

CGS 3175: Internet Applications Fall 2009

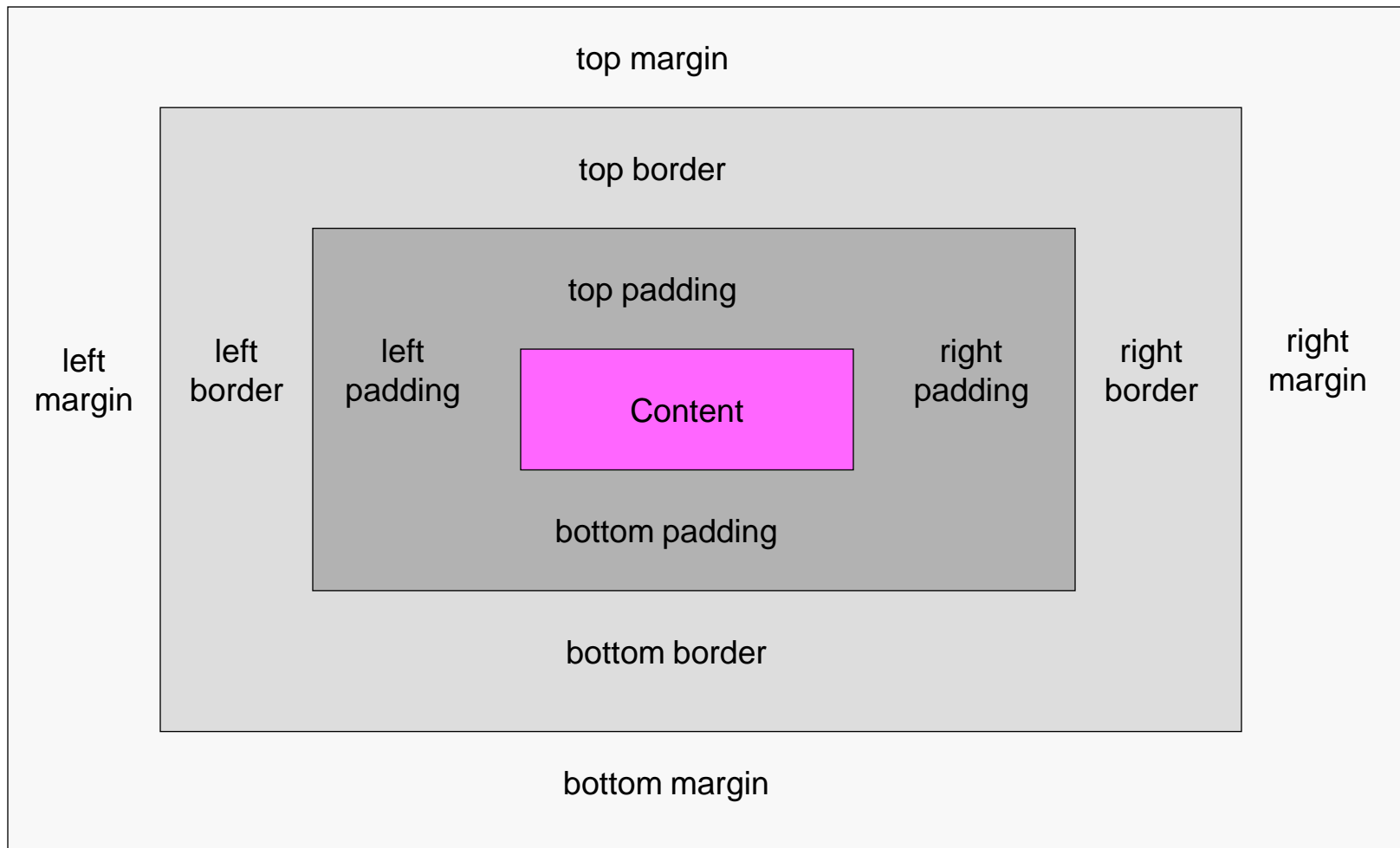
Cascading Style Sheets – Page Layout - Part 2

Instructor : Dr. Mark Llewellyn
markl@cs.ucf.edu
HEC 236, 407-823-2790
<http://www.cs.ucf.edu/courses/cgs3175/fall2009>

School of Electrical Engineering and Computer Science
University of Central Florida



The CSS Box Model



An Aside – CSS Shorthand Notation

- It can get quite tedious writing a separate style for each of the four sides of an element, whether you are specifying margins, padding, or borders. CSS provides several different shorthand ways to specify these properties within a single declaration.
- In any shorthand notation the order is always top, right, bottom, left.
- So instead of writing:

```
{margin-top:5px; margin-right:10px; margin-bottom:8px; margin-left:4px}
```

You can write: `{margin: 5px 10px 8px 4px;}`



An Aside – CSS Shorthand Notation

- Note that there is just a space between each of the values, no delimiters such as a comma are used.
- You also do not need to specify all four values when using the shorthand notation. If you do not provide all four values, the opposite side's value is used as the missing value.
- For example: `{margin: 12px 10px 6px;}` will set the missing value for the left margin as 10px which is the value that is specified for the right margin.
- Similarly: `{margin: 12px 10px;}` will set the missing bottom margin to 12px and the missing left margin to 10px.



An Aside – CSS Shorthand Notation

- If only a single value is specified, such as `{margin: 12px; }` then all four sides are set to this single value.
- Note when using the shorthand notation, it is not possible to set only the bottom and left margins without providing some values for the top and right, even if those values are both zero. In cases such as this, you can write 0 without supplying a value type, such as:

```
{margin: 0 0 4px 8px; }
```

- The same shorthand rules apply to padding and border, as well as margin.



The CSS Box Model

- You can adjust three different aspects of the box with CSS: **margin**, **border**, and **padding**.
- For the margin you can set the distance between this box and adjacent elements in the markup.
- For the border you can set the thickness, style, and color.
- For the padding you can set the distance of the box's content is to be separated from its border.



The CSS Box Model

- A simple way to think about these properties is that margins push outward from the border and padding pushes inward from the border.
- Since every box has four sides, properties associated with margins, border, and padding each have four settings: `top`, `right`, `bottom`, and `left`.
- We'll now look more closely at the properties associated with a box's margin, border, and padding and give lots of examples illustrating how to manipulate these properties to achieve the results you want.



