

# Computer Science Principles

## Web Programming

### Web-Presentation Design with CSS

CHAPTER 14: BOX MODEL

DR. ERIC CHOU

IEEE SENIOR MEMBER



# CSS3

---

Chapter 11: CSS Hierarchy and Selectors

Chapter 12: Text, Image and Foreground (Contents)

Chapter 13: Color and Background (Contents)

**Chapter 14: Box Model (Padding, Border, and Margin)**

Chapter 15: Layout Management (Floating and Positioning: where should the Element go)

Chapter 16: Layout Management (Page Level Planning)

Chapter 17: Layout Management (Transition, Transforms, and Animation: space and time domain transformation)

Chapter 18: CSS Techniques (Put Everything Together)

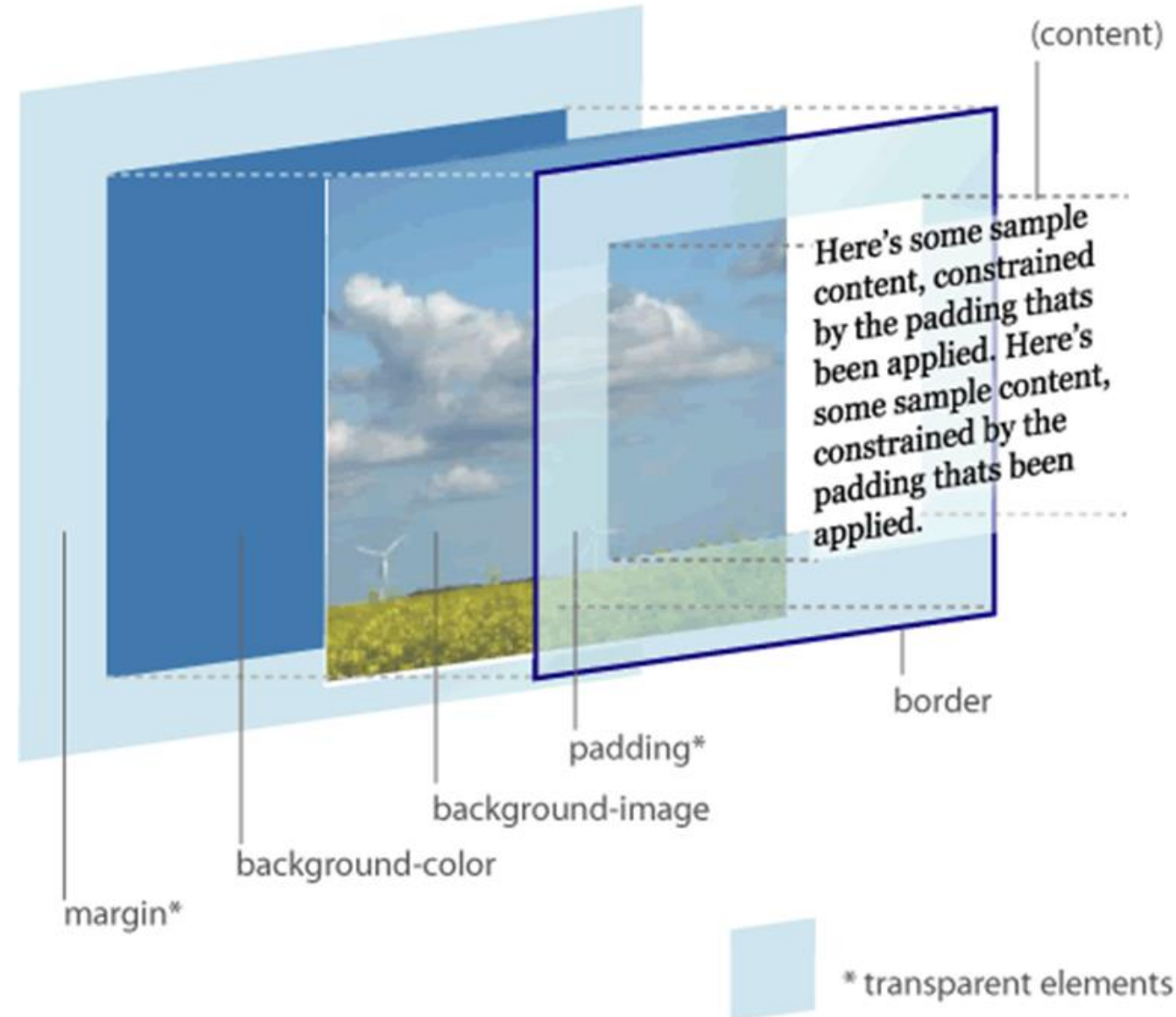


# Overview

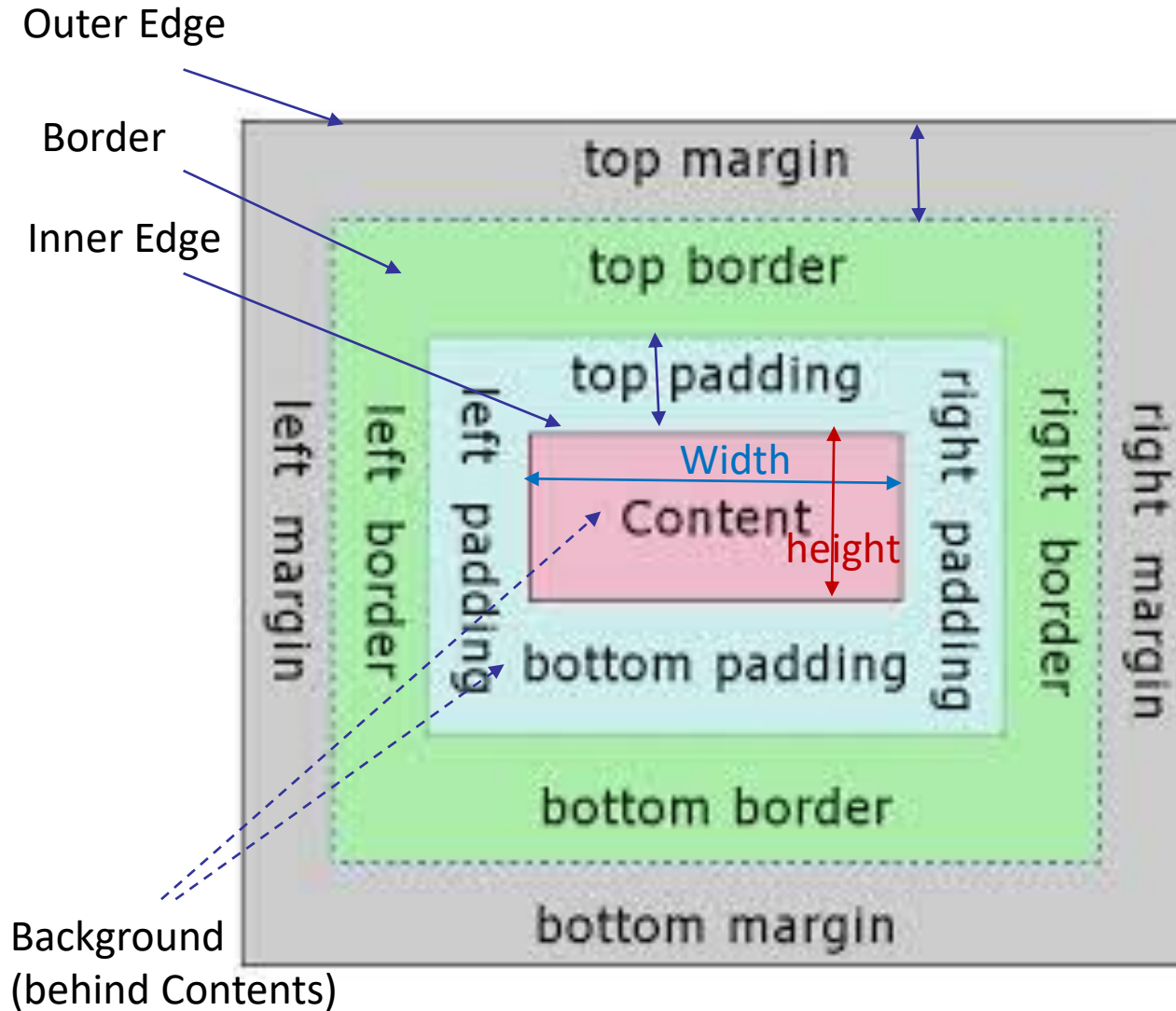
---

LECTURE 1

## THE CSS BOX MODEL HIERARCHY



# CSS3 Box Model



# Box Classes

Box model applies to all of the Elements.

