HNDIT1022 – Web Design

Week 6: Understanding the CSS Box Model and Positioning

Subtopics

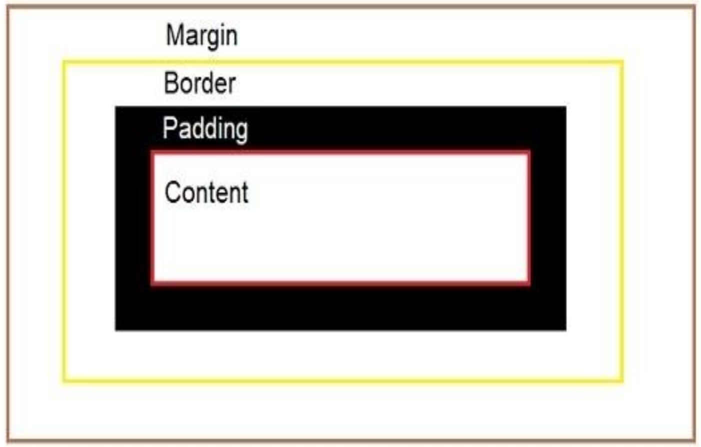
- Conceptualize CSS box model
- How to position elements
- Use CSS to do more with lists, text and navigations

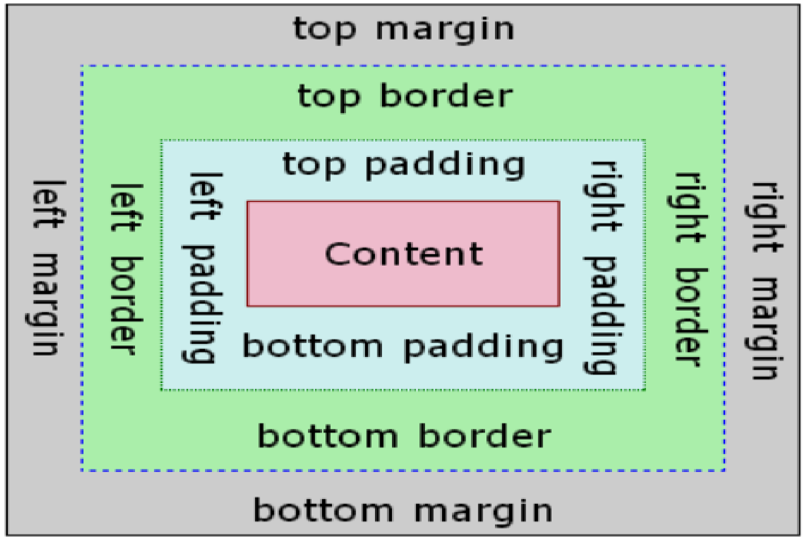
What is CSS Box Model?

All HTML elements can be considered as boxes. In CSS, the term “box model” is used when talking about design and layout.

The CSS box model is essentially a box that wraps around every HTML element. It consists of margins, borders, padding, and the actual content.

The image below illustrates the box model:





Contd..

- The CSS Box Model defines the layout of HTML block elements such as divisions (<div>s) and paragraphs (<p>s). It does not apply to inline elements (<span>s, anchors (<a>s), and a bunch of others).

- Every block element is laid out (by the browser) as a rectangular box with four parts (boxes). The innermost part (box) is the content (or content box) which is surrounded by padding (or padding box). The border (or border box) encloses the padding; the outermost part is the margin (margin box).
- The padding is used to specify the amount of spacing surrounding the content within the border box, while the margin is used to specify the spacing separating the element from its surroundings.
- For screen display, padding and margin are normally specified in pixels.

Contd..

Explanation of the different parts:

Content - The content of the box, where text and images appear
Padding - Clears an area around the content. The padding is transparent
Border - A border that goes around the padding and content
Margin - Clears an area outside the border. The margin is transparent

Width and Height of an Element

When you set the width and height properties of an element with CSS, you just set the width and height of the **content area**.
To calculate the full size of an element, you must also add padding, borders and margins.