The Box Border

- The box border has three associated properties:
 - **Width**. This includes thin, medium, thick, or any unit (ems, px, %, ...).
 - **Style**. This includes none, hidden, dotted, dashed, solid, double, groove, ridge, inset, and outset.
 - **Color**. Any color value can be used.
- The shorthand notation for styling all four borders the same, width, style, and color is:

```
.borderclass {border: 4px solid green; }
```




```

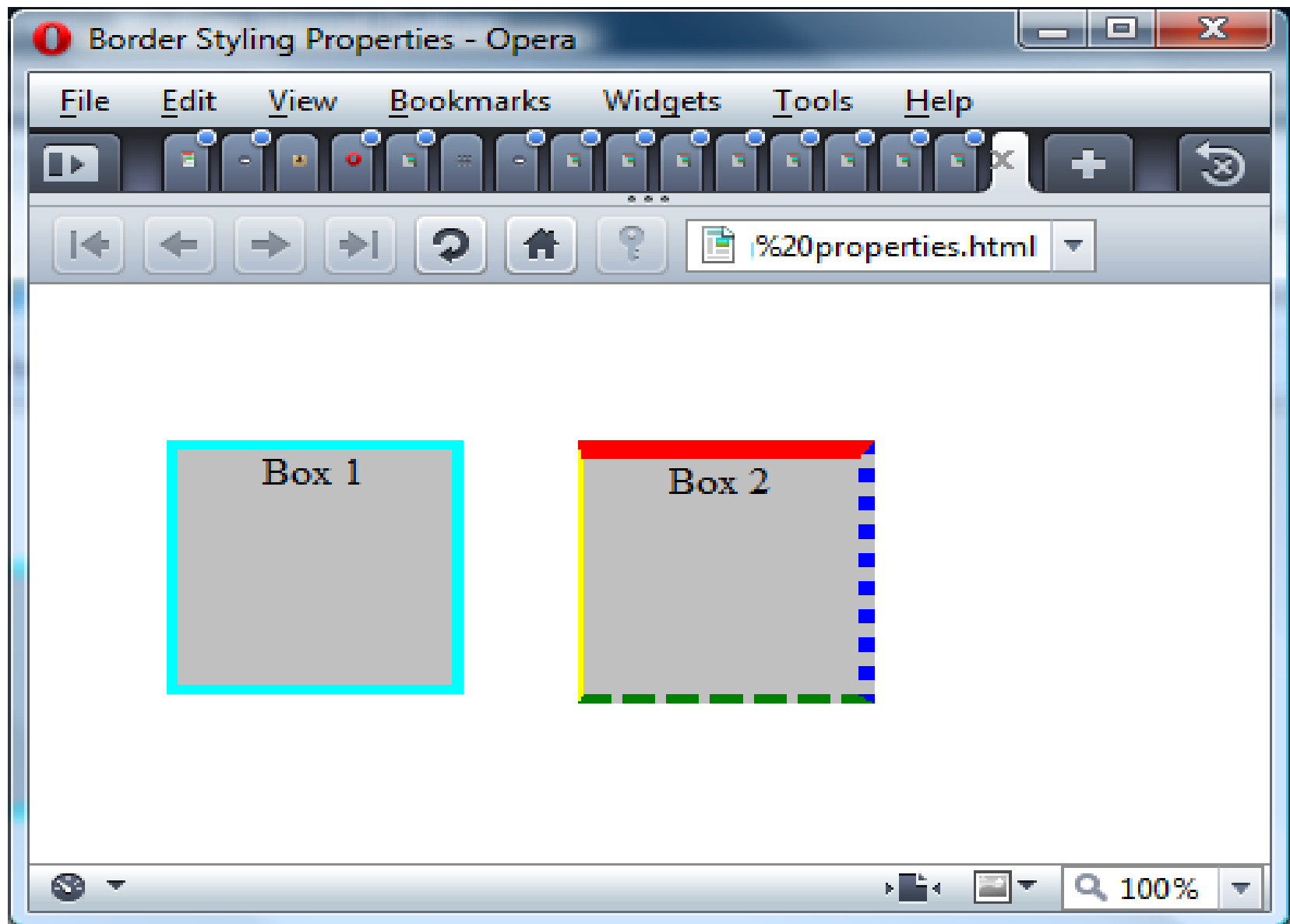
1  <?xml version="1.0" encoding="UTF-8" standalone="no"?>
2  <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
3    "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
4  <html xmlns="http://www.w3.org/1999/xhtml">
5    <head>
6      <title>Border Styling Properties</title>
7      <style type="text/css">
8        <!--
9          p{text-align:center;}
10         #box1 {border: 4px solid aqua; position:absolute; top:50px; left:50px; width:100px; height:100px; background-color:silver;}
11         #box2 {position: absolute; border-top:8px solid red; border-right:6px dotted blue; border-bottom:4px dashed green;
12               border-left:2px double yellow; top:50px; left:200px; width:100px; height:100px; background-color:silver;}
13         -->
14      </style>
15    </head>
16    <body>
17      <p id="box1">Box 1</p>
18      <p id="box2">Box 2</p>
19    </body>
20  </html>

```

#box1 uses the shorthand notation to style all four sides using the same styles

#box2 styles each side of the box differently so four different border properties are set.





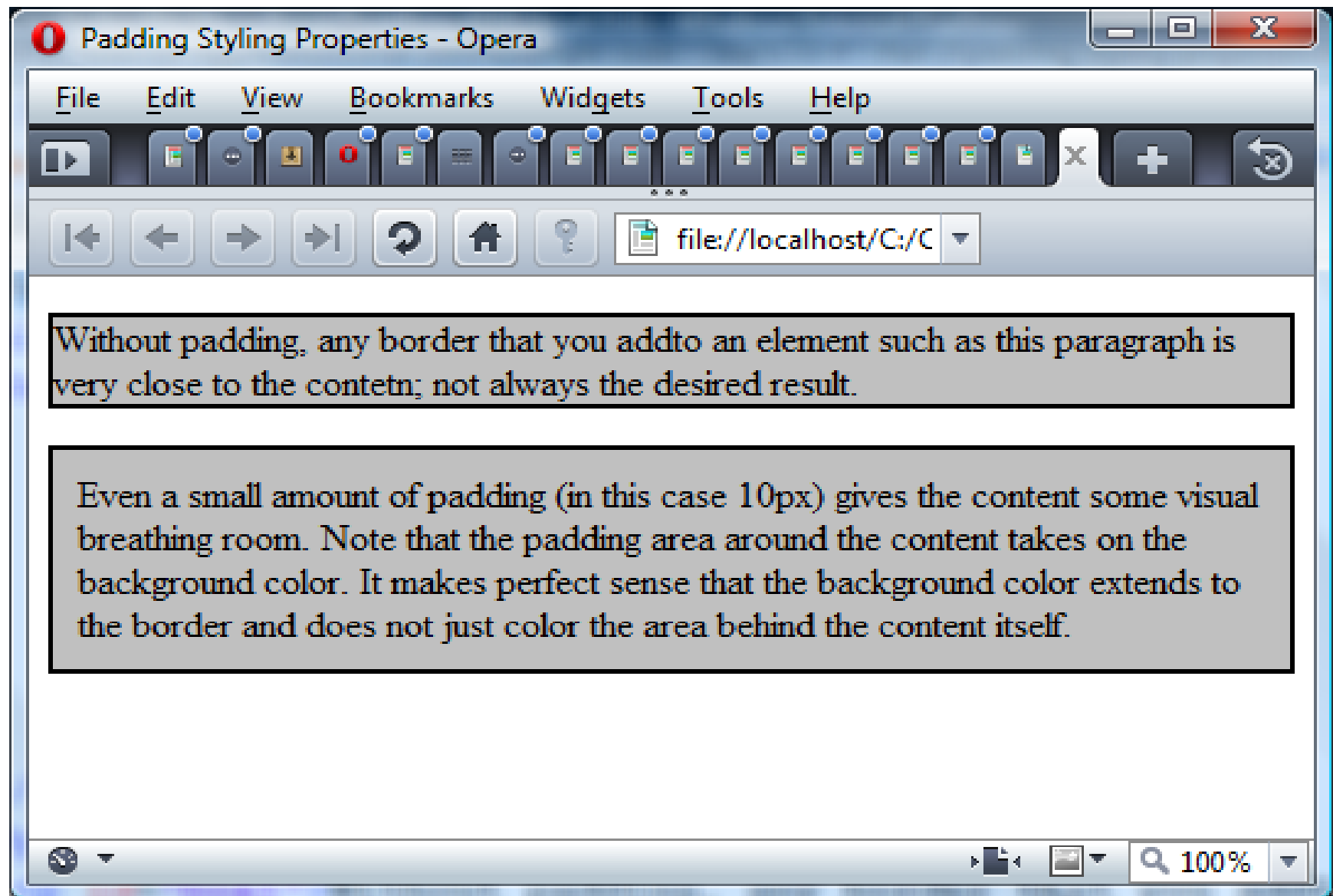
The Box Padding

- Padding adds space between the box's content and the border of the box.
- As part of the inside of the box, it takes on the color of the box's background.
- The markup on the next page illustrates two paragraphs, one with and one without padding.



```
3  "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
4  <html xmlns="http://www.w3.org/1999/xhtml">
5  <head>
6      <title>Padding Styling Properties</title>
7      <style type="text/css">
8      <!--
9          #box1 {border: 2px solid black; background-color:silver; padding: 0px;}
10         #box2 {border: 2px solid black; background-color:silver; padding: 10px;}
11      -->
12      </style>
13  </head>
14  <body>
15      <p id="box1">Without padding, any border that you add to an element such as
16          this paragraph is very close to the content; not always the desired
17          result.</p>
18      <p id="box2">Even a small amount of padding (in this case 10px) gives the content
19          room. Note that the padding area around the content takes on the background
20          color. It makes perfect sense that the background color extends to the border
21          and does not just color the area behind the content itself.</p>
22  </body>
```