**Text**

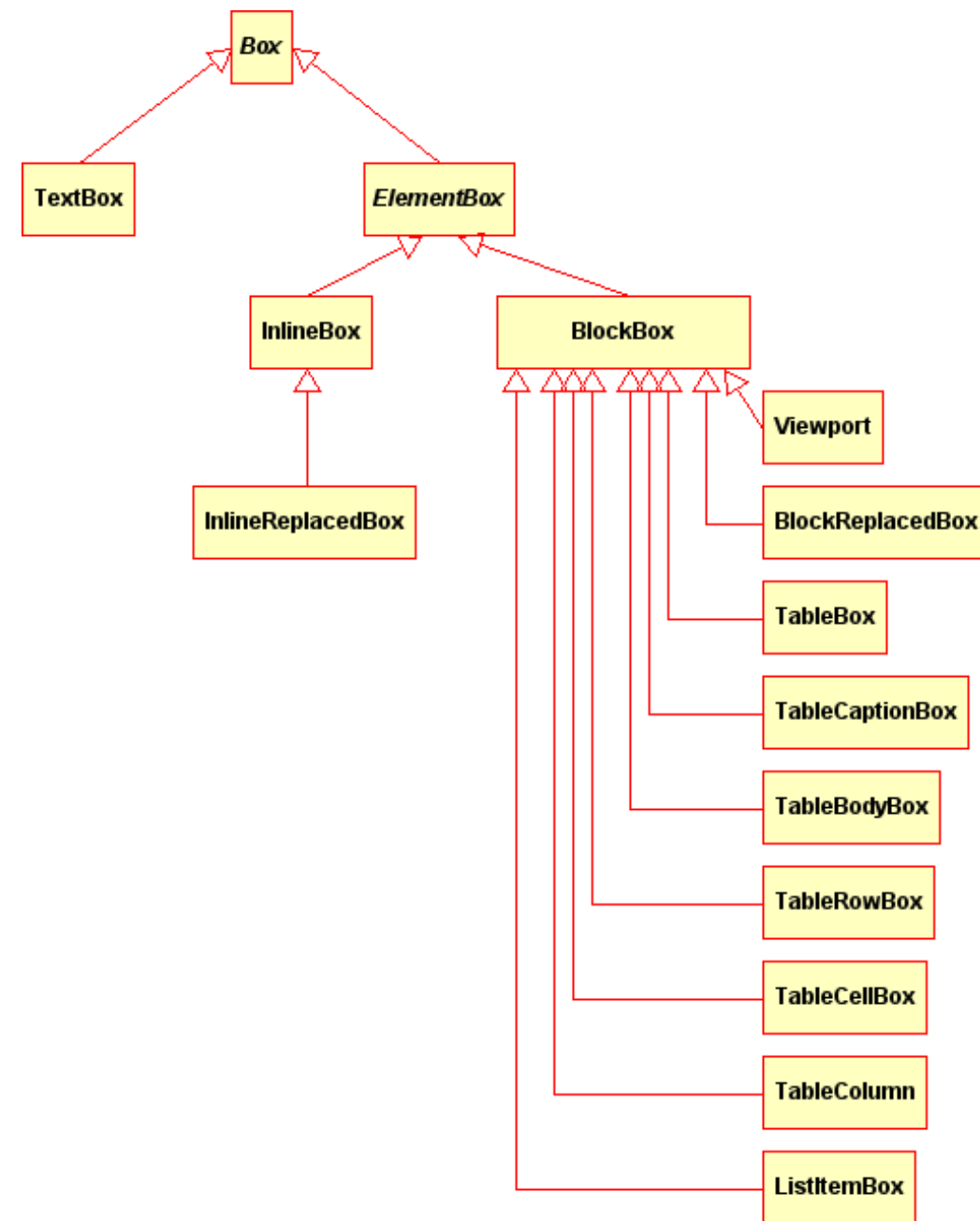
**Table**

**Image**

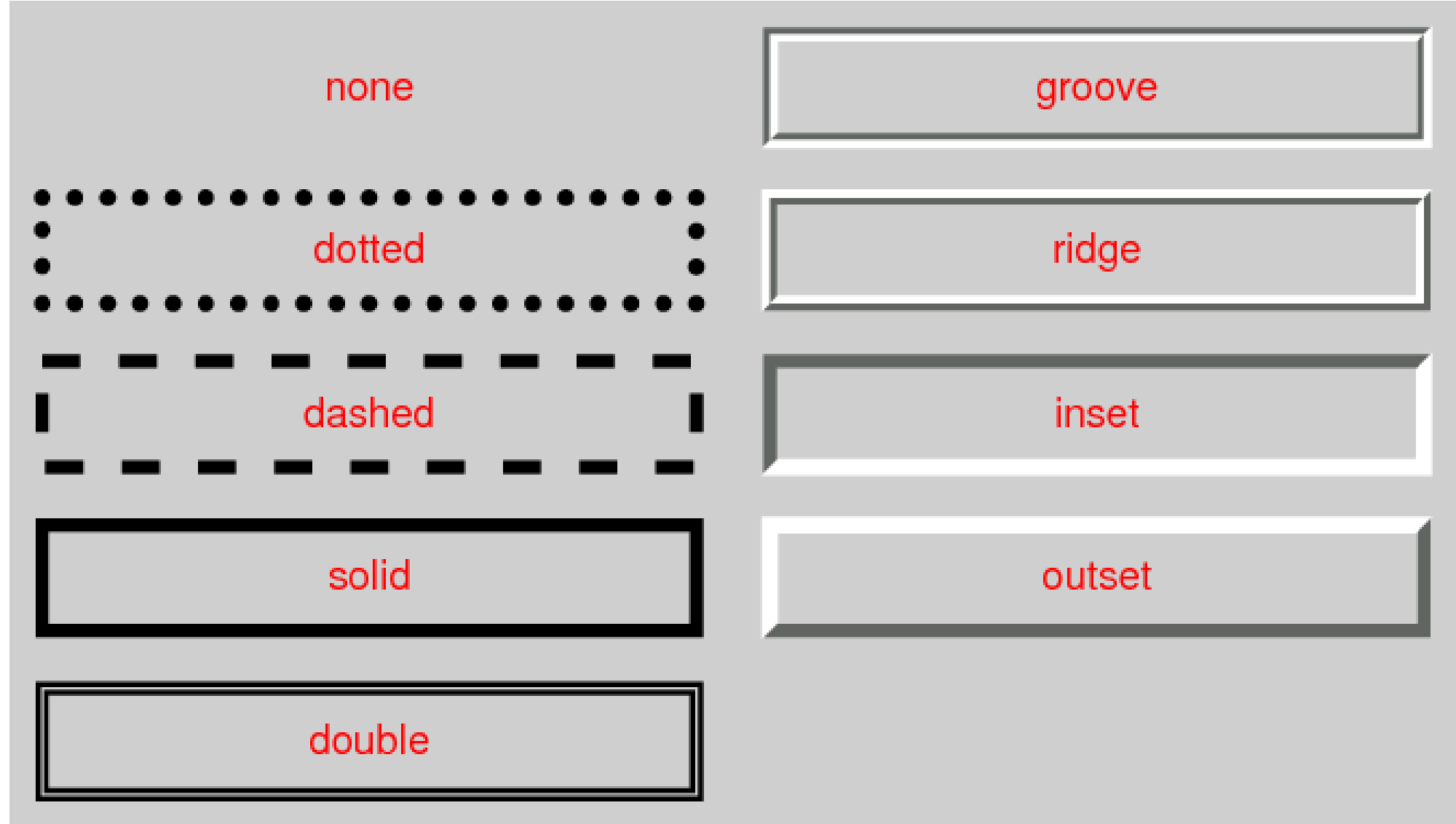
**List Item**

**Form**

And many others.



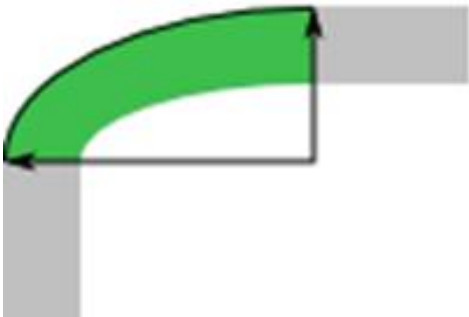
# Border



# Radius and Width

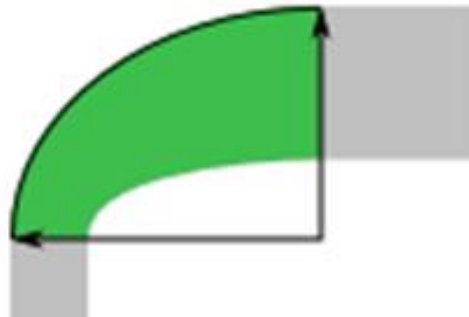
A

```
border-width: 20px;  
border-radius:  
80px / 40px;
```



B

```
border-width:  
40px 20px;  
border-radius:  
80px / 60px;
```



C

```
border-width:  
40px 80px;  
border-radius:  
80px / 40px;
```



D

```
border-width:  
60px 40px;  
border-radius:  
80px / 40px;
```





Example 1

```
-moz-border-radius: 1em;
```

Example 2

```
-moz-border-radius-topright: 2em;  
-moz-border-radius-topleft: 2em;
```

Example 3

```
-moz-border-radius: 2em 0;
```

Example 4

```
-moz-border-radius: 3em 1em;
```

Example 5

```
-webkit-border-radius: 1em;
```

Example 6

```
-webkit-border-top-right-radius: 24px;  
-webkit-border-top-left-radius: 24px;
```

Example 7

```
-webkit-border-radius: 24px 0;
```

Example 8

```
-webkit-border-radius: 36px 12px;
```

Example 9

```
-webkit-border-top-right-radius: 40px 30px;  
-webkit-border-bottom-right-radius: 40px 30px;
```

# CSS Padding

## Short hand

**padding: 25px 50px 75px 100px;**

top padding is 25px  
right padding is 50px  
bottom padding is 75px  
left padding is 100px

**padding: 25px 50px 75px;**

top padding is 25px  
right and left padding are 50px  
bottom padding is 75px

**padding: 25px 50px;**

top and bottom padding are 25px  
right and left padding are 50px

**padding: 25px;**

all four padding are 25px

# CSS Margin

- margin-top
- margin-bottom
- margin-right
- margin-left

## General

```
margin-top:100px;  
margin-bottom:20px;  
margin-right:50px;  
margin-left:30px;
```



# Box

---

## LECTURE 2

# Box Dimension

---

- CSS has two ways to specify the size of an element. The default method = introduced way back in CSS1 – applies the width and height values to the content box.
- The other method – introduced as part of the new box-sizing property in CSS3 – introduced as part of the new box-sizing property in CSS3 – applies the width and height values to the border box, which includes the content, padding, and border.

**box-sizing: content-box | border-box**

**width: length measurement | percentage | auto | inherit**

**height: length measurement | percentage | auto | inherit**

# Example of Content Box Sizing

---

```
p {  
  background: #c2f670;  
  width: 500px;  
  height: 150px;  
  padding: 20px;  
  border: 2px solid gray;  
  margin: 20px;  
}
```