Contd..

The total width of an element should be calculated like this:

Total element width = width + left padding + right padding + left border + right border + left margin + right margin

The total height of an element should be calculated like this:

Total element height = height + top padding + bottom padding + top border + bottom border + top margin + bottom margin

By default (if no width is set), a block element takes up the whole width of the browser window and is resized when the browser window is resized. The element's height (if no height is set) adjusts to accommodate its content (the user may have to scroll).

Contd..

Example :

This <div> element will have a total width of 350px:

```
div {
  width: 300px;
  padding: 20px;
  border: 5px solid gray;
  margin: 0;
}
```

Here is the calculation:

300px (width)

+ 40px (left + right padding)

+ 10px (left + right border)

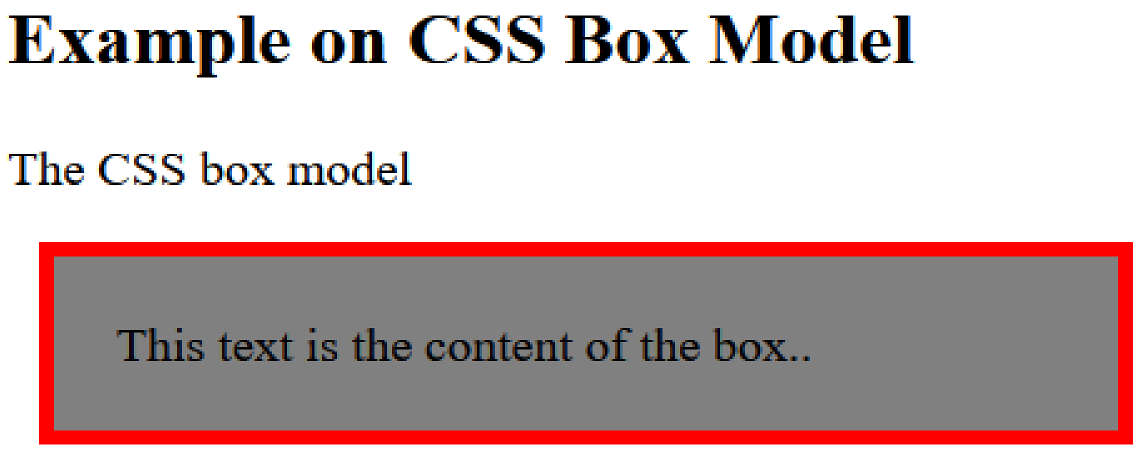
+ 0px (left + right margin)

= 350px

Exercise 1

```
<!DOCTYPE html>
<html>
<head>
<style>
div {
  background-color: grey;
  width: 300px;
  border: 5px solid red;
  padding: 20px;
  margin: 10px;
}
</style>
</head>
<body>
<h2>Example on CSS Box
Model</h2>
<p>The CSS box model</p>
<div>This text is the content of
the box..</div>
</body>
</html>
```

Output



Example on CSS Box Model

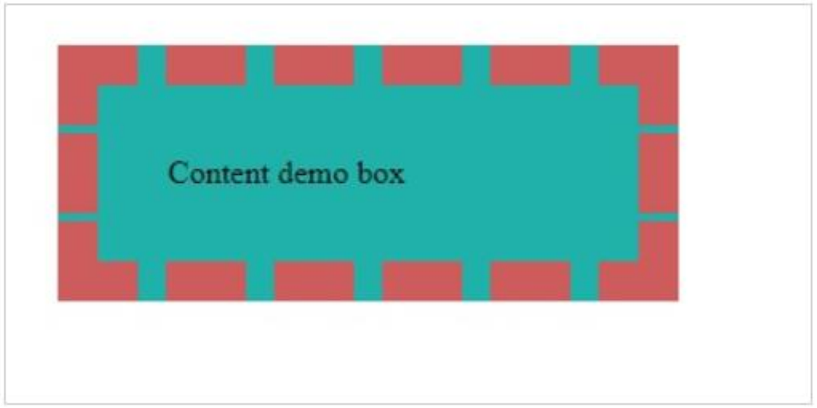
The CSS box model

This text is the content of the box..

