**Day 19**





**“Web Development + Security”**

**CSS Sizing Units:**

| **Unit** | **Meaning** | **Based On** | **Common Use** |
| --- | --- | --- | --- |
| **px** | Pixels (absolute unit) | Fixed size | Precise control, borders, icons |
| **em** | Relative to parent’s font size | Parent element | Buttons, nested text scaling |
| **rem** | Relative to root (HTML) font size | <html> | Consistent typography |
| **%** | Percentage | Parent element’s size | Responsive widths/heights |
| **vh** | Viewport height | 1vh = 1% of viewport height | Full-screen sections |
| **vw** | Viewport width | 1vw = 1% of viewport width | Responsive layouts, text |

**CSS display properties:**

**What is the CSS display Property?**

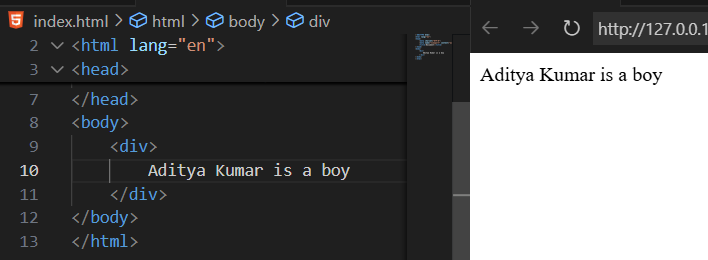
The display property defines how an element is shown (rendered) on a webpage — i.e., how it behaves in the document layout (block, inline, grid, flex, etc).

It controls whether elements sit side-by-side, stack vertically, or become layout containers.

**Common display values:**

| **Value** | **Description** | **Example Use** |
| --- | --- | --- |
| **block** | Takes full width, starts on a new line | <div>, <p>, <h1> |
| **inline** | Takes only as much width as needed, stays in same line | <span>, <a> |
| **inline-block** | Acts inline but allows width/height | Buttons, small boxes |
| **none** | Hides the element completely | Toggle visibility |
| **flex** | Turns element into a flex container (for alignment) | Navigation bars, layouts |
| **grid** | Turns element into a grid container | Complex layouts |
| **inline-flex / inline-grid** | Same as flex/grid but inline | Inline layouts |
| **table** | Acts like an HTML table | Special structure layouts |