Total size for width: 20px + 2px + 20px + **500px** + 20px + 2px + 20px = 584 px

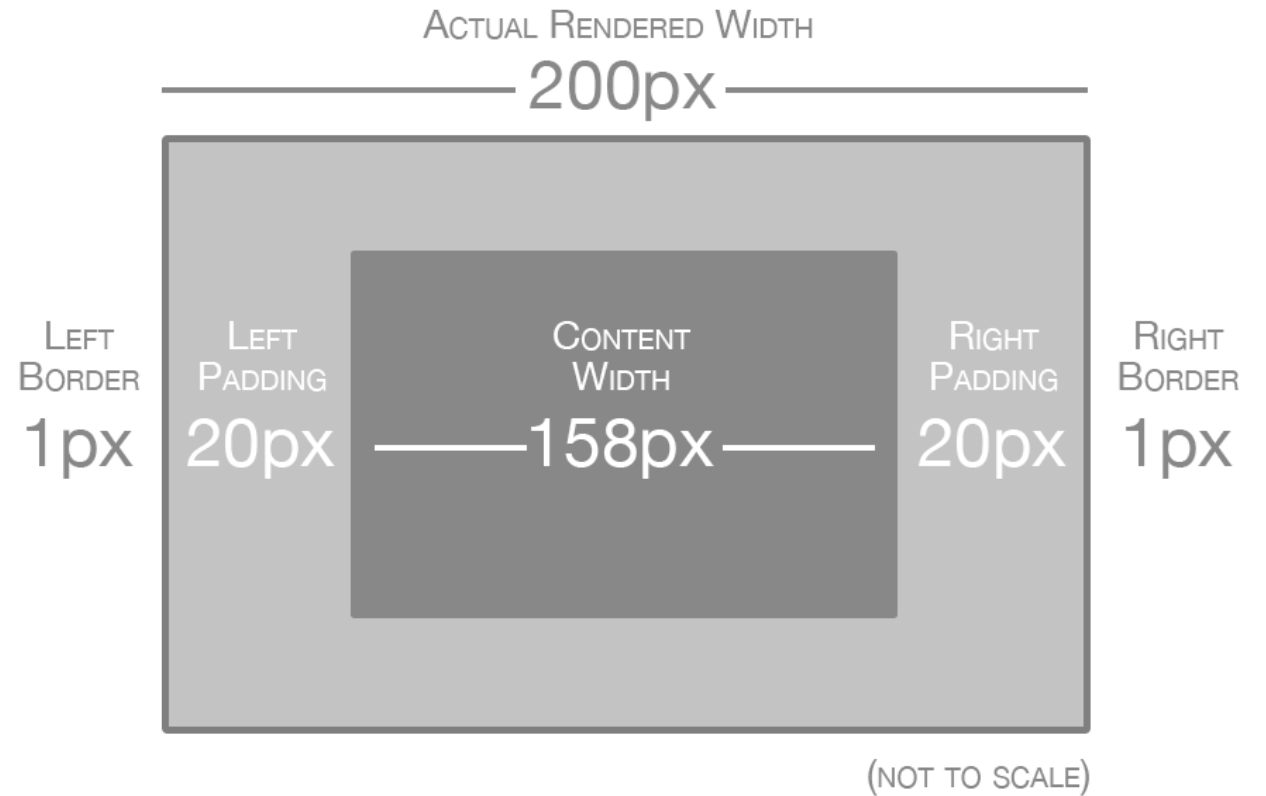
Total visible box width = 2px + 20px + **500px** + 20px + 2px = 544px

Total size for height: 20px + 2px + 20px + **150px** + 20px + 2px + 20px = 234px

Total visible box height = 2px + 20px + **150px** + 20px + 2px = 194px

# Content Box Sizing

```
.sidebar {
  width: 158px;
  padding: 20px;
  border: 1px solid #DDD;
}
```





# Example of Border Box Sizing

```
p {
  box-sizing: border-box
  background: #c2f670;
  width: 500px;
  height: 150px;
  padding: 20px;
  border: 2px solid gray;
  margin: 20px;
}
```

Total size for width: 20px + 500px + 20px = 540px

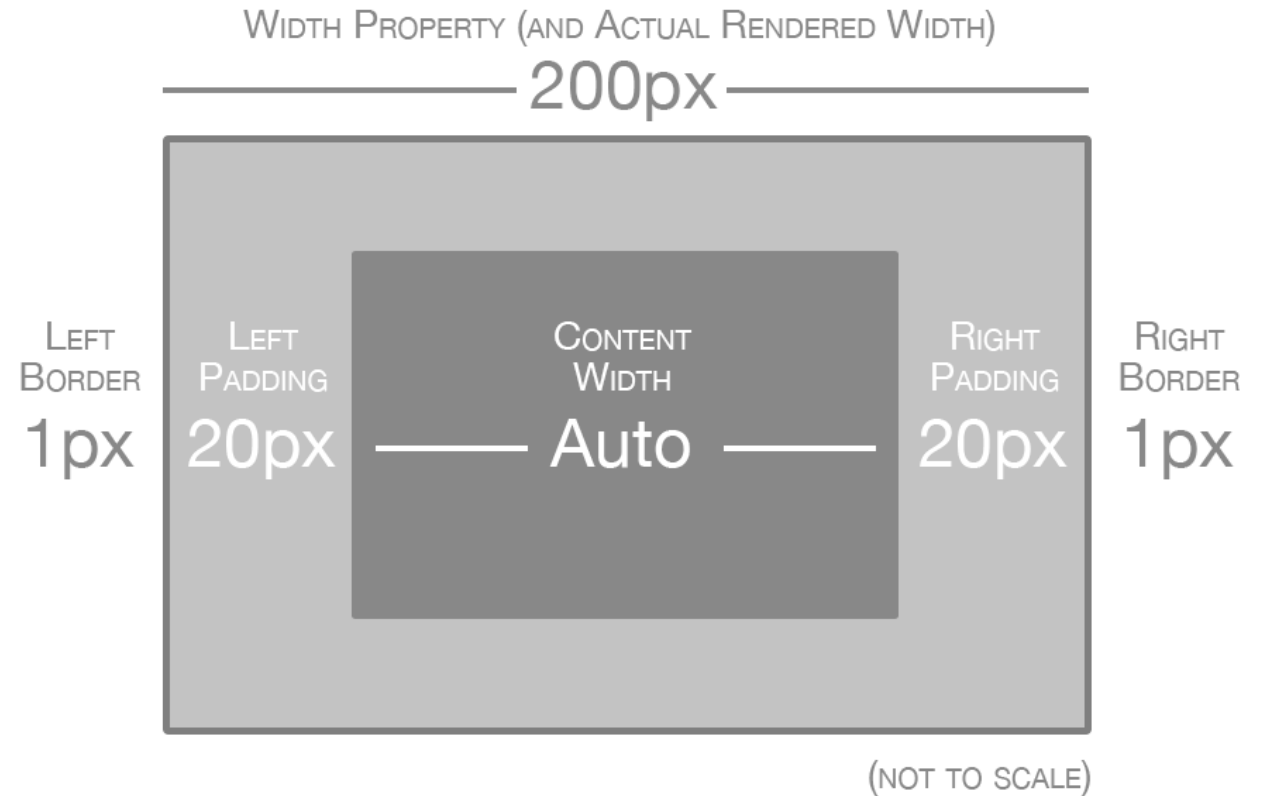
Total content box width = -2px - 20px + 500px - 20px - 2px = 456px

Total size for height: 20px + 150px + 20px = 234px

Total content box height = -2px - 20px + 150px - 20px - 2px = 116px

# Border Box Sizing

```
.sidebar {
  box-sizing: border-box;
  width: 200px;
  padding: 20px;
  border: 1px solid #DDD;
}
```



# max-height, max-width, min-height, min-width

---

**max-height, max-width, min-height, min-width: length | percentage | none | inherit**

- The properties work with block-level and replaced elements (like image) only. When the content-box model is used, the value applies to the content area only, so if you apply padding, borders, or margins, it will make the overall element box larger \, even if a max-width or max-height property has been specified. These properties are not supported by Internet Explorer 6 and earlier.

**max-height, max-width** is good for mobile device.

**min-height, min-width** can make sure the page is reasonably showing enough contents.



# Overflow

---

LECTURE 3

# Overflow handling

---

Overflow: **visible** | **hidden** | **scroll** | **auto** | **inherit**

**visible:** visible is default value, which allows the content to hang out over the element box so that it all can be seen.

**hidden:** when overflow is set to hidden , the content that does not fit gets clipped off and does not appear beyond the edges of the element's content area.

**scroll:** When scroll is specified, scrollbars are added to the element box to let users scroll through the content. Be aware that when you set the value to scroll, the scrollbars will always be there, even if the content fits in the specified height just fine.

**auto:** the auto value allows the browser to decide how to handle overflow. In most cases, scrollbars are added only when the content doesn't fit and they are needed.