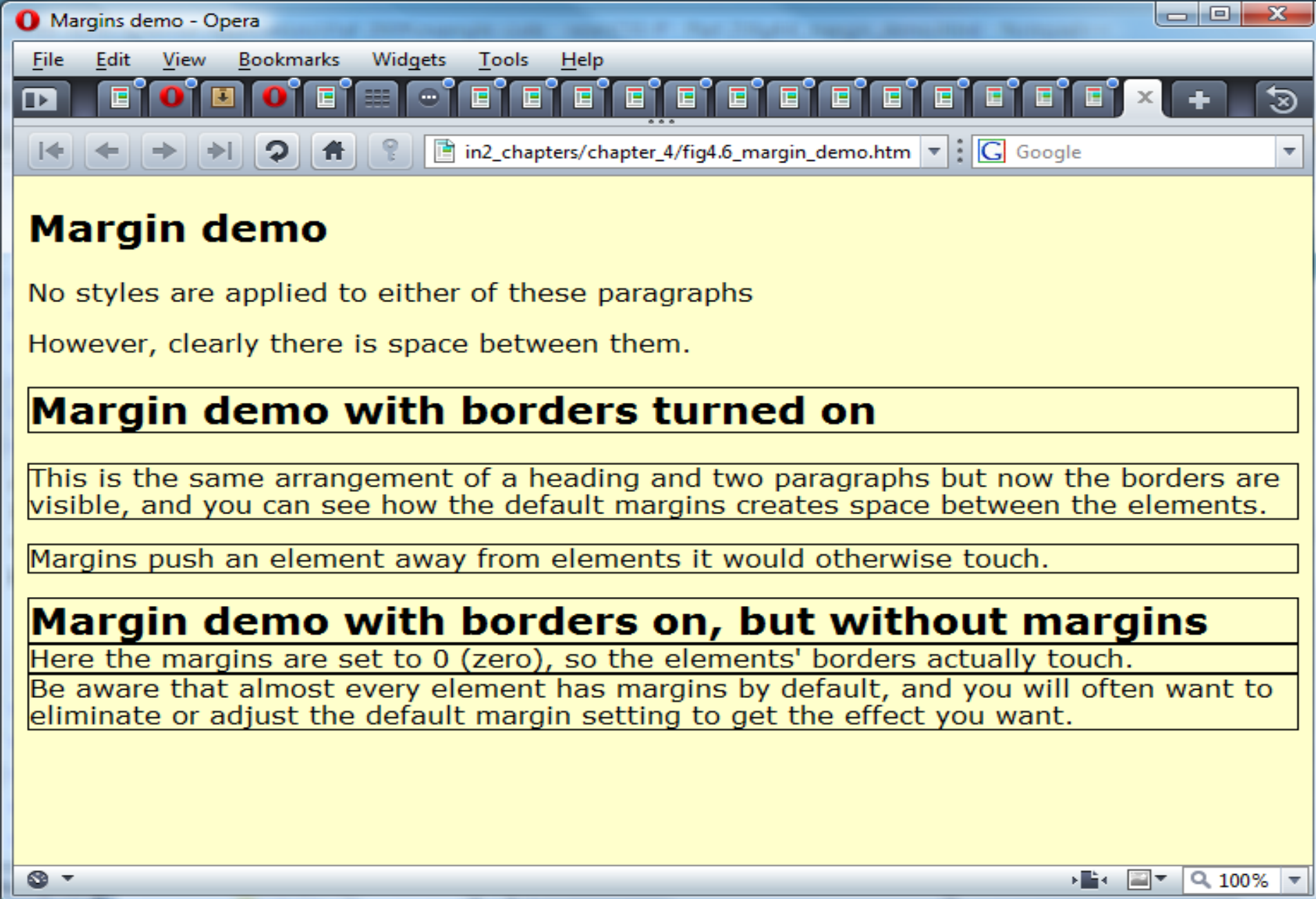




The Box Margins

- Margins are slightly more complex than borders and padding.
- Most block level elements (paragraphs, headings, lists, etc.) have default margins (see CSS-P – Part 1 page 10).
- Consider the example shown on the next page which illustrates a heading and two paragraphs.
 - The first case shows the default case.
 - The second case shows the borders and backgrounds turned “on” to illustrate how the margins create space between the elements.
 - The third case illustrates what happens if the margins are set to zero. The elements then touch one another.





The Box Margins

- It is typically a good practice in your CSS to place the following declaration at the top of a style sheet:

```
* {margin:0; padding:0; }
```

- This will set the default margins and padding of all elements to 0 and that way you won't get confused as to which margins and padding are being set by the browser and which you are setting.
- Now you can add them back to just the elements that you want to have margins as you style the page.
- As we'll see later, different browsers apply default padding and margins differently to element sets such as forms and lists, and by “neutralizing” the default settings in this manner and then adding your own, will result in a more consistent cross-browser effect.



The Box Margins

- Often you will mix units when you set margins for text elements such as paragraphs.
- For example, the left and right margins of a paragraph might be set in pixels so that the text remains a fixed distance from a navigation sidebar, but you might want to set the top and bottom margins in ems so that the vertical spacing between paragraphs is relative to the size of the paragraphs' text. So you might do something like this:

```
p {font-size:1em; margin:0.75em 30px; }
```



The Box Margins

```
p {font-size:1em; margin:0.75em 30px; }
```

In this case, the space between paragraphs is always three-quarters of the height of the text: if you increase the overall type size in the body element, not only will paragraphs' text get bigger, but the space between the paragraphs also increased proportionally. The left and right margins, set in pixels will remain unchanged by the redefinition of the body element type size.

- This will become more important when we begin looking at overall page layouts.



