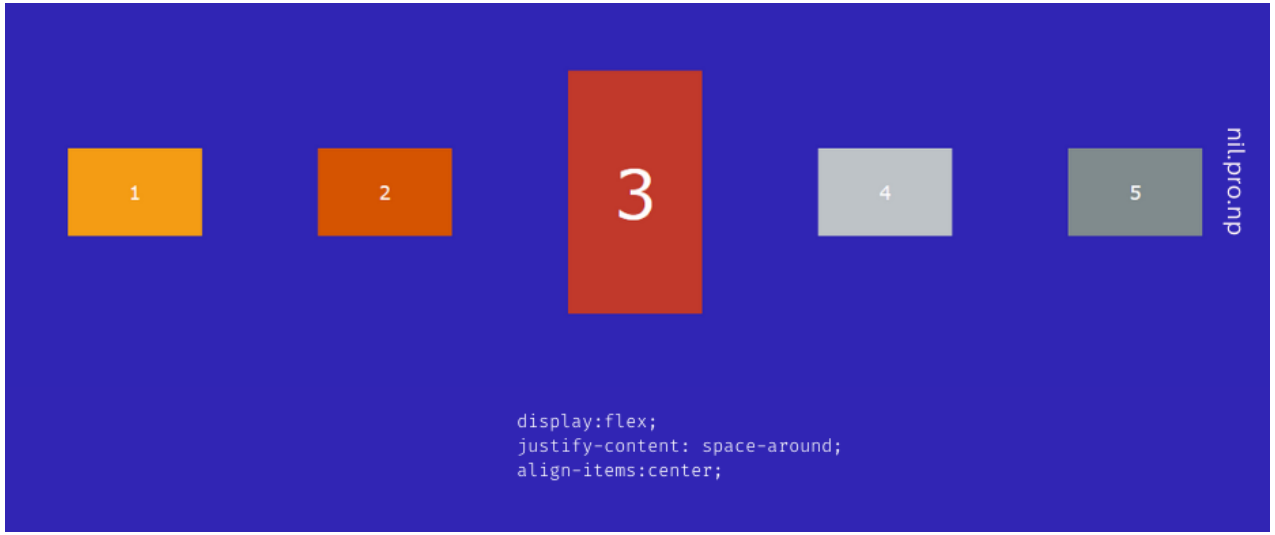
display: flex

The flex box is flexible box designed to build one-dimensional layouts in css. One dimensional means flexbox can build layout in one dimension at one time.

For two-dimensional layouts, we have **CSS Grids** that can handle both row and column.

Display:flex is used to build flexbox.

Flex can build one dimension layout which is better than float-based layout.



netlink

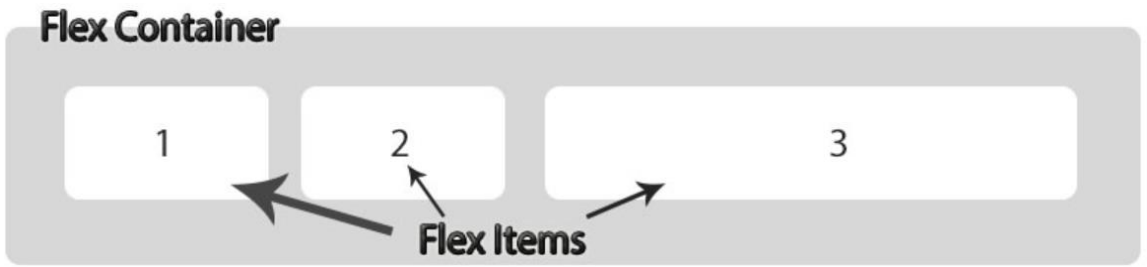
display : flex;