# Padding

---

**padding-top:** length measurement | percentage | inherit

**padding-right:** length measurement | percentage | inherit

**padding-bottom:** length measurement | percentage | inherit

**padding-left:** length measurement | percentage | inherit

# Shorthand Padding Property

---

**padding:** length measurement (x N) | percentage (x N) | inherit, where N can be 1, 2, 3, 4 numbers.

1 value

padding: 10px

Applied to all sides

2 values

padding: 10px 6px

First top and bottom; second is left and right.

Try Exercise 14-1.

3 values

padding: 10px 6px 4px

First top, second left and right, third bottom.

4 values

padding: 10px 6px 4px 10px

First top, second right, third bottom, and fourth left.



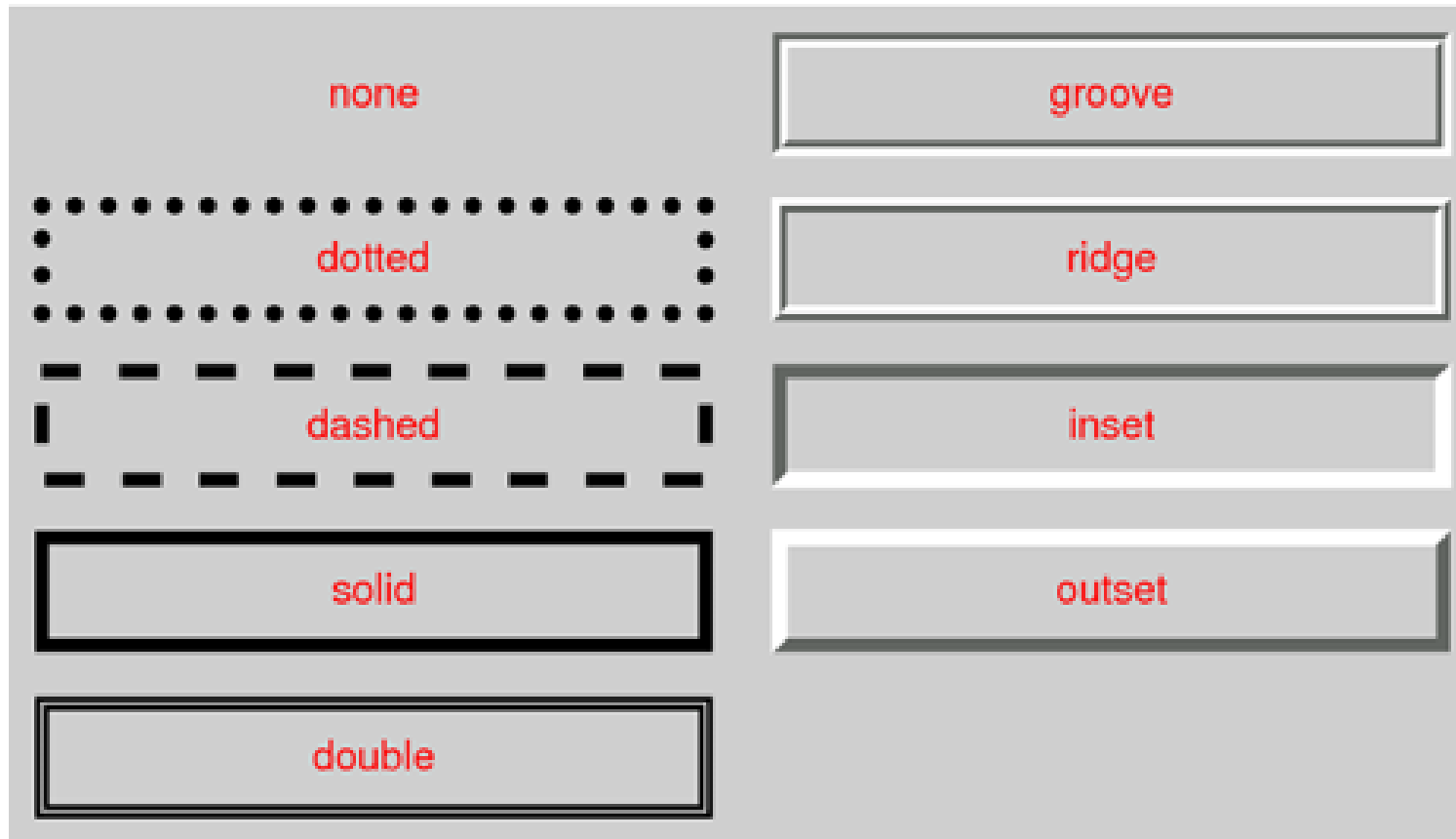
# Border Styles

---

LECTURE 4



# border-style



# border-width

border-width: **length units** | **thin** | **medium** | **thick** | **inherit**

```
border-width: thin;
```

```
border-width: medium;
```

```
border-width: thick;
```

# border-color

border-color: **color name** on **RGB value** | **transparent** | **inherit**

## Border Color

The border-color property is used to set the color of the border.

name - specify a color name, like "red"


RGB - specify a RGB value, like "rgb(255,0,0)"

Hex - specify a hex value, like "#ff0000"


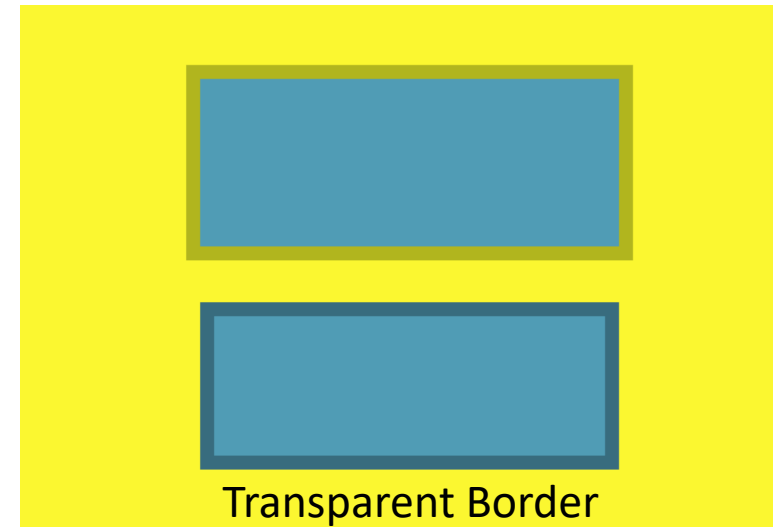
You can also set the border color to "transparent".

```
p.one
{
border-style:solid;
border-color:red;
}
p.two
{
border-style:solid;
border-color:#98bf21;
}
```