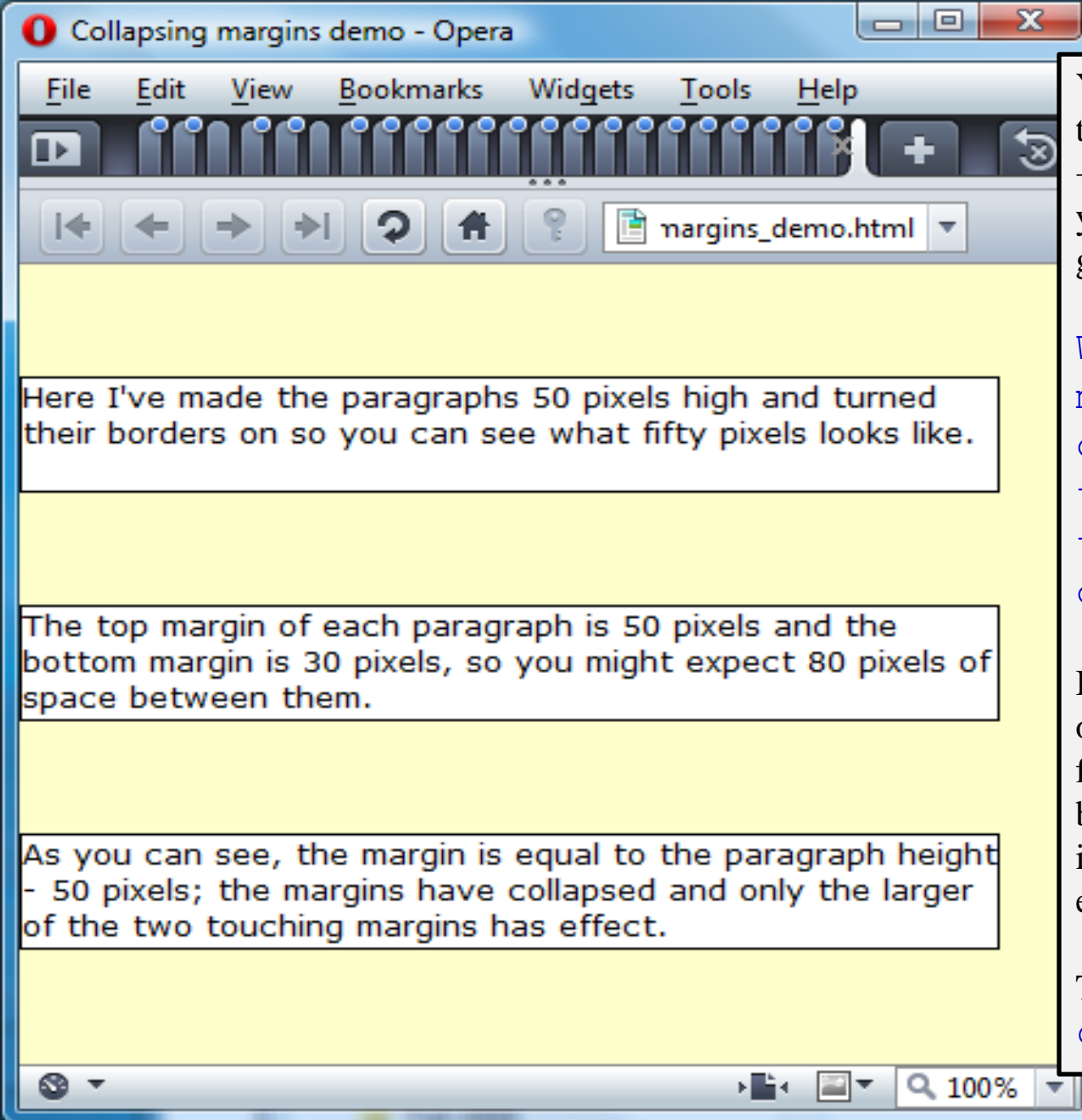
The Box Margins

- Vertical margins collapse. Horizontal margins do not.
- Consider the following scenario: we have three paragraphs of text, one after the other in normal flow, and each is styled using the following CSS rule:

```
p {width:400px; height:50px; border:1px solid black;  
margin-top:50px; margin-bottom:30px;}
```

- How would you expect these paragraphs to appear? How much space (in terms of pixels) will appear between the paragraphs? Will it be 30px, 50px or 80px?





You might reasonably assume that there would be 80 pixels (50 + 30) between the paragraphs, but you would be wrong as the actual gap is only 50 pixels.

When top and bottom margins meet, they overlap until one of the margins touches the border of the other element.

In this case, the larger top margin of the lower paragraph touches first (i.e., it touches the bottom border of the upper paragraph), so it determines how far apart the elements will be set.

This effect is known as **collapsing**.



The Box Margins

- This collapsing margin effect ensures that when an element is first or last in a group of headings, paragraphs, or lists, for example, the element can be kept away from the top or bottom of the page or containing element.
- When the same elements appear between other elements, both margins are not needed, so they simply collapse into each other, and the larger one sets the distance.



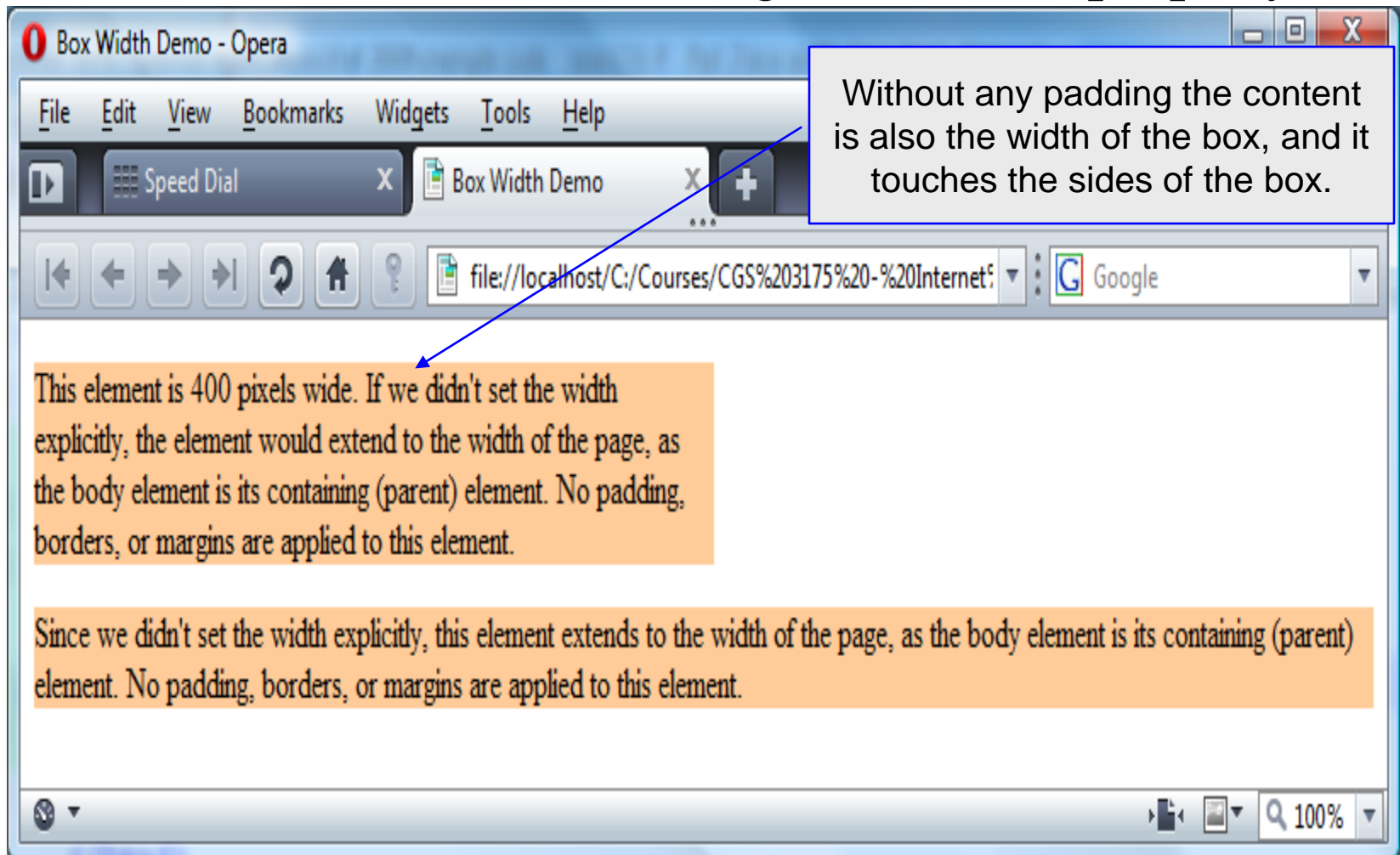
How Big Is A Box?