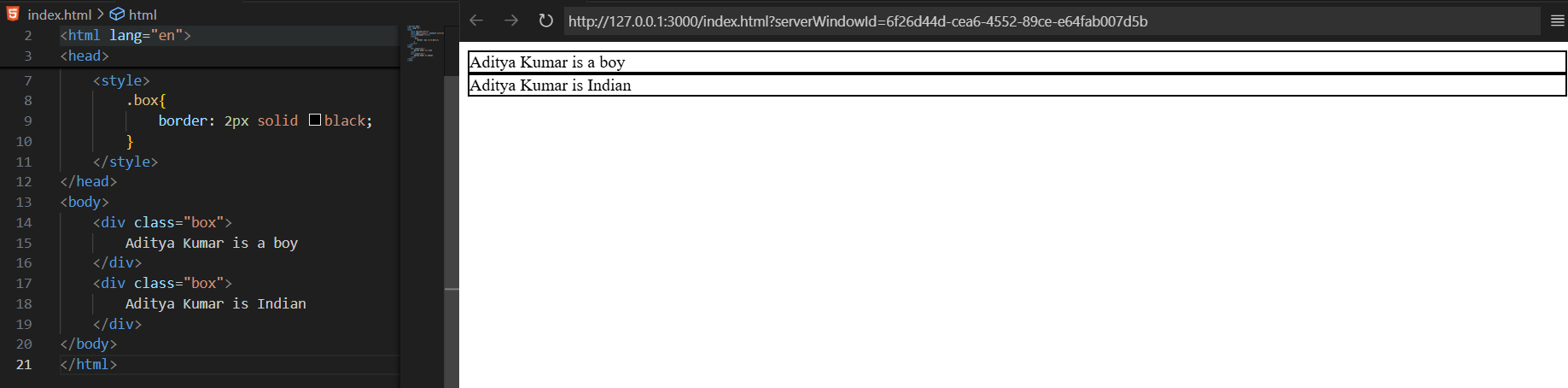
Example: a simple <div>



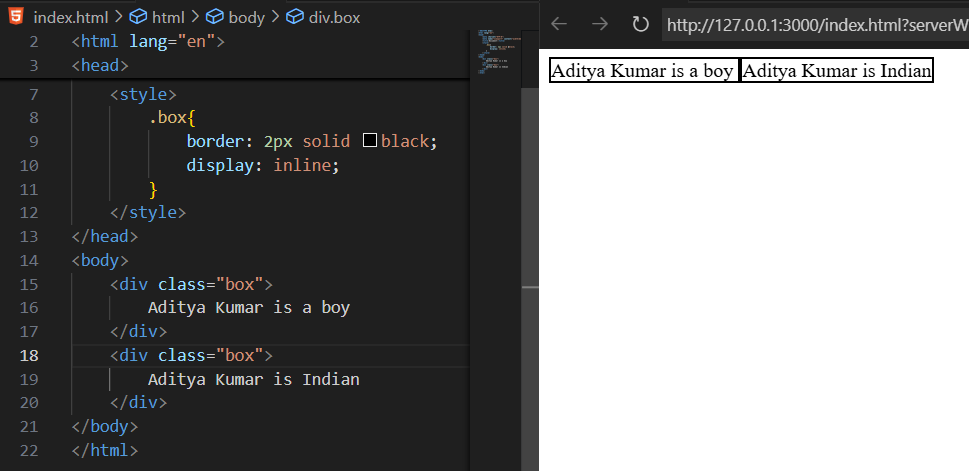
But, <div> is a block element. It means it will occupy whole block space. We can prove it by adding border to it. As shown below:



So, as <div> is block element, if we write one more <div> after it, then we will see that it goes to new line. As shown below:



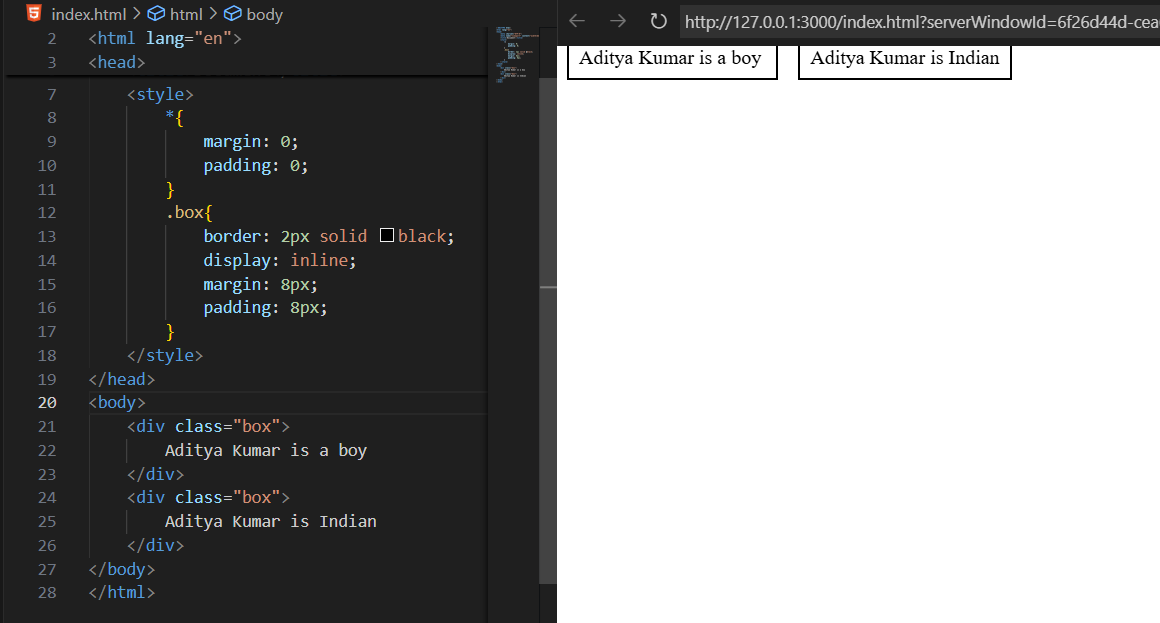
Example: changing the block elements to inline elements using the **display: inline** property