Exercise 2

```
<!DOCTYPE html>
<html>
<head>
<style>
div {
  margin: 25px;
  padding: 35px;
  width: 200px;
  background-color:
lightseagreen;
  border: 20px dashed
indianred;
}
</style>
</head>
```

```
<body>
<div>
Content box<br/>
</div>
</body>
</html>
```



Content demo box

CSS Layout - The position Property

The position property specifies the type of positioning method used for an element

There are five different position values:

- static
- relative
- fixed
- absolute
- sticky

Elements are then positioned using the top, bottom, left, and right properties. However, these properties will not work unless the position property is set first. They also work differently depending on the position value.

Position : static

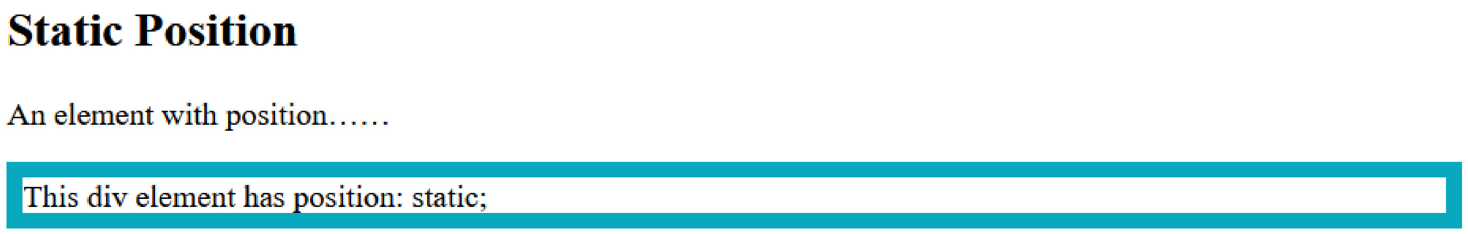
HTML elements are positioned static by default.

Static positioned elements are not affected by the top, bottom, left, and right properties.

An element with position static is not positioned in any special way, it is always positioned according to the normal flow of the page

Exercise 3

```
<!DOCTYPE html>
<html>
<head>
<style>
div.static {
  position: static;
  border: 8px solid #08a9bf;
}
</style>
</head>
<body>
<h2>Static Position</h2>
<p>An element with
position……</p>
<div class="static">
This div element has position:
static;
</div>
</body>
</html>
```



Static Position

An element with position……

This div element has position: static;

Position : relative

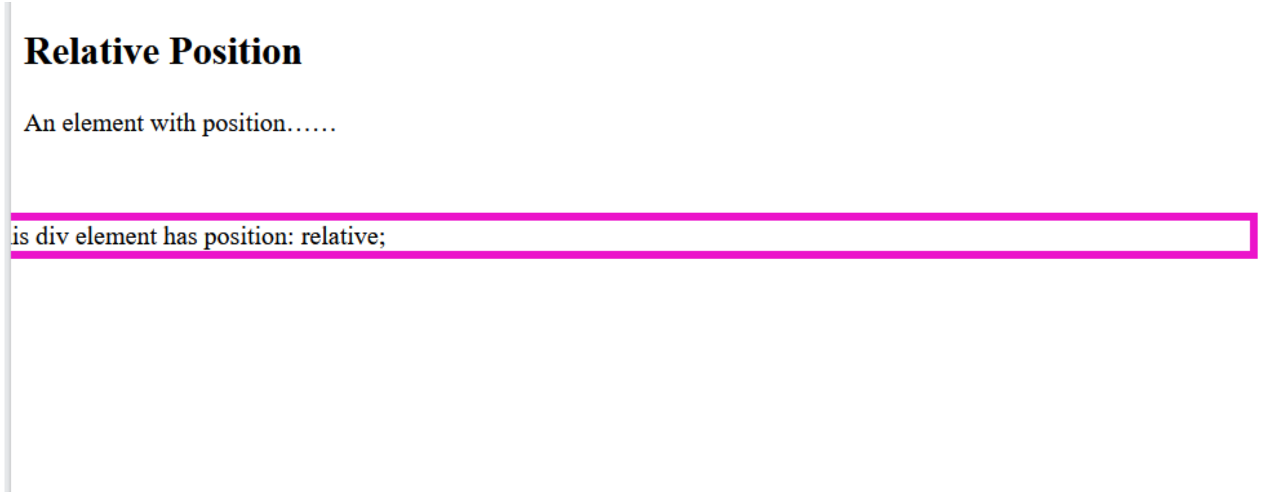
An element with position relative is positioned relative to its normal position.