- The way the box model works is the root of most problems for beginning (and some expert) CSS developers.
- For now we'll focus only on block level elements such as headings, paragraphs, and lists. Inline elements will behave differently.
- Let's go back over the box model step-by-step in a little more depth, focusing first on the width of a box, since managing element width is critical to creating multi-column layouts (but the same basic logic will apply to the height of boxes as well).



How Big Is A Box?

- The width of a box is set using the `width` property.



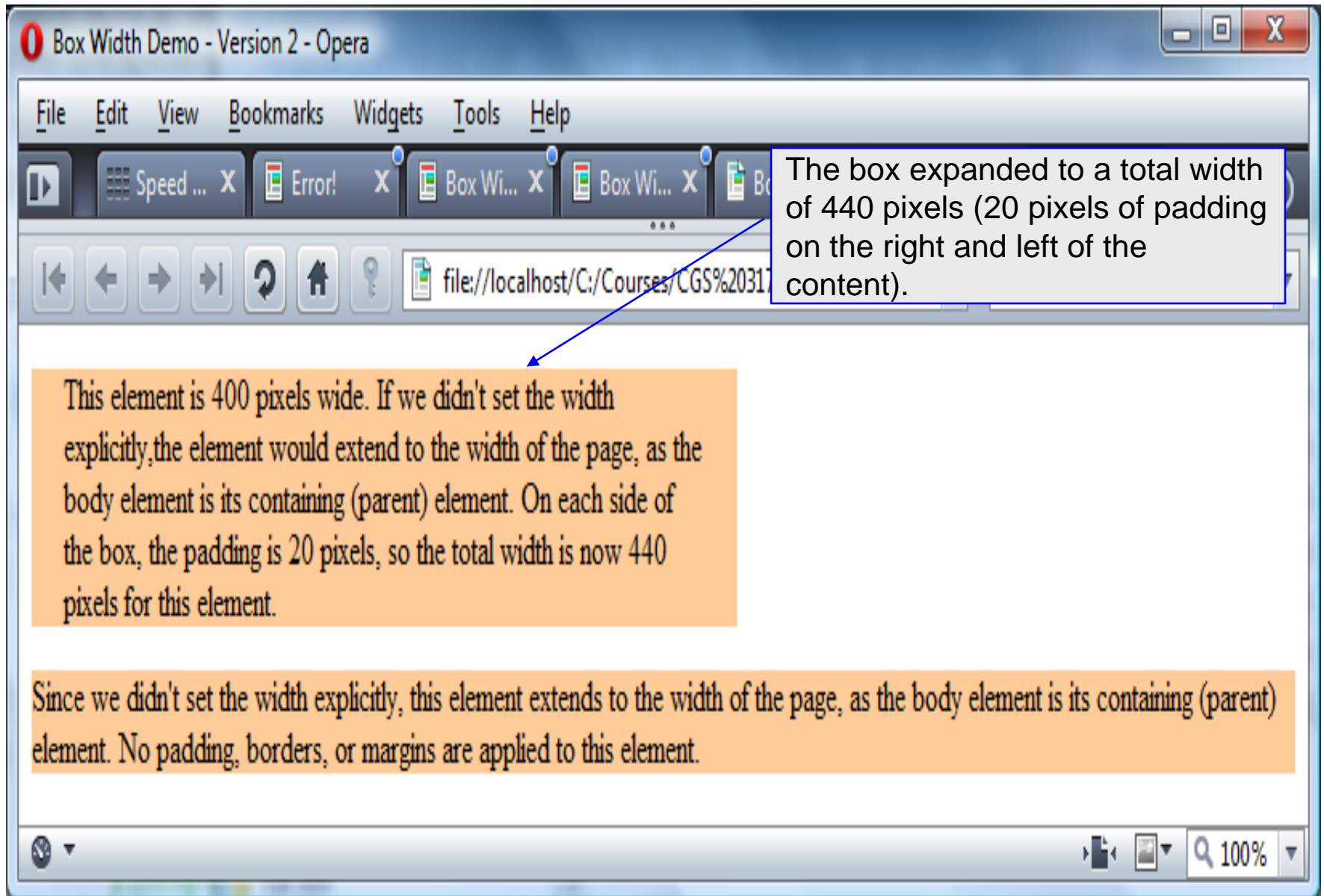
```
4 <html xmlns="http://www.w3.org/1999/xhtml">
5   <head>
6     <title>Box Width Demo</title>
7     <style type="text/css">
8       <!--
9         p.box {width:400px; background-color: #FFCC99;}
10        p {background-color: #FFCC99;}
11      -->
12    </style>
13  </head>
14  <body>
15    <p class="box">This element is 400 pixels wide.  If we didn't set the width
16      explicitly, the element would extend to the width of the page, as the body
17      element is its containing (parent) element.  No padding, borders, or margins
18      are applied to this element.
19    </p>
20    <p>Since we didn't set the width explicitly, this element extends to the width
21      of the page, as the body element is its containing (parent) element.
22      No padding, borders, or margins are applied to this element.
23    <p>
24  </body>
25 </html>
```



How Big Is A Box?

- In the previous example, since we did not include any padding in the box, the content extended to the very edge of the box.
- What happens when we add some padding to the box, in this case 20 pixels to the right and left sides of the box?
- What will be the total size of the box? In other words, we set its width to be 400 pixels, will the content now be squished into 360 pixels of width, or will the box expand to a total of 440 pixels in width?





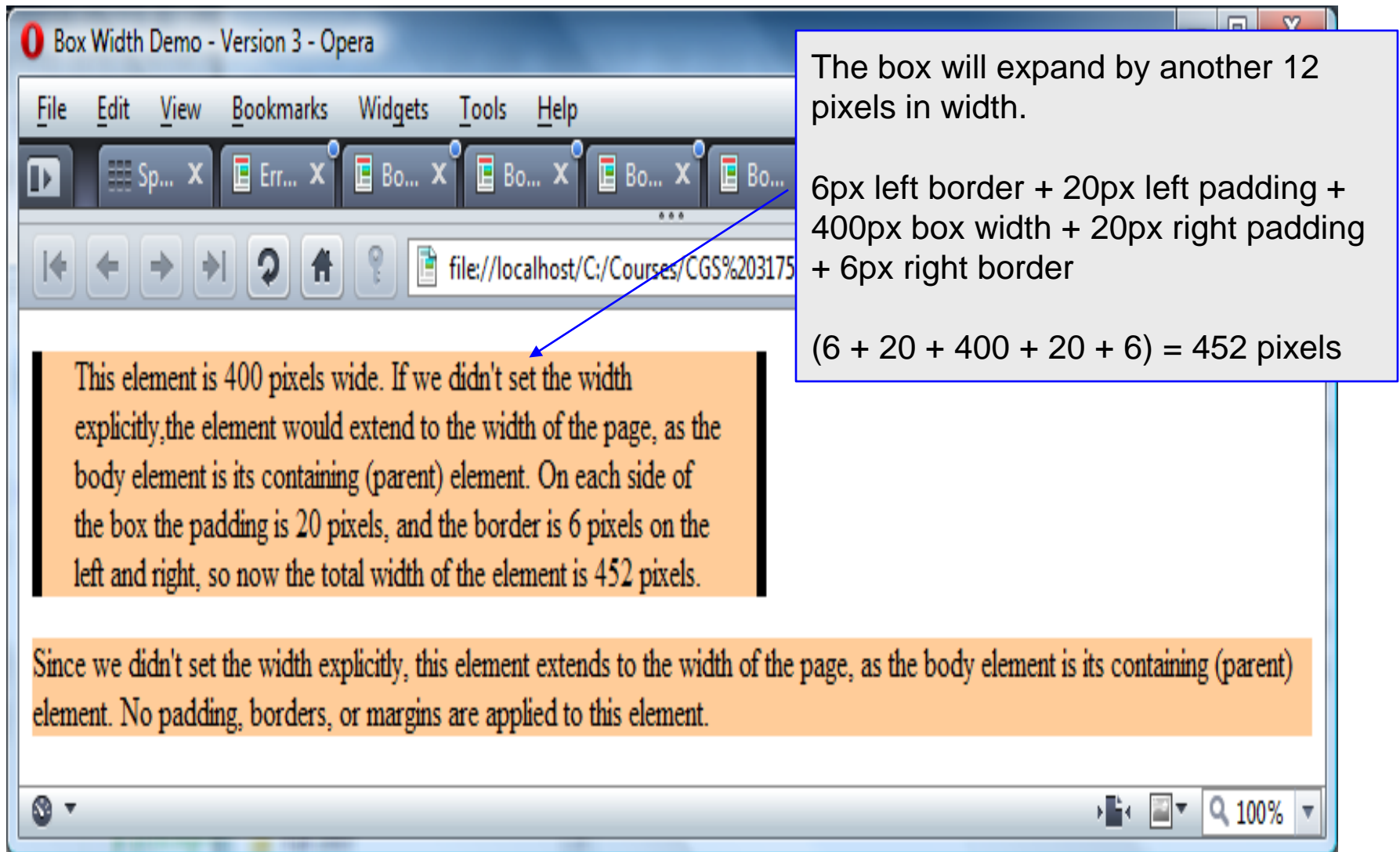
How Big Is A Box?

