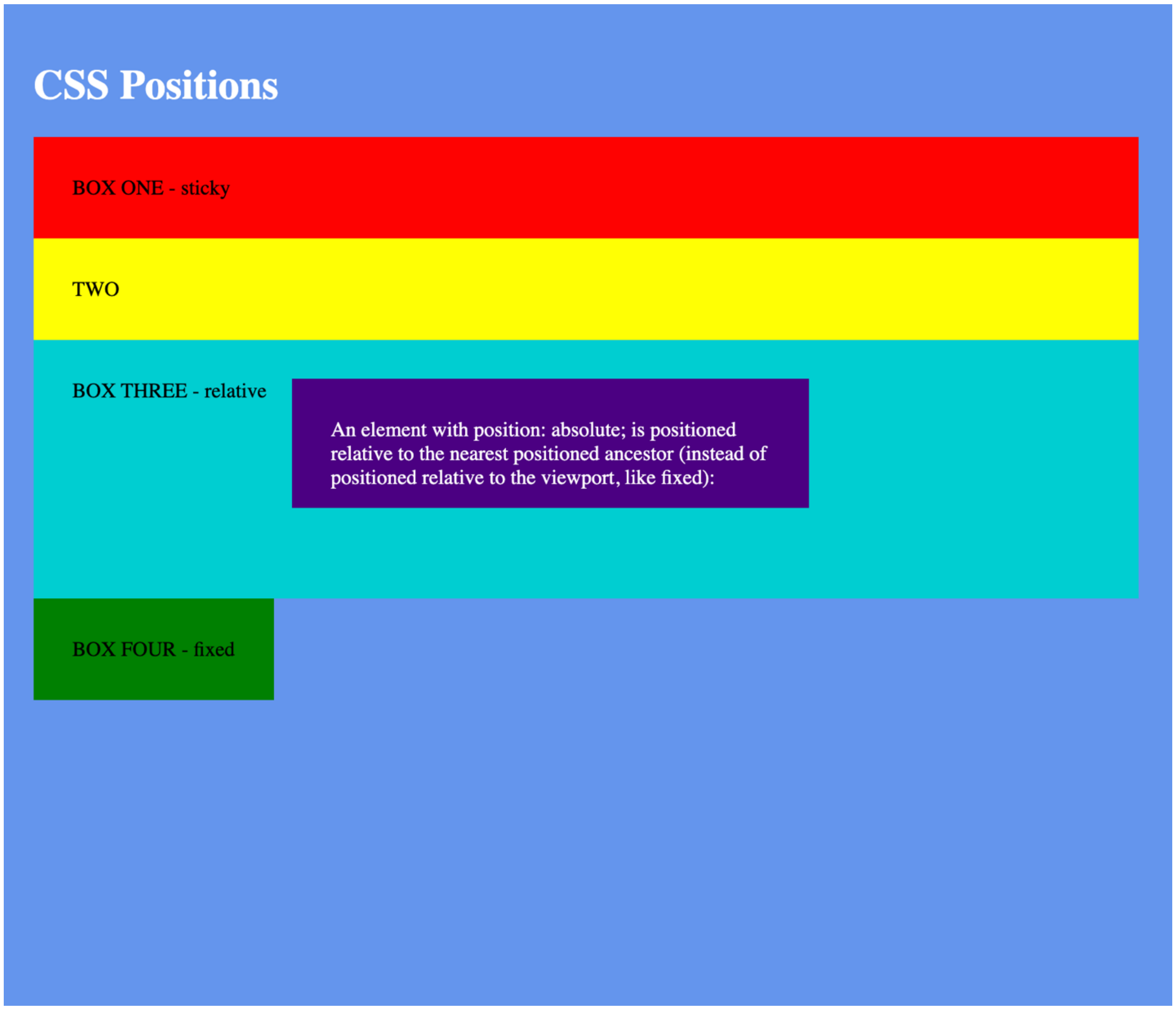
CSS Basic Position Code Snippet

```html
<!DOCTYPE html>
<html>

<head>
  <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
  <!-- The viewport is the user's visible area of a web page. -->
  <!-- The width=device-width part sets the width of the page to follow the screen-width of the device (which will vary depending on the device). -->
  <!-- The initial-scale=1.0 part sets the initial zoom level when the page is first loaded by the browser. -->
  <meta name="viewport" content="width=device-width, initial-scale=1" />
  <title>CSS Basic Position</title>
  <!-- reference: https://www.w3schools.com/css/css_positioning.asp -->
  <style>
    /* Internal CSS */
    * {
      /* set how the total width and height of an element is calculated
           border-box tells the browser to account for any border and padding in the values you specify for an element's width and height.
           If you set an element's width to 100 pixels, that 100 pixels will include any border or padding you added,
           and the content box will shrink to absorb that extra width.
           Reference: https://developer.mozilla.org/en-US/docs/Web/CSS/box-sizing */
      box-sizing: border-box;
    }

    .parent-box {
      width: 100%;
      height: 1500px;
      padding: 30px;
      background: cornflowerblue;
    }

    .parent-box>h1 {
      color: white;
    }

    .child-box {
      padding: 30px;
    }

    .background-red {
      background-color: red;
    }

    .background-yellow {
      background-color: yellow;
    }

    .background-green {
      background-color: green;
    }
    .background-darksalmon {
      background-color: darkturquoise;
    }

    .background-deeppink {
      background-color: indigo;
    }

    .child-one {
      /* sticky has 2 stages:
           stage 1 - relative when top/left/bottom/right measures are not met yet
           stage 2 - absolute when top/left/bottom/right measures are met */
      position: sticky;
      top: 0;
    }

    .child-three {
      position: relative;
      height: 200px;
    }

    .child-three-child {
      position: absolute;
      top: 30px;
      /* top takes precedent over bottom */
      left: 200px;
      /* left takes precedent over right */
      right: 100px;
      bottom: 10px;
      width: 400px;
      height: 100px;
      color: white;
    }

    .child-four {
      /* 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.
        */
      position: fixed;
      top: 475px;
      left: 38px;
    }
  </style>
</head>

<body>
  <div class="parent-box">
    <h1>CSS Positions</h1>
    <div class="child-one child-box background-red">BOX ONE - sticky</div>
    <div class="child-box background-yellow">TWO</div>
    <div class="child-three child-box background-darksalmon">BOX THREE - relative
      <!-- parent has darksalmon background color -->
```

```html
      <!-- child has deeppink background color that cascaded the parent's color -->
      <!-- the child also cascaded the parent's box size and position -->
      <div class="child-three-child child-box background-deeppink">
        An element with position: absolute; is positioned relative to the nearest positioned ancestor
(instead of
        positioned
        relative to the viewport, like fixed):
      </div>
    </div>
    <div class="child-four child-box background-green">BOX FOUR - fixed</div>
  </div>

</body>

</html>
```

CSS Basic Position Output



CSS Positions

BOX ONE - sticky

TWO

BOX THREE - relative

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

BOX FOUR - fixed