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.

Exercise 4

```
<!DOCTYPE html>
<html>
<head>
<style>
div.relative {
  position: relative;
 top: 30px;
 right: 50px;
  border: 5px solid #eb14ca;
}
</style>
</head>
```

```
<body>
<h2>Relative Position</h2>
<p>An element with
position……</p>
<div class="relative">
This div element has position:
relative;
</div>
</body>
</html>
```



Relative Position

An element with position……

is div element has position: relative;

Position : fixed

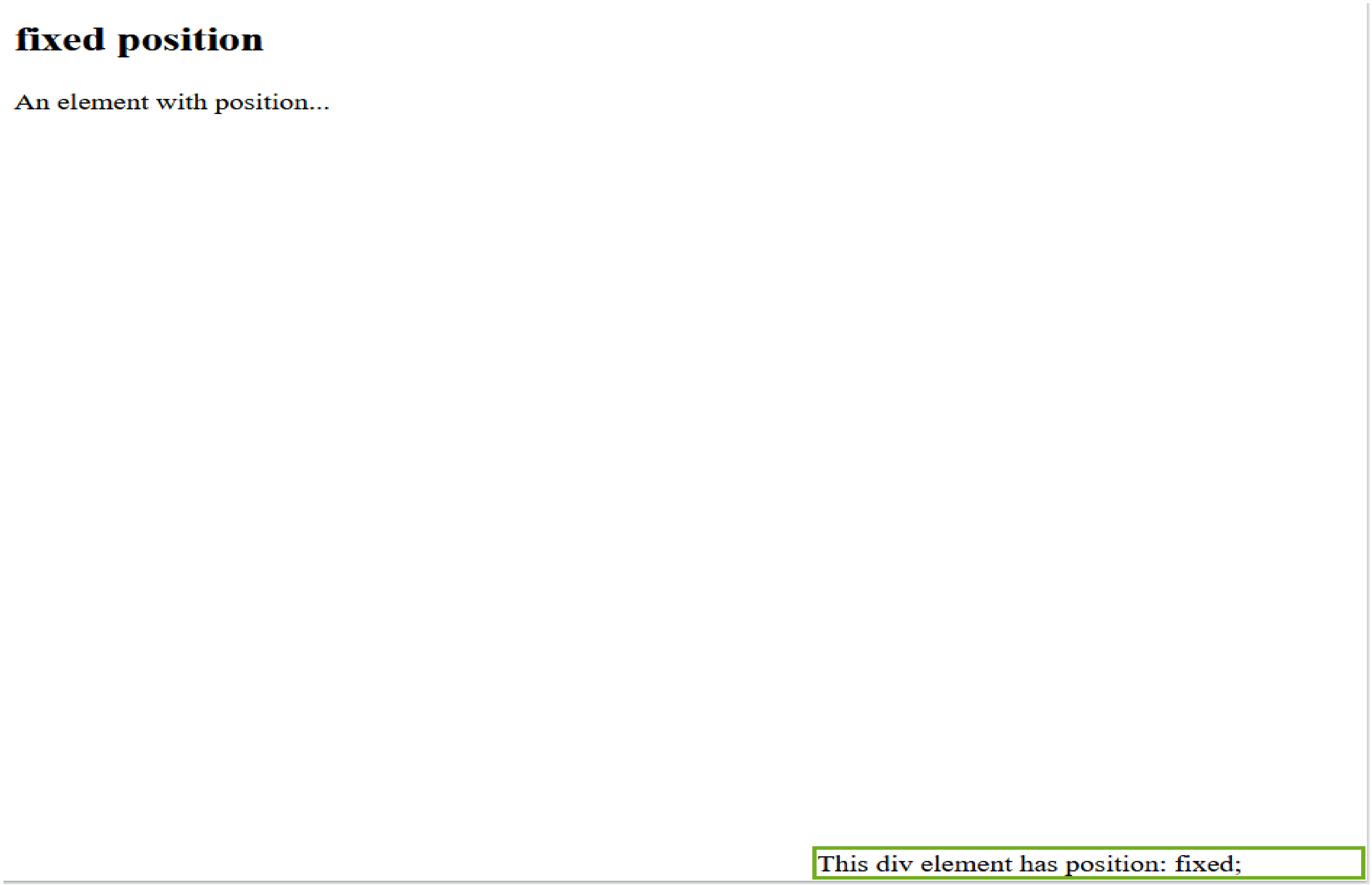
An element with position fixed is positioned relative to the viewport, which means it always stays in the same place even if the page is scrolled. The top, right, bottom, and left properties are used to position the element.