- What would happen to the overall size of the box if we now add a 6 pixel border to the right and left sides of the box?

```
5 <head>
6   <title>Box Width Demo - Version 3</title>
7   <style type="text/css">
8     <!--
9       p.box {width:400px; background-color: #FFCC99; padding: 0 20px; border: black solid; border-width: 0 6px 0 6px;}
10      p {background-color: #FFCC99; }
11    -->
12  </style>
13 </head>
14 <body>
15   <p class="box">This element is 400 pixels wide.  If we didn't set the width
16     explicitly, the element would extend to the width of the page, as the body
17     element is its containing (parent) element.  On each side of the box the
18     padding is 20 pixels, and the border is 6 pixels on the left and right,
19     so now the total width of the element is 452 pixels.
20   </p>
21   <p>Since we didn't set the width explicitly, this element extends to the width
```



How Big Is A Box?



The box will expand by another 12 pixels in width.

6px left border + 20px left padding + 400px box width + 20px right padding + 6px right border

$$(6 + 20 + 400 + 20 + 6) = 452 \text{ pixels}$$

This element is 400 pixels wide. If we didn't set the width explicitly, the element would extend to the width of the page, as the body element is its containing (parent) element. On each side of the box the padding is 20 pixels, and the border is 6 pixels on the left and right, so now the total width of the element is 452 pixels.

Since we didn't set the width explicitly, this element extends to the width of the page, as the body element is its containing (parent) element. No padding, borders, or margins are applied to this element.



How Big Is A Box?

- Now let's add left and right margins to create space around the sides of the element. Now how much space will the element occupy?

```
5  </head>
6  <title>Box Width Demo - Version 4</title>
7  <style type="text/css">
8  <!--
9      p.box {width:400px; background-color: #FFCC99; padding: 0 20px; border: black solid; border-width: 0 6px 0 6px;
10         margin:0 30px;}
11      p {background-color: #FFCC99; }
12  -->
13  </style>
14  </head>
15  <body>
16      <p class="box">This element is 400 pixels wide.  If we didn't set the width
17         explicitly, the element would extend to the width of the page, as the body
18         element is its containing (parent) element.  On each side of the box the
19         padding is 20 pixels and the left and right borders are 6 pixels each, so
20         now the total width of the element is 452 pixels.  With the addition of
21         30 pixels of margins on the left and right sides of the element, the total
22         width claimed by this element will be 512 pixels.
```



How Big Is A Box?

Box Width Demo - Version 4 - Opera

The margins add space around the element but do not expand the size of the element itself, since the margins are outside of the box.

This element is 400 pixels wide. If we didn't set the width explicitly, the element would extend to the width of the page, as the body element is its containing (parent) element. On each side of the box the padding is 20 pixels and the left and right borders are 6 pixels each, so now the total width of the element is 452 pixels. With the addition of 30 pixels of margins on the left and right sides of the element, the total width claimed by this element will be 512 pixels.

Since we didn't set the width explicitly, this element extends to the width of the page, as the body element is its containing (parent) element. No padding, borders, or margins are applied to this element.

file://localhost/C:/Courses/CGS%203175%20-%20Internet%20Applications/... Google

100%



Box Model Observation #1

- Dimensional boxes (those where the width is explicitly specified) expand to occupy more horizontal space as padding, borders, and margins are added.
- Effectively, the width property sets the width of the box's content, not the box itself when the box's width is explicitly stated.
- This behavior can have important implications if you build a layout with multiple columns where the columns must maintain their widths for the layout to work properly.