A solid red border



A solid green border

# border-radius

border-top-left-radius, border-top-right-radius, border-bottom-right-radius, border-bottom-left-radius:

length measurement | percentage

Border-radius: 1,2,3,4 length or percentage

1

2

4

3



No rounded corner



Rounded using an  
arc of circle



Rounded using an  
arc of ellipse

# border-radius

border-top-left-radius: 50px 25px

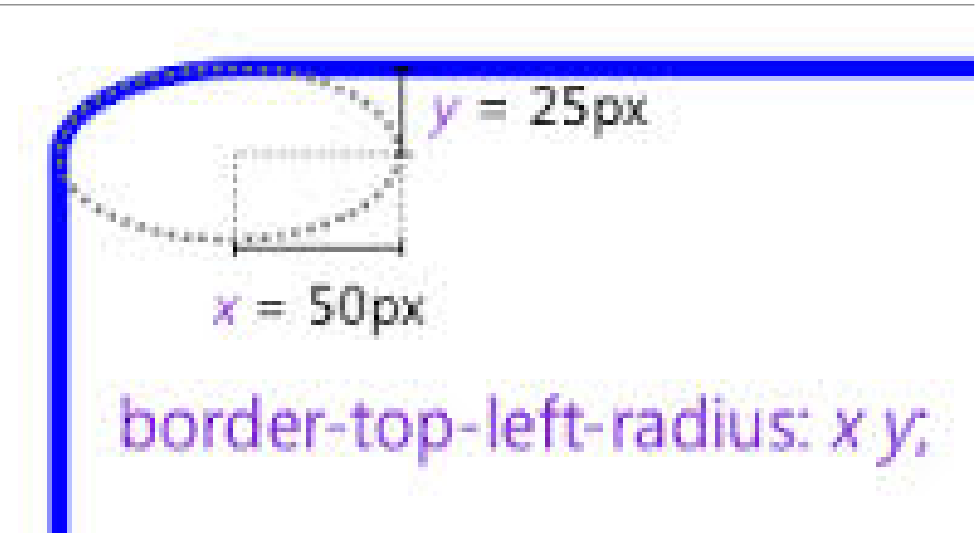
For all of the four possible border corners, the first value is for x and the second value is for y.

border-radius: 60px / 40px ;

( x / y )

border-radius: 36px 40px 60px 20px / 12px 10px 30px 36px

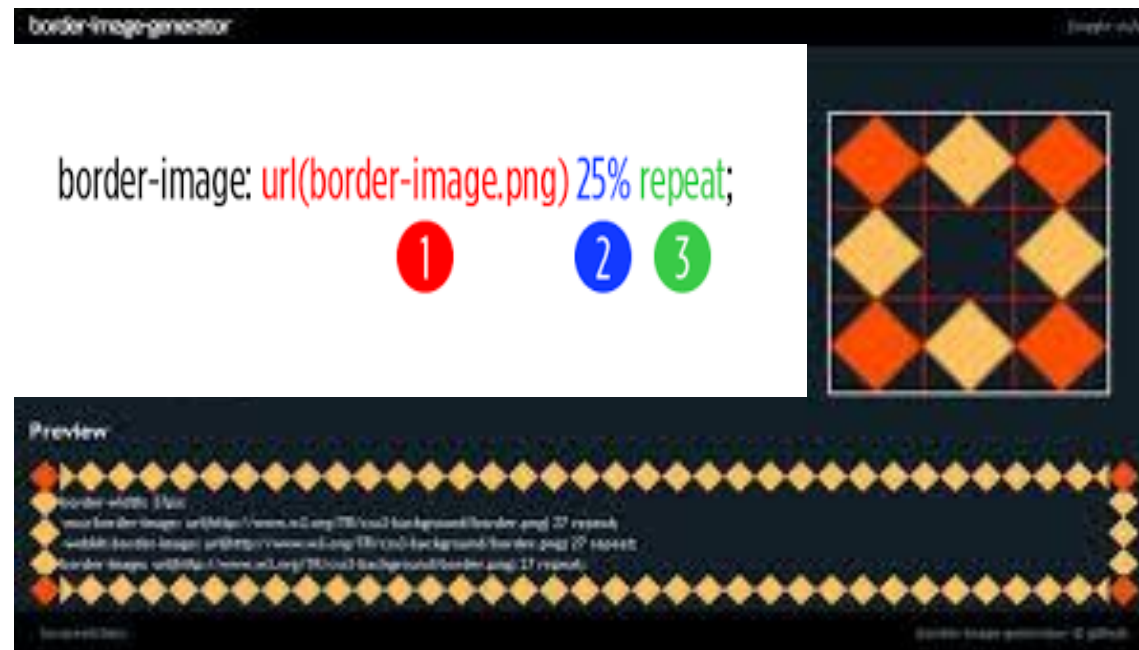
( x1 x2 x3 x4 / y1 y2 y3 y4 )



# border-image

border-image: border-image-source border-image-slice (1 2 3 4, rule same as padding shorthand) border-image-width