A fixed element does not leave a gap in the page where it would normally have been located.

Exercise 5

```
<!DOCTYPE html>
<html>
<head>
<style>
div.fixed {
  position: fixed;
  bottom: 0;
  right: 0;
  width: 300px;
  border: 3px solid #73AD21;
}
</style>
</head>
<body>
<h2>fixed position</h2>
<p>An element with
position...</p>
<div class="fixed">
This div element has position:
fixed;
</div></body>
</html>
```

Contd…



fixed position

An element with position...

This div element has position: fixed;

Position : absolute

An element with position: absolute; is positioned relative to the nearest positioned ancestor (instead of positioned relative to the viewport, like fixed).

However; if an absolute positioned element has no positioned ancestors, it uses the document body, and moves along with page scrolling.

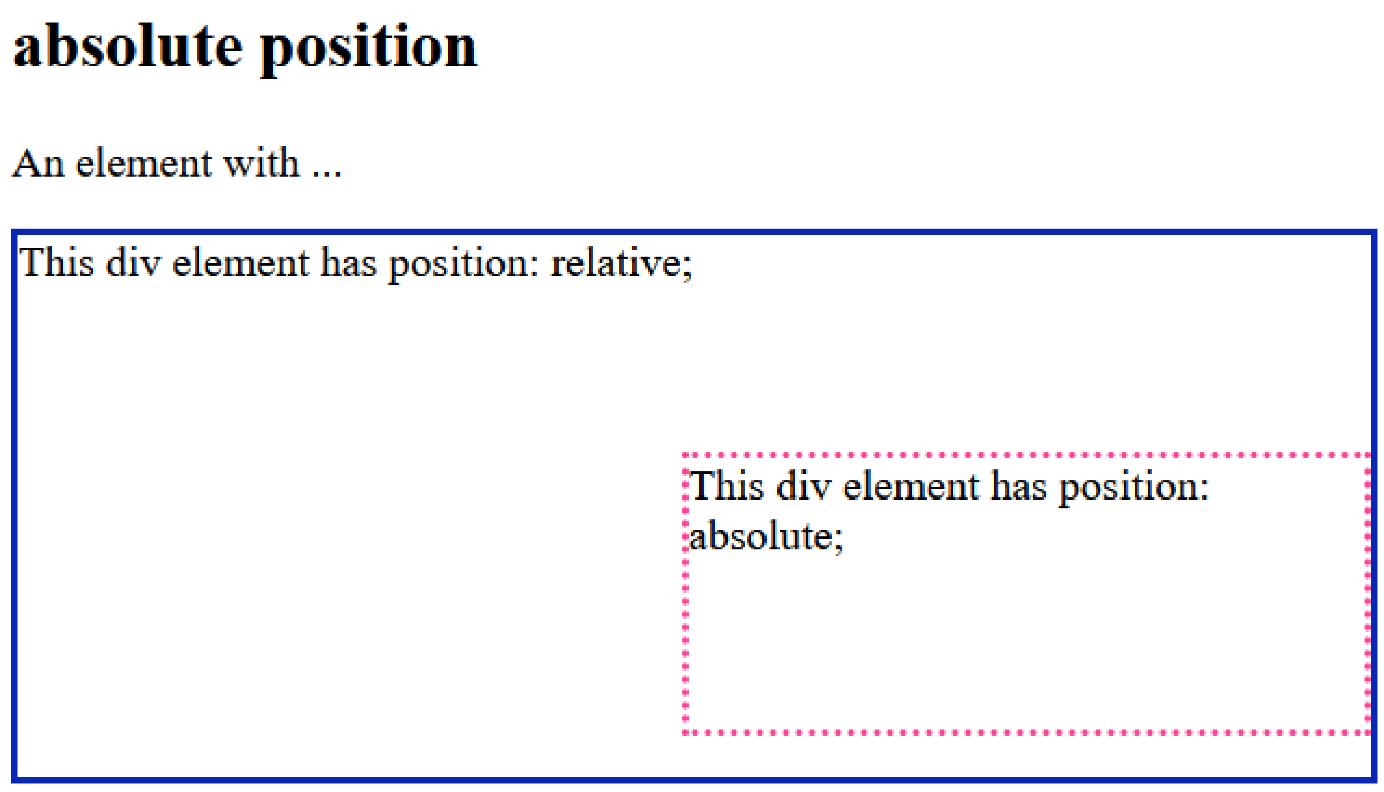
Note: Absolute positioned elements are removed from the normal flow, and can overlap elements.

Exercise 6

```
<!DOCTYPE html>
<html>
<head>
<style>
div.relative {
  position: relative;
  width: 500px;
  height: 200px;
  border: 3px solid #0a27b3;
}
div.absolute {
  position: absolute;
  top: 80px;
  right: 0;
  width: 250px;
  height: 100px;
  border: 3px dotted #f83e90;
}
</style>
</head>
```

```
<body>

<h2>absolute position</h2>

<p>An element with ...</p>

<div class="relative">This div element
has position: relative;
  <div class="absolute">This div
element has position: absolute;</div>
</div>

</body>
</html>
```

Contd…



absolute position

An element with ...

This div element has position: relative;

This div element has position: absolute;

Position : sticky

An element with position sticky is positioned based on the user's scroll position.