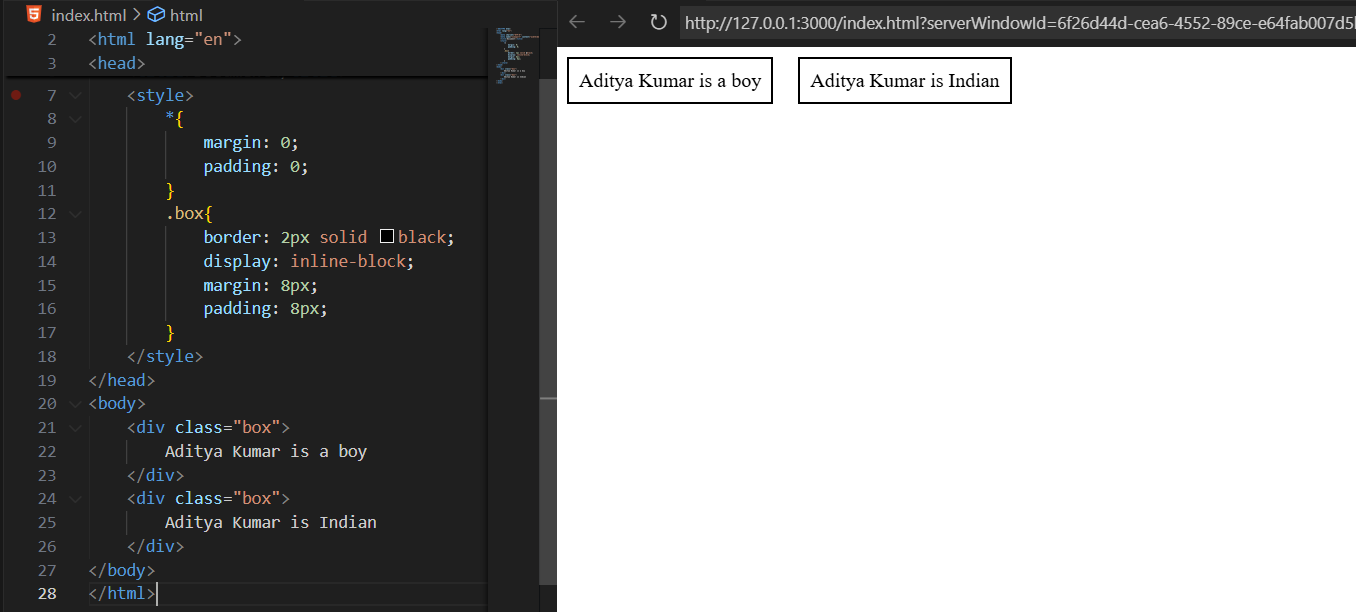
Clearly, the <div> adjusted in one line as per the inline element properties.

But, there is an issue with the “display:inline”, when we apply margin or padding to such, we will see that it will not get applied from the top.

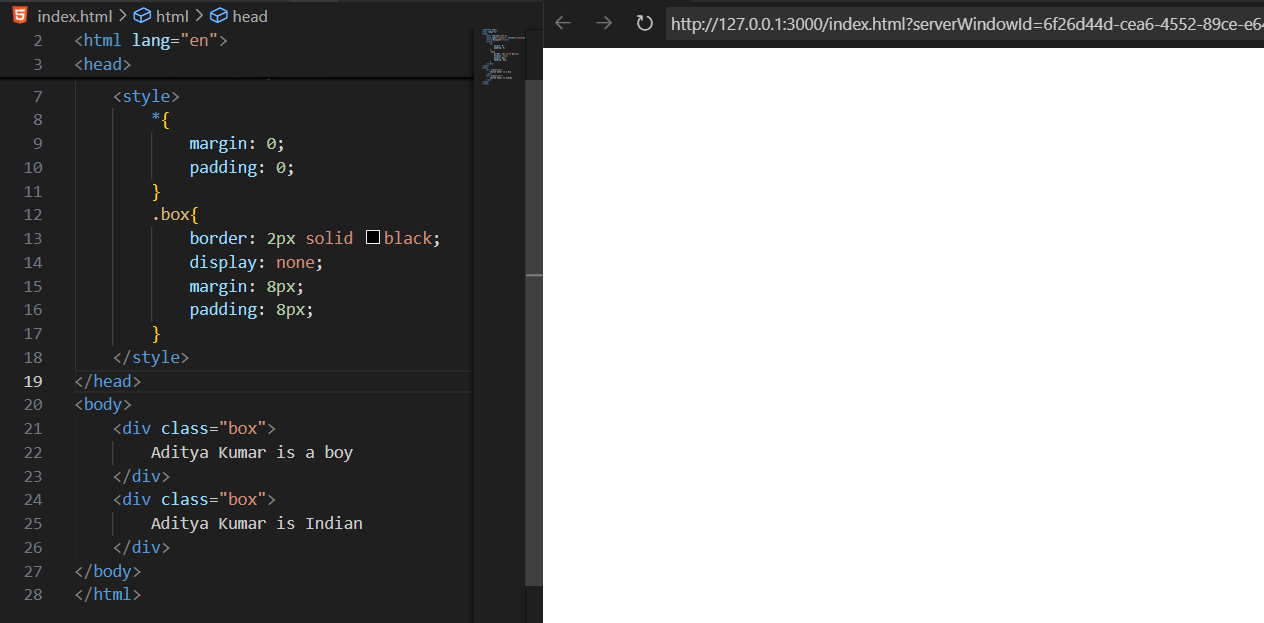
Example:



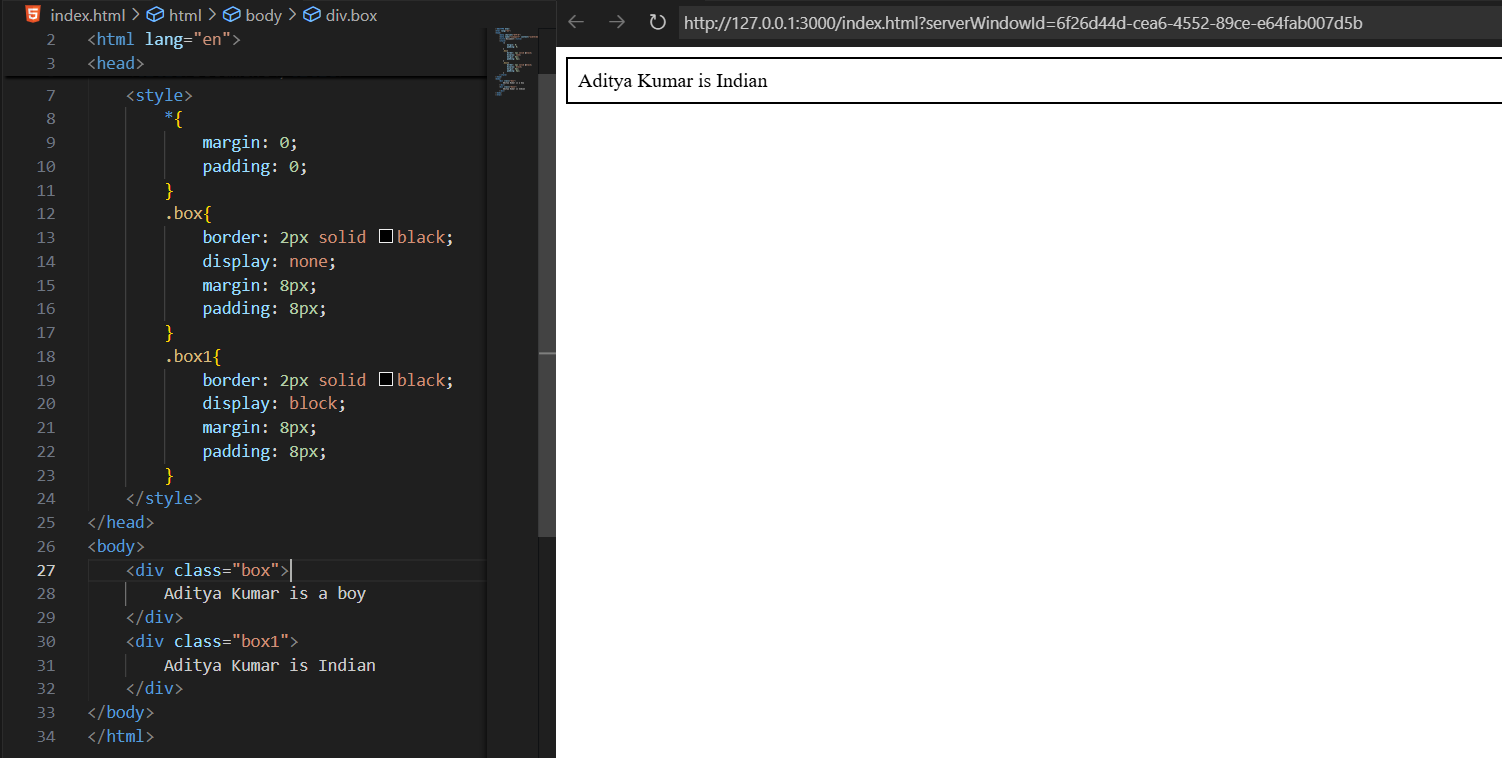
So, how to deal with this issue? We will use the “display:inline-block” instead of “display:inline”.