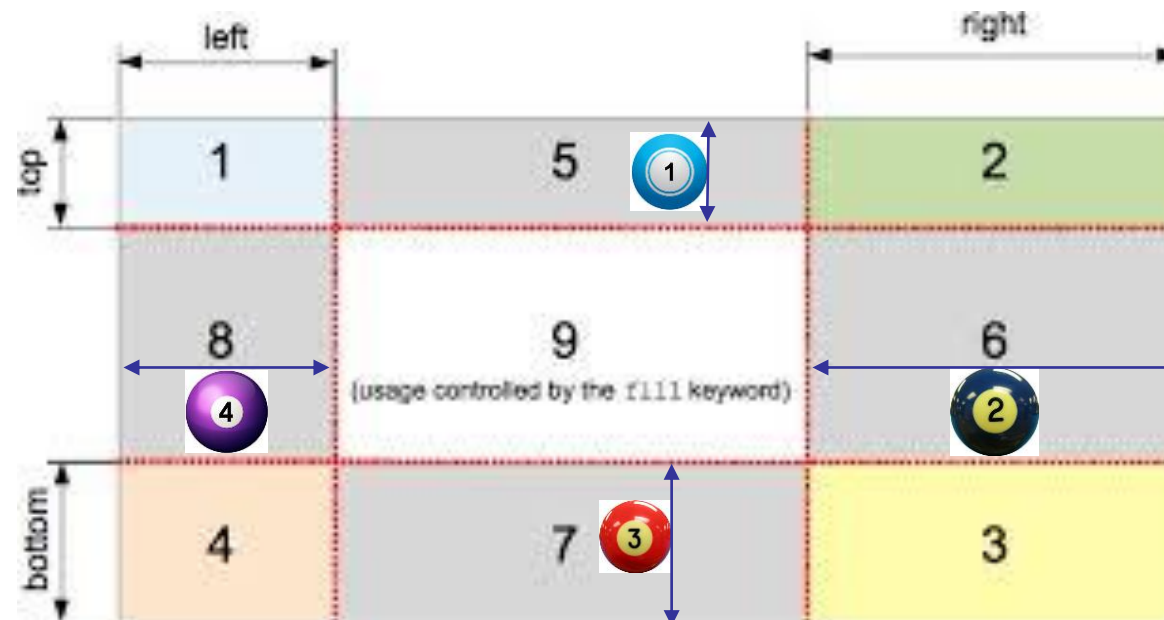
border-image-outset border-image-repeat



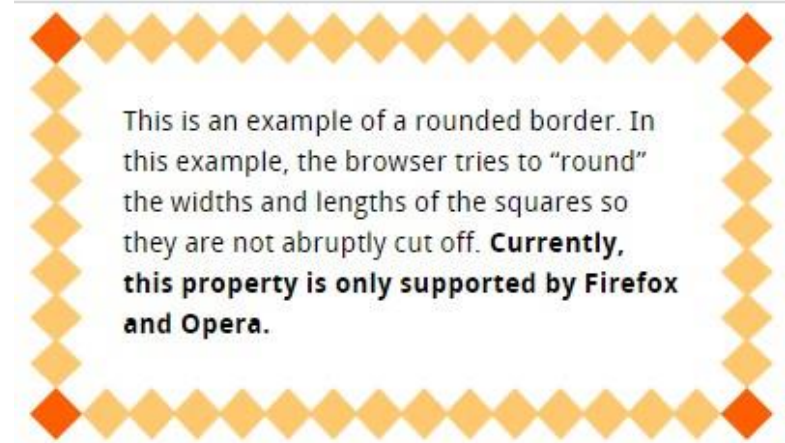
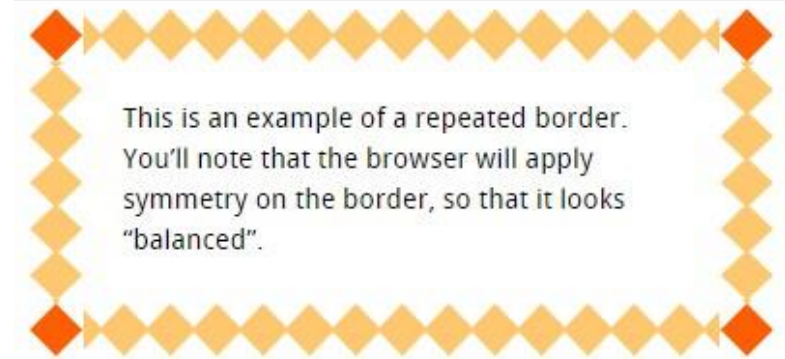
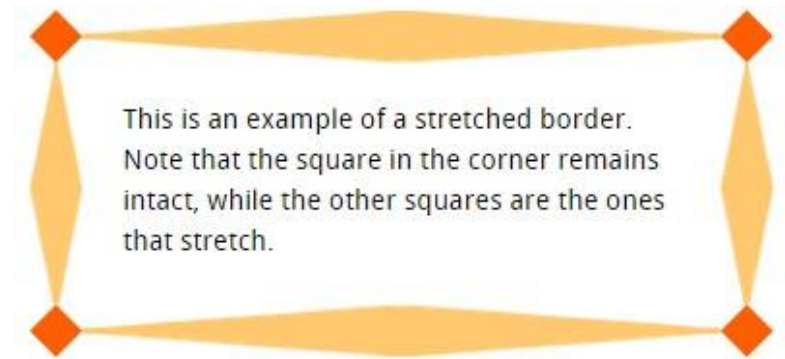
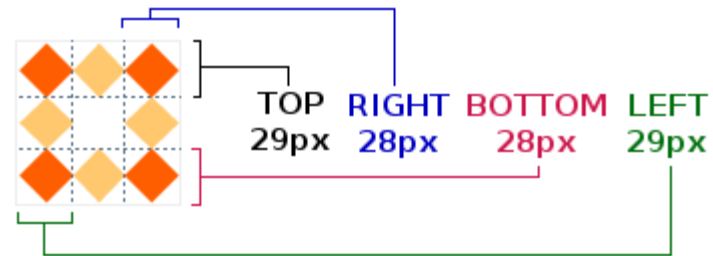
# border-image

border-image: border-image-source border-image-slice (1 2 3 4, rule same as padding shorthand) border-image-width

border-image-outset border-image-repeat



# border-image-slice border-image-repeat





# border-image-outset

---



# margins

margin: length measurement | percentage | auto | inherit

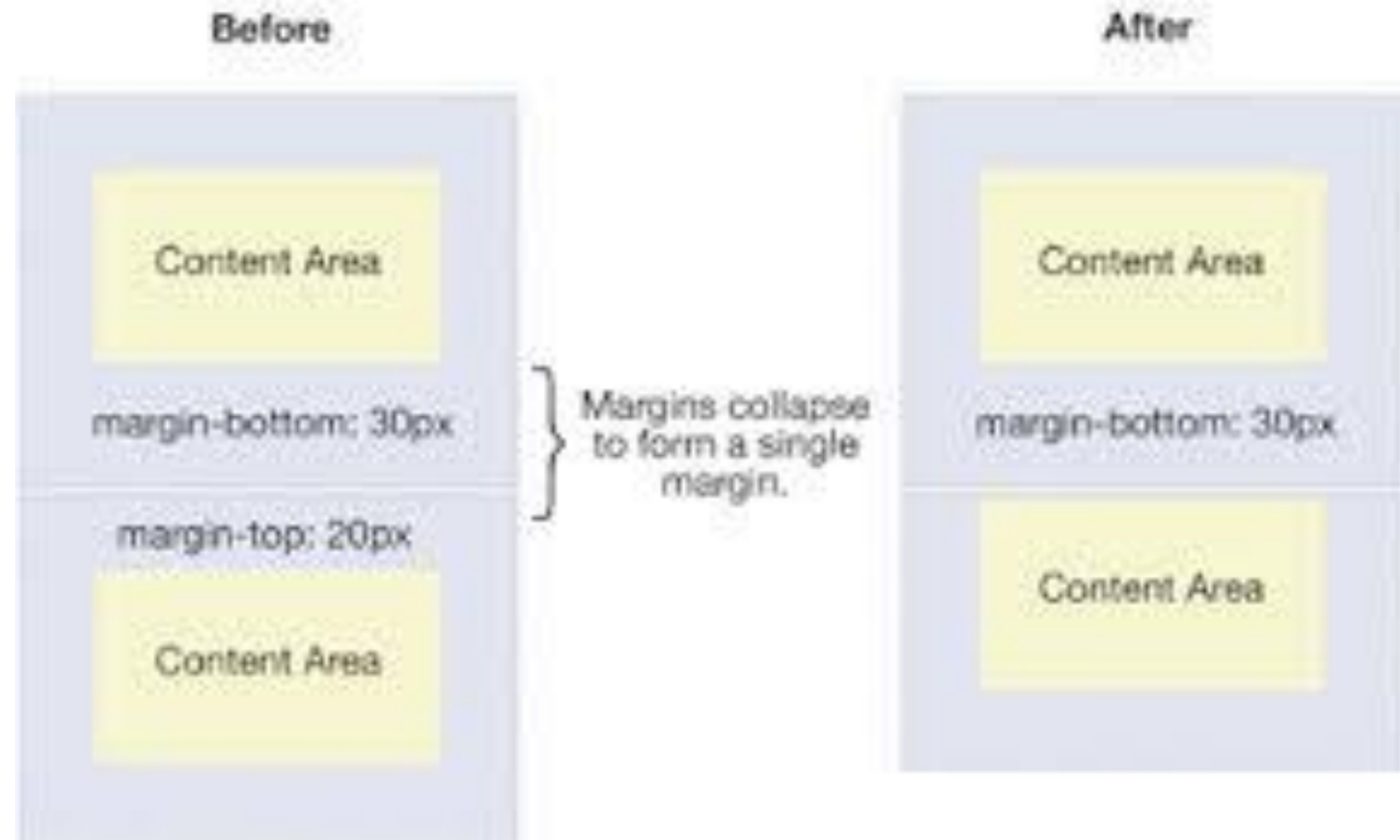
(same rule as padding for the 4 margins)

margin-top, margin-right, margin-bottom, margin-left: length measurement | percentage | auto



# Collapsing margins (Ex. 14-3)

When collapsing,  
Larger value is used.  
Some element uses  
negative margin.