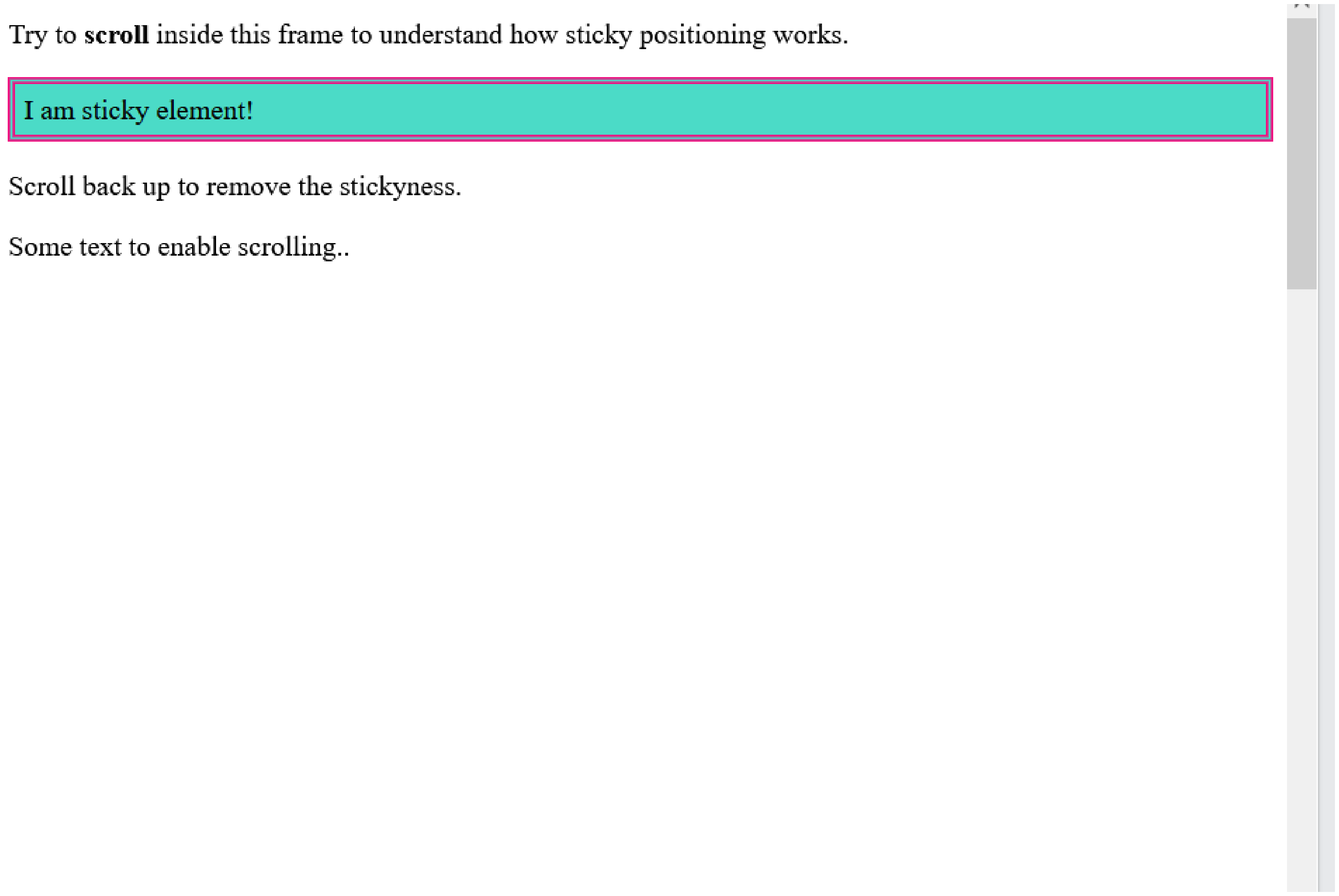
A sticky element toggles between relative and fixed, depending on the scroll position. It is positioned relative until a given offset position is met in the viewport - then it "sticks" in place (like position : fixed).

Note*: Internet Explorer does not support sticky positioning. Safari requires a -webkit- prefix (see example below). You must also specify at least one of top, right, bottom or left for sticky positioning to work.*

Exercise 7

```
<!DOCTYPE html>
<html>
<head>
<style>
div.sticky {
  position: -webkit-sticky;
  position: sticky;
  top: 0;
  padding: 5px;
  background-color: #4bdbc7;
  border: 4px double #f10d7f;
}
</style>
</head>
<body>

<p>Try to <b>scroll</b> inside this
frame to understand how sticky
positioning works.</p>

<div class="sticky">I am sticky
element!</div>

<div style="padding-bottom:2000px">
  <p></p>
  <p>Scroll back up to remove the
stickyness.</p>
  <p>Some text to enable
scrolling..</p>

</div>

</body>
</html>
```

Contd…


Try to **scroll** inside this frame to understand how sticky positioning works.

I am sticky element!

Scroll back up to remove the stickyness.

Some text to enable scrolling..