



# The position property

---

Property	Meaning	Values
position	Location of element on page	<b>static</b> : default position <b>relative</b> : offset from its normal static position <b>absolute</b> : at a particular offset within its containing element <b>fixed</b> : at a fixed location within the browser window
top, bottom, left, right	Offsets of element's edges	A size in px, pt, em or %

## `position: static;`

---

- ▶ **static** is the default position value for all elements.
- ▶ An element with **position: static;** is not positioned in any special way.
- ▶ A static element is said to be not positioned and an element with its position set to anything else is said to be positioned.

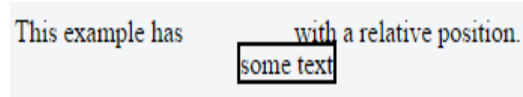


# position: relative;

- ▶ Set the location of an element to an offset from its normal static position.
- ▶ **relative** behaves the same as **static** unless you add some extra properties. Setting the **top**, **right**, **bottom**, and **left** properties of a relatively-positioned element will cause it to be adjusted away from its normal position. Other content will not be adjusted to fit into any gap left by the element. **relative** element stays in its place!

```
<p> This example has <span id="lifted">some text</span> with  
a relative position. </p>
```

```
#lifted {  
  position: relative;  
  left: 2em;  
  top: 1em;  
  border: 2px solid black;  
}
```



This example has some text with a relative position.

See example: [lesson3\\_examples/position-relative.html](lesson3_examples/position-relative.html)



# position: absolute;

---

- ▶ Elements that are positioned relatively are still considered to be in the normal flow of elements in the document.
- ▶ In contrast, an element that is positioned absolutely is taken out of the flow and thus takes up no space when placing other elements.
- ▶ The absolutely positioned element is positioned relative to *nearest **positioned** ancestor (non-static)*. If a positioned ancestor doesn't exist, the initial container is used.

```
#menubar{  
  width: 100px;  
  height: 50px;  
  position: absolute;  
  top: 20px;  
  left: 50px;  
}
```

See example:

[lesson3\\_examples/position-absolute.html](#)

[lesson3\\_examples/position-absoluteblock.html](#)



# position: fixed;

---

- ▶ Fixed positioning is similar to absolute positioning, with the exception that the element's containing block is the viewport. This is often used to create a floating element that stays in the same position even after scrolling the page.

```
#one {  
  position: fixed;  
  top: 80px;  
  left: 10px;  
}
```

See example:  
[lesson3\\_examples/position-fixed.html](#)

- ▶ A fixed element does not leave a gap in the page where it would normally have been located.
- ▶ A fixed element loses its space in the flow
- ▶ A fixed element does not move when you scroll (stays in place)



# Overlapping Elements

---

- ▶ When elements are positioned, they can overlap other elements.
- ▶ The z-index property specifies the stack order of an element.
- ▶ An element can have a positive or negative stack order.

# float

---

Float has several important uses for layout

- ▶ One is for wrapping text around images.
- ▶ Another is to position elements on the left or right or center, and possibly columns (being replaced by Flexbox and grid)

```
img {  
  float: right;  
  margin: 0 0 1em 1em;  
}
```

- When we float a `div` element it will comply to `width` and `height` properties rather than behaving like a block element (does not take whole width)
- A floating element is removed from normal document flow.
- Underlying text wraps around it as necessary.
- See example:  
[lesson3\\_examples/floatrightimage.html](lesson3_examples/floatrightimage.html)

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Phasellus imperdiet, nulla et dictum interdum, nisi lorem egestas odio, vitae scelerisque enim ligula venenatis dolor. Maecenas nisl est, ultrices nec congue eget, auctor vitae massa. Fusce luctus vestibulum augue ut aliquet. Mauris ante ligula, facilisis sed ornare eu, lobortis in odio. Praesent convallis urna a lacus



# Common float bug: missing width

---

- ▶ often floating block elements must have a width property value
  - ▶ if no width is specified, the floating element will size to fit its content. Large content may occupy 100% of the page width, so no content can wrap around it
  - ▶ See example: [lesson3\\_examples/floatingwithoutwidth.html](lesson3_examples/floatingwithoutwidth.html)



# The clear property

(Stop the float I want to get off!)

```
img.hoveringicon { float: left; margin-right: 1em; }  
h2 { clear: left; background-color: yellow; }  
p { background-color: fuchsia; }
```



Starhome Sprinter is a Flash animated Internet cartoon. It mixes surreal humour with references to 1980s and 1990s pop culture, notably video games, classic television and popular music.

[My Starhome Sprinter Fan Site](#)

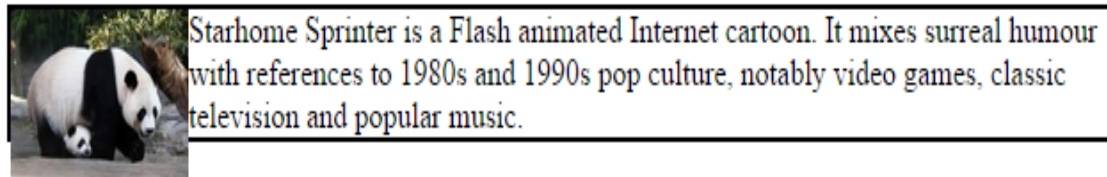
See example: [lesson3\\_examples/clear.html](lesson3_examples/clear.html)

Property	Meaning	Values
clear	Whether to move this element below any prior floating elements in the document	left, right, both, none(default)

# Common error: container too short

---

- ▶ If you place a tall *floating element* inside a block element without much other content, the floating element may hang down past the bottom edge of the block element that contains it.



- ▶ See example:  
[lesson3\\_examples/commonerrorcontenttooshort.html](http://lesson3_examples/commonerrorcontenttooshort.html)

# The overflow property

---

overflow: hidden

## 2 scenarios for use of overflow

- ▶ *Floating elements* that are too tall for containing block
  - ▶ Previous slide
  - ▶ `lesson3_examples/overflow.html`
- ▶ Blocks of fixed height that have more content than their space allows
  - ▶ [https://www.w3schools.com/css/tryit.asp?filename=trycss\\_overflow\\_visible](https://www.w3schools.com/css/tryit.asp?filename=trycss_overflow_visible)

Property	Meaning	Values
overflow	Action to take if element's content is larger than the element itself	visible(default), hidden, scroll, auto

# Alignment vs. float vs. position

---

1. if possible, lay out an element by aligning its content
  - ▶ horizontal alignment: text-align
    - ▶ set this on a block element; it aligns the content (text and inline elements) within it (not the block element itself)
    - ▶ E.g., see lesson3\_examples/textalign.html example
    - ▶ Might need to make block elements have display: inline
  - ▶ vertical alignment: vertical-align
    - ▶ set this on an inline element, and it aligns it vertically within its containing element
2. if alignment won't work, try floating the element
3. if floating won't work, try positioning the element
  - ▶ absolute/fixed positioning are a last resort and should not be overused
4. OR, use flexbox or grid instead of float and positioning