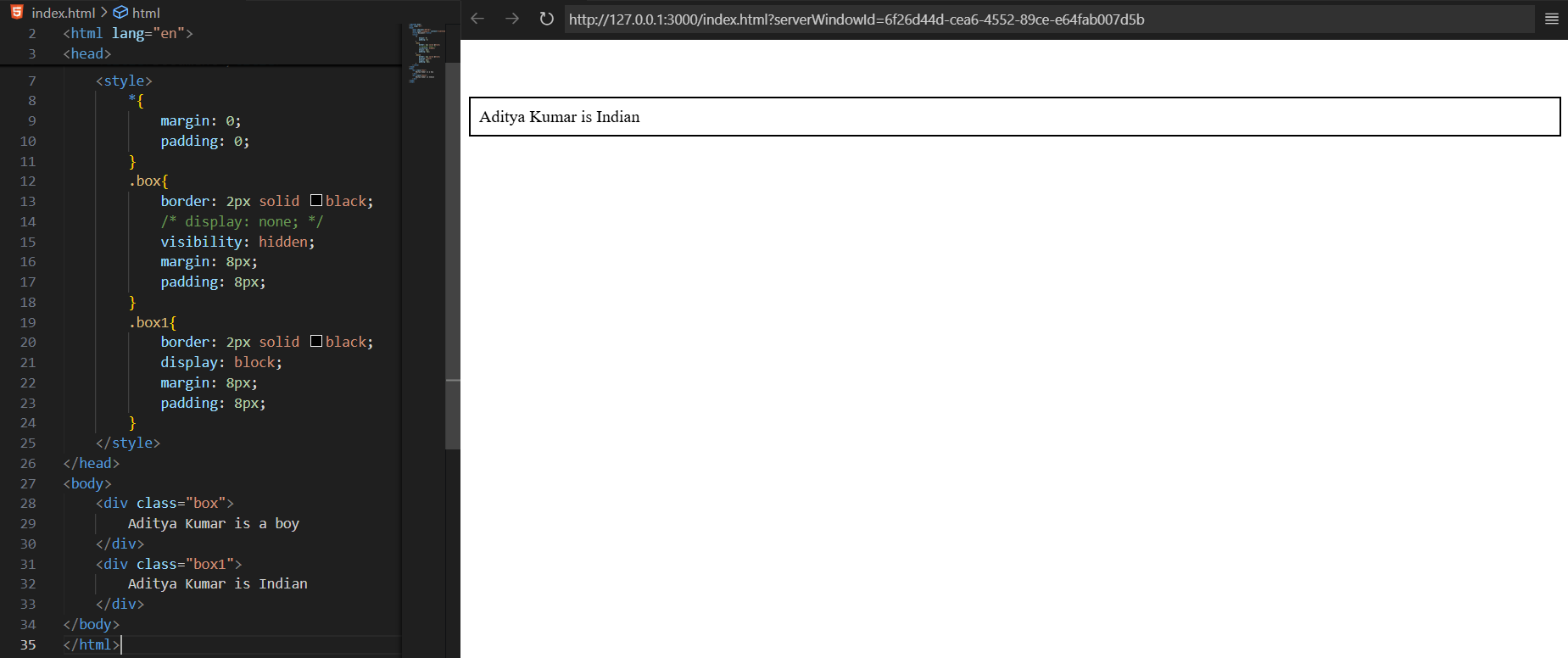
In some cases we may wish that a particular <div> should not be displayed, then in that case we will use the “display:none”, as shown below:



Also, we can observe that in the display:none, we are seeing that the element space is also removed. For demonstration:

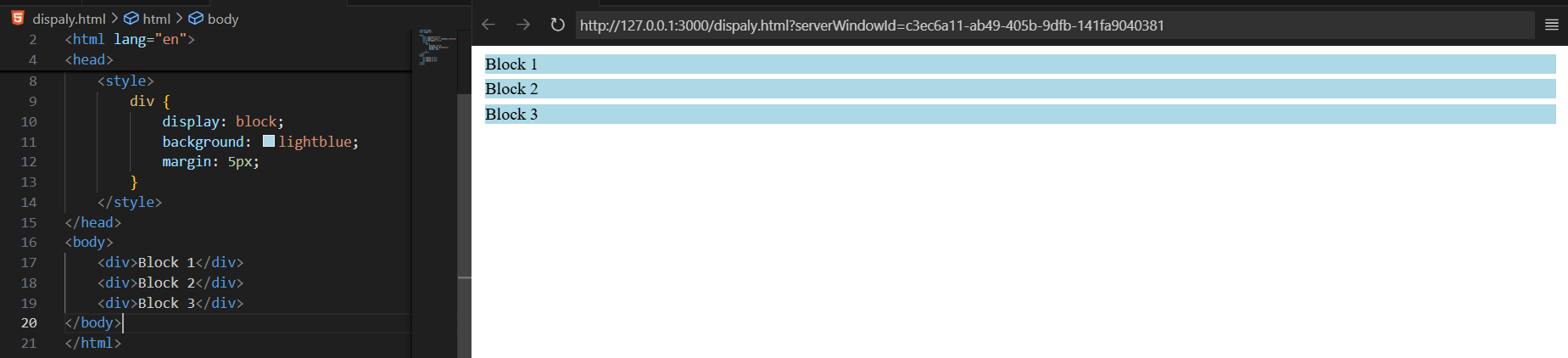


Whereas, in case we did “visibility:hidden”, then the space of that element be still there in page:

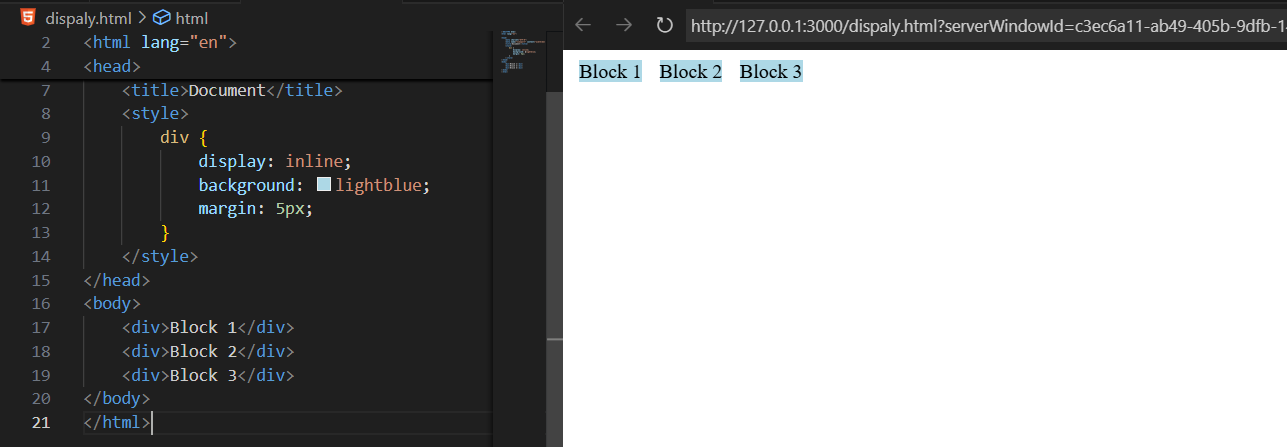


Now, some standard examples for future reference:

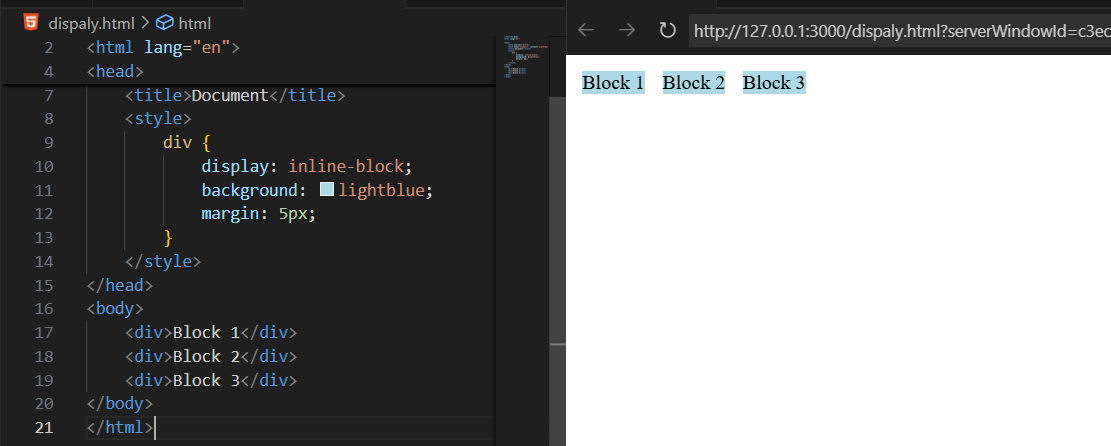
Example: display:block Each box starts on a new line and takes full width.



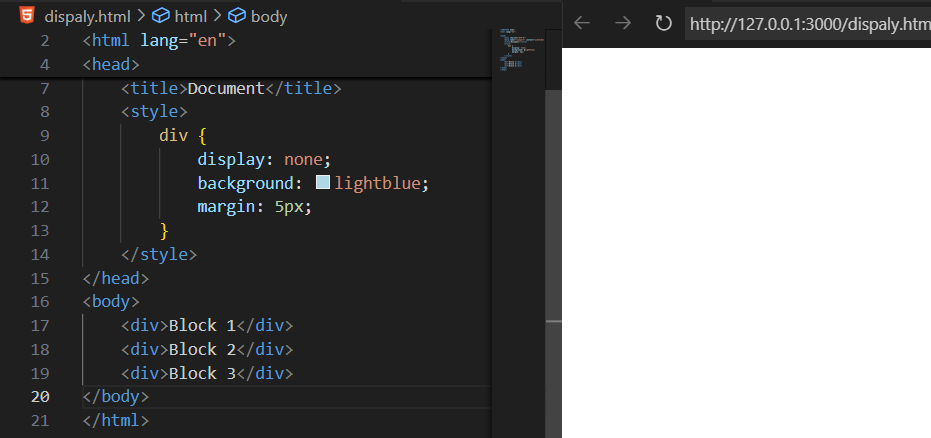
Example: display:inline All boxes appear in the same line (no line breaks).



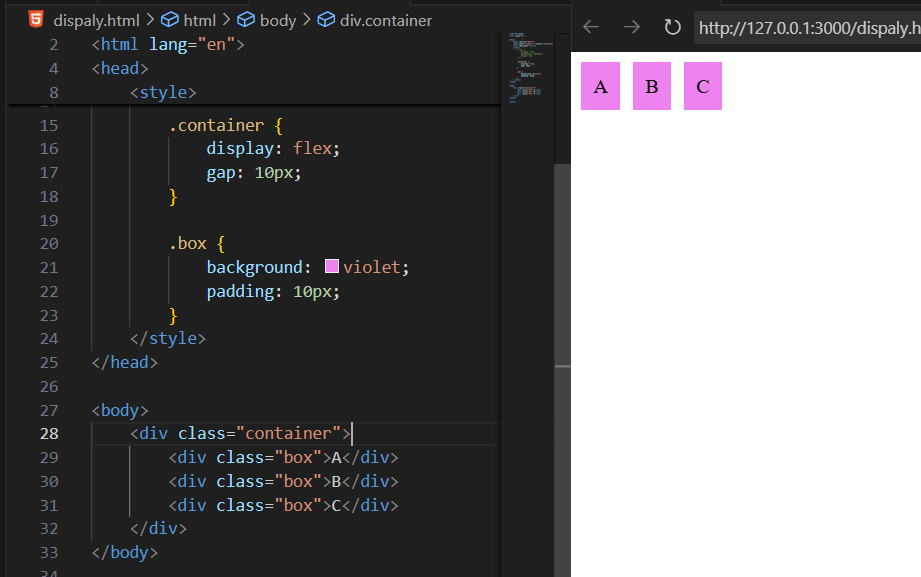
Example: display: inline-block Boxes appear in one line but can have width and height.



Example: display: none The <div> is completely hidden — not even space is left.



Example: display:flex Boxes align side by side and adjust flexibly.



Example: display:grid Boxes arranged in a 3-column grid layout.