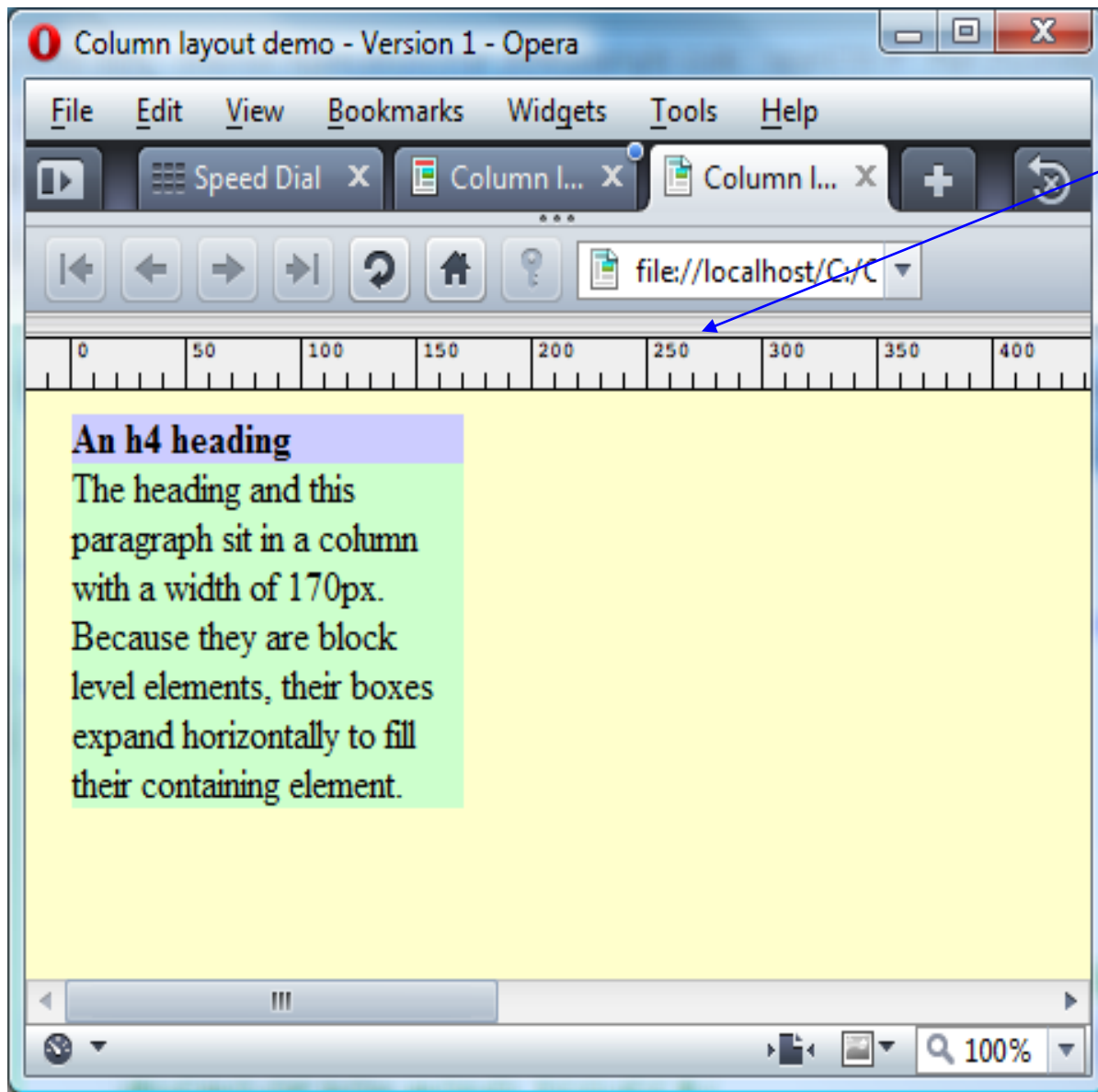
Box Model Observation #1

- Floated layouts (which we'll get to very soon) can display incorrectly if a column width gets inadvertently altered by changes to the padding, margins, or borders.
- Typically, you will create a column in your layout using a dimensioned (defined width) `<div>` and then nest all the column's content elements (headings, paragraphs, navigation lists, and so on) inside it.
- The example on the following few pages will illustrate this concept.




```
C:\Courses\CGS 3175 - Internet Applications\Fall 2009\example code - new\CSS-P - Part 2\column layout demo.html - Not...
File Edit Search View Format Language Settings Macro Run TextFX Plugins Window ?
xhtml strict template.html CSS-P - Part 1 - practice problem 3.html fig4.12_expanding_blocks1.html column layout demo.html
1 <?xml version="1.0" encoding="UTF-8" standalone="no"?>
2 <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
3 "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
4 <html xmlns="http://www.w3.org/1999/xhtml">
5 <head>
6 <title>Column layout demo - Version 1</title>
7 <style type="text/css">
8 <!--
9 /* set up for demo rules */
10 * {margin:0px; padding:0;}
11 body {padding-left:20px; background-color:#FFC;}
12 #ruler {position:relative; left:-51px; top:0px; margin-bottom:5px;}
13
14 /*DEMO OF BOX MODEL ISSUES*/
15 #column { width:170px; background:#FCC; /* not visible in this first step */ }
16 h4 {background:#CCF;}
17 p {background:#CFC;}
18 -->
19 </style>
20 </head>
21 <body>
22 
23 <div id="column">
```





I added a ruler graphic to the top of the window so you can see the width change as the CSS changes.

I've also colored the backgrounds of the heading and paragraph so you can see that they completely fill the column horizontally.

Block level elements have a default size of auto, which effectively means "as large as possible."



Box Model Observation #2

- Undimensioned elements (width not explicitly set) will always fill the width of their corresponding containing element.
- Because of this, adding horizontal margins, borders, and padding to an undimensioned element does cause the element to change width.
- Continuing with the example, since the text is jammed against the sides of the column (see page 34), your first instinct might be to add some padding to the `<div>` to create some breathing space around the text. Let's do this!



C:\Courses\CGS 3175 - Internet Applications\Fall 2009\example code - new\CSS-P - Part 2\column layout demo - version 2...

File Edit Search View Format Language Settings Macro Run TextFX Plugins Window ?

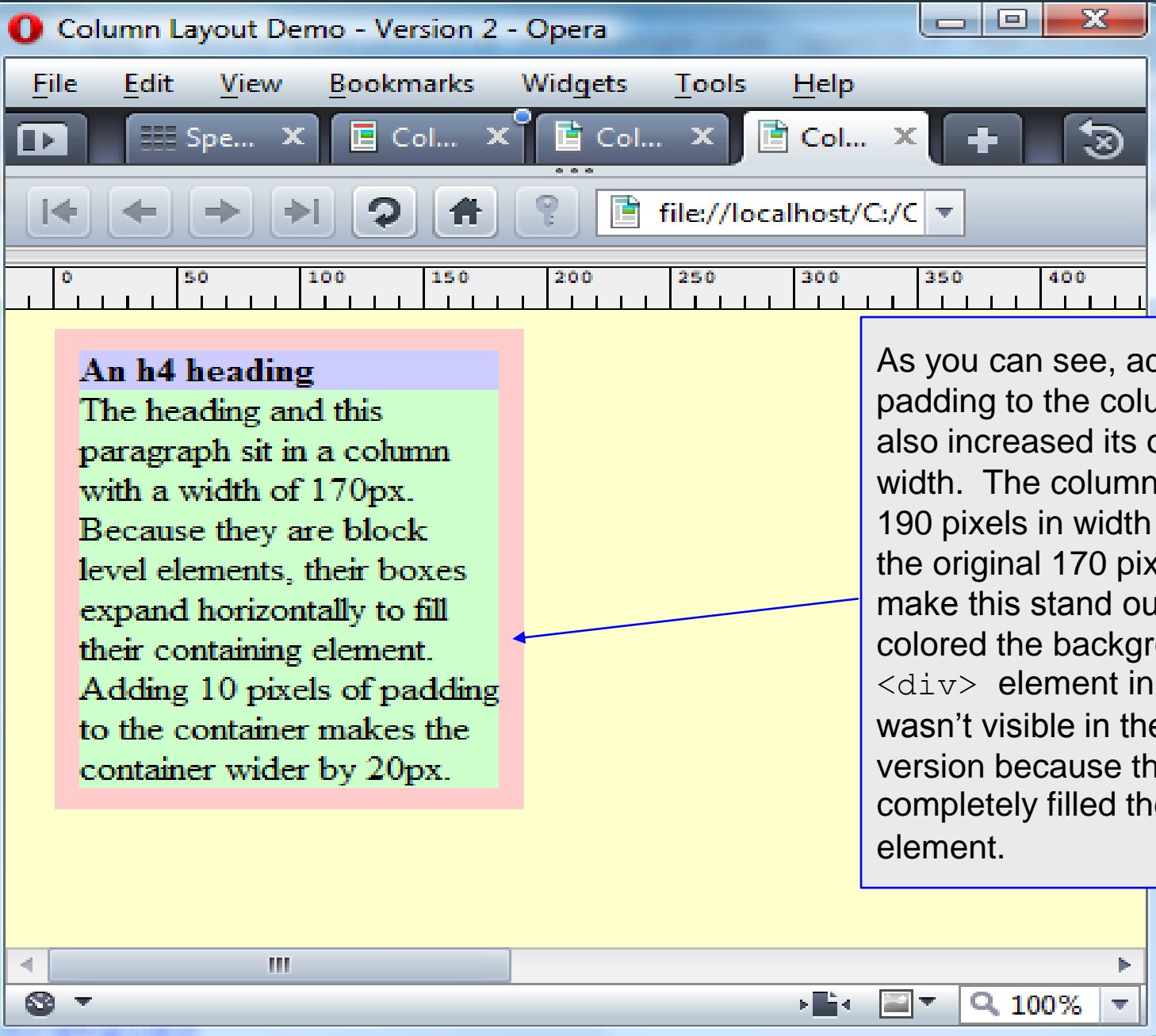
column layout demo - version 2.html CSS-P - Part 1 - practice problem 3.html fig4.12_expanding_blocks1.html column layout demo.html

```
7 <style type="text/css">
8 <!--
9     /* set up demo rules */
10    * {margin:0px; padding:0;}
11    body {padding-left:20px; background-color:#FFC;}
12    #ruler {position:relative; left:-51px; top:0px; margin-bottom:5px;}
13
14    /*DEMO OF BOX MODEL ISSUES */
15    #column {
16        width:170px;
17        padding:10px;
18        background:#FCC;
19    }
20    h4 {background:#CCF;}
21    p {background:#CFC;}
22 -->
23 </style>
24 </head>
25 <body>
26 
27 <div id="column">
28     <div id="column_inner">
29         <h4>An h4 heading</h4>
```