Properties of flex container

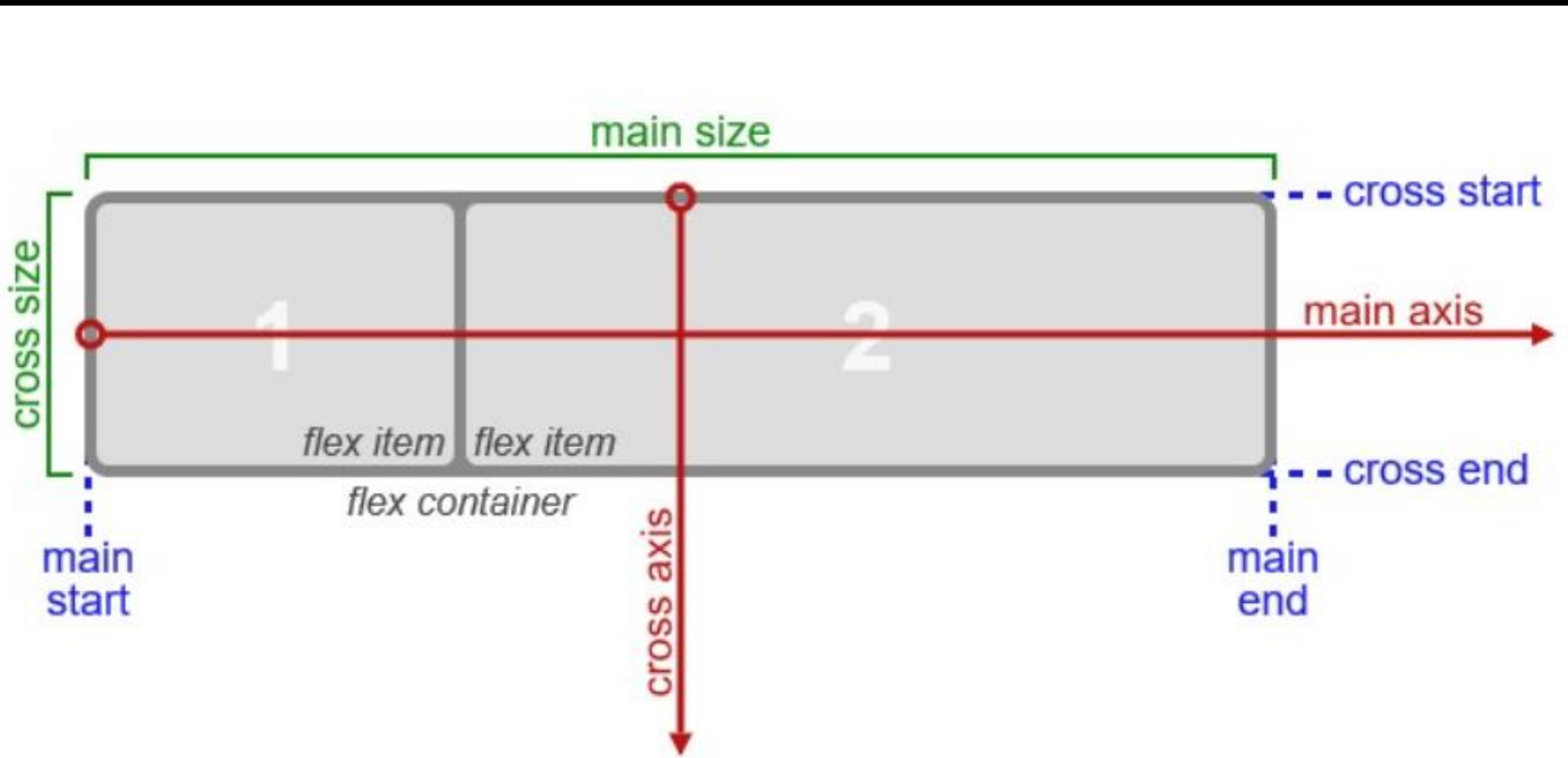
1. display *(flex or inline-flex)*
2. flex-direction
3. flex-wrap
4. flex-flow
5. justify-content
6. align-items
7. align-content

Properties of flex items (child of container)

1. order
2. flex-grow
3. flex-shrink
4. flex-basic
5. flex
6. align-self



netlink


main size
cross start
cross size
main axis
1
2
flex item
flex item
cross end
flex container
main
start
cross axis
main
end

CSS Flex rule that can be applied to Container

```
display: flex | inline-flex;
flex-direction: row | row-reverse | column | column-reverse; [default row]
flex-wrap: nowrap | wrap | wrap-reverse; [default nowrap]
flex-flow: <flex-direction> || <flex-wrap>; [default row nowrap]
justify-content: flex-start | flex-end | center | space-between | space-around
| space-evenly; [default flex-start]
align-items: stretch | flex-start | flex-end | center | baseline; [default
stretch]
align-content: flex-start | flex-end | center | space-between | space-around |
```

CSS Flex rule that can be applied to items/children within a container

```
order: <integer>; [ default  0 ]
flex-grow: <number>; [ default 0 ]
flex-basis: <length>; [ default auto ]
flex-shrink: <number>; [ default 1 ]
flex: none | [ <'flex-grow'> <'flex-shrink'>? || <'flex-basis'> ];
align-self: auto | flex-start | flex-end | center | baseline | stretch; [
default auto ]
```

Thank you

netlink