10 pixels of padding added to all sides of the column

Hyper Te nb char: 1078 nb line: 37 Ln: 17 Col: 22 Sel: 0 UNIX ANSI INS





Box Model Observation #2

- As you can see in the previous slide using the ruler graphic, the 10 pixels of padding added to each side of the column has increased its width to 190 pixels. While this neatly pads all elements inside the `<div>` away from the edges with a single style, in order to keep the overall width at 170 pixels, we would now have to subtract the corresponding amount ($10+10=20\text{px}$) from the current box width value and set it to 150 pixels.
- It gets tiresome to keep changing the column width every time you alter the column padding, especially with a multi-column layout.



Box Model Observation #2

- An alternative is to apply identical margins to every element inside the column, but that again can mean a lot of elements to keep track of and change if we decide to adjust the distance between the column's sides and its content.
- The simple solution to this problem is to add another `<div>` immediately inside the column `<div>`. The padding is then applied to this inner `<div>`. This allows you to have a single style to control the column padding without having issues with the column changing width. This is illustrated on the next page.



*C:\Courses\CGS 3175 - Internet Applications\Fall 2009\example code - new\CSS-P - Part 2\column layout demo.html - No...

File Edit Search View Format Language Settings Macro Run TextFX

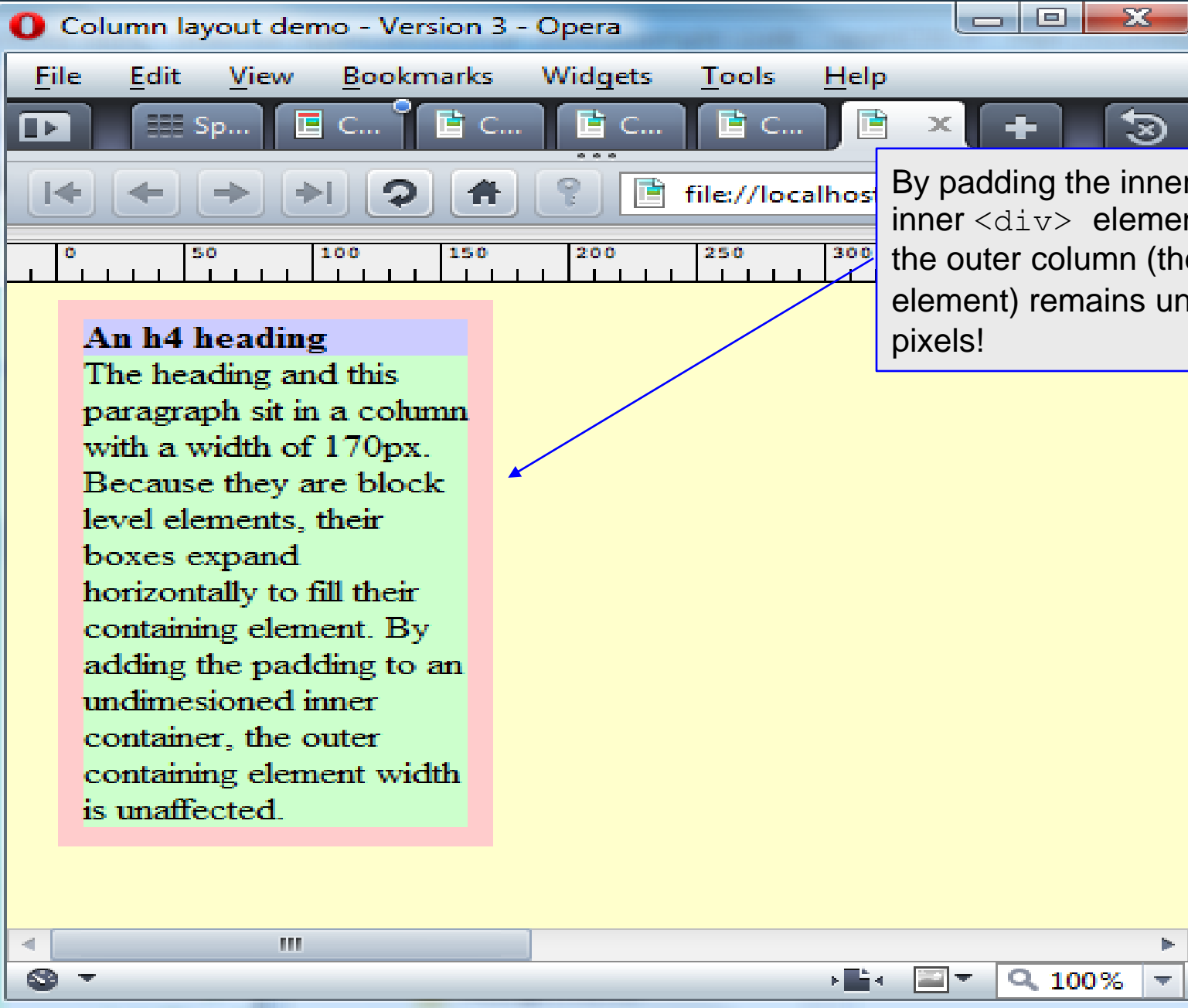
CSS-P - Part 1 - practice problem 3.html fig4.12_expanding_blocks1.html column la

```
4 <html xmlns="http://www.w3.org/1999/xhtml">
5 <head>
6   <title>Column layout demo - Version 3</title>
7   <style type="text/css">
8     <!--
9     /* set up demo rules */
10    * {margin:0px; padding:0;}
11    body {padding-left:20px; background-color:#FFC;}
12    #ruler {position:relative; left:-51px; top:0px; margin-bottom:5px;}
13
14    /*DEMO OF BOX MODEL ISSUES*/
15    #column {
16      width:170px;
17      background:#FCC;
18    }
19    #column_inner {
20      padding:10px; /* creates space around the content */
21    }
22    h4 {background:#CCF;}
23    p {background:#CFC;}
24    -->
25  </style>
26 </head>
```

The column id is used to style the outer column for setting the width property of the column and the column_inner id is used to style the padding for the content inside the outer column.

Hyper Te nb char: 1196 nb line: 38 Ln: 6 Col: 31 Sel: 0 Dos\Windows ANSI INS





By padding the inner column (the inner `<div>` element) the width of the outer column (the outer `<div>` element) remains unchanged at 170 pixels!

An h4 heading

The heading and this paragraph sit in a column with a width of 170px. Because they are block level elements, their boxes expand horizontally to fill their containing element. By adding the padding to an undimensioned inner container, the outer containing element width is unaffected.



Box Model Observation #2

- Since the inner `<div>` is undimensioned, Box Model Observation #2 applies, and the content gets squeezed down. Now by adjusting just one margin setting, all the elements inside the column can be moved away from its edge and the column's overall width remains unchanged.
- We'll make use of this technique in many of the upcoming page layouts, so be sure that you understand this technique.



The Box Model

- The main thing to remember is that with all modern browsers when you set the width of an element, you are really setting the width of the content within it, and any padding, borders, and margins you set increase the overall space the element occupies over and above the specified width value.