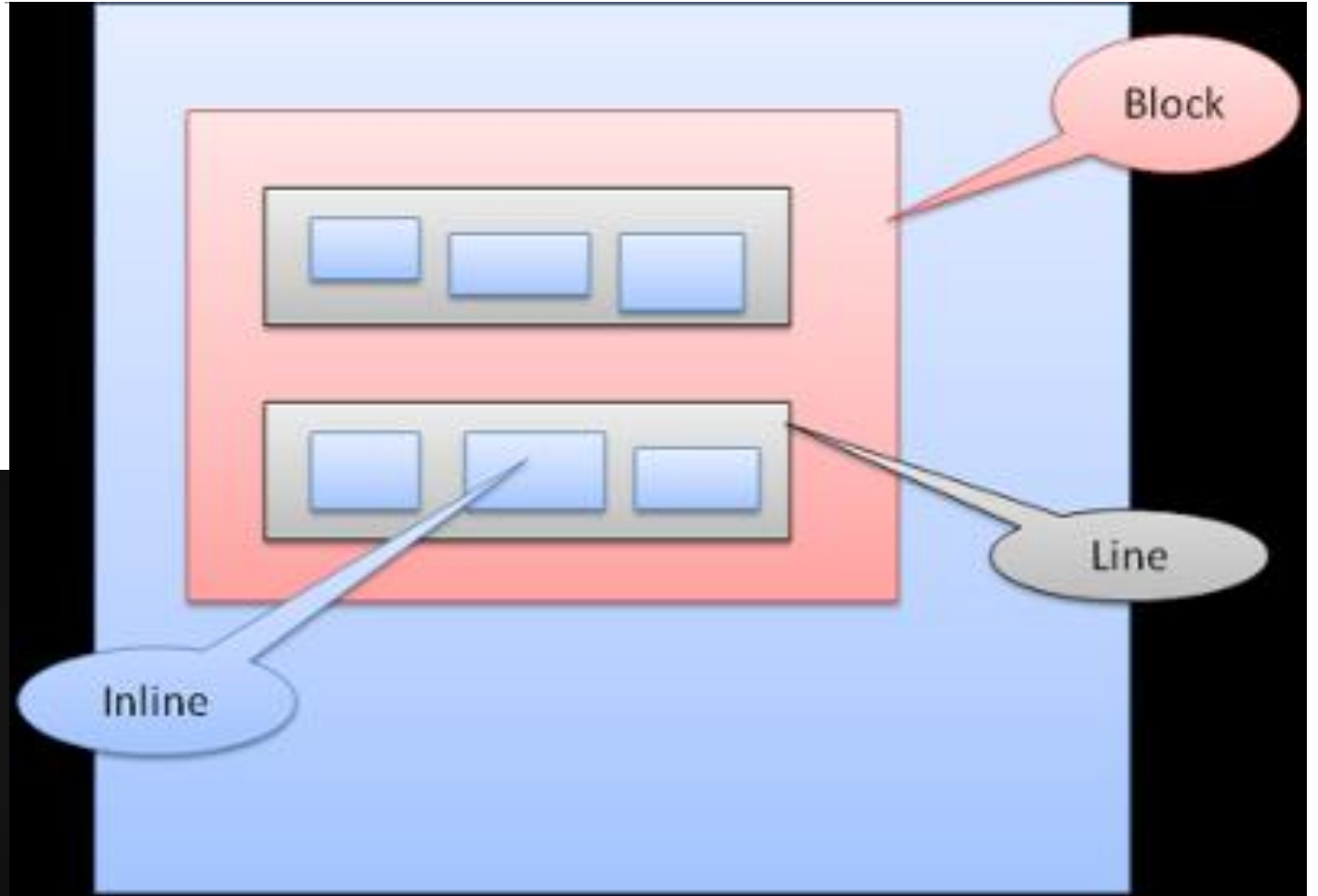
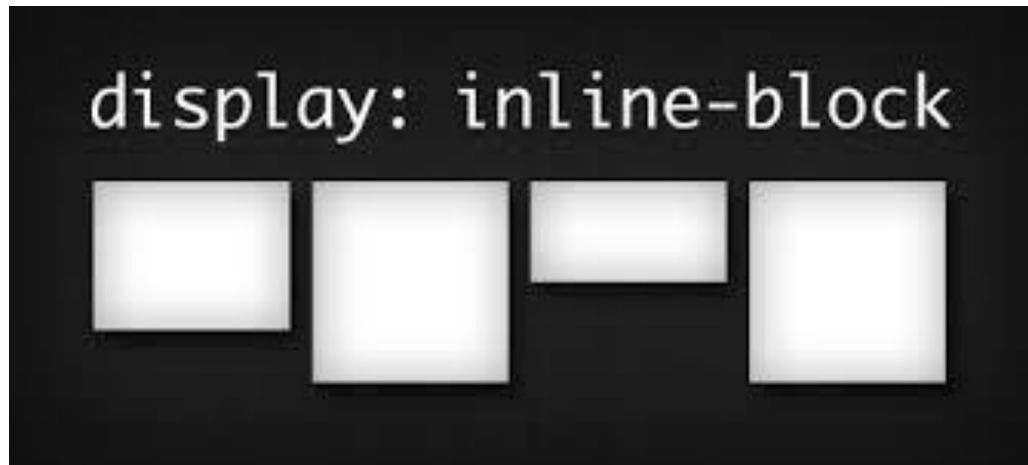
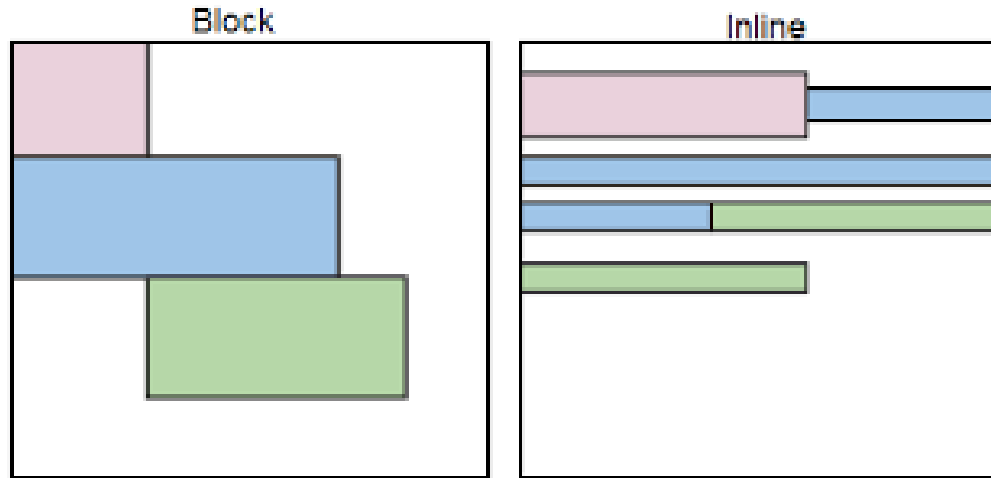
# Display Styles

---

LECTURE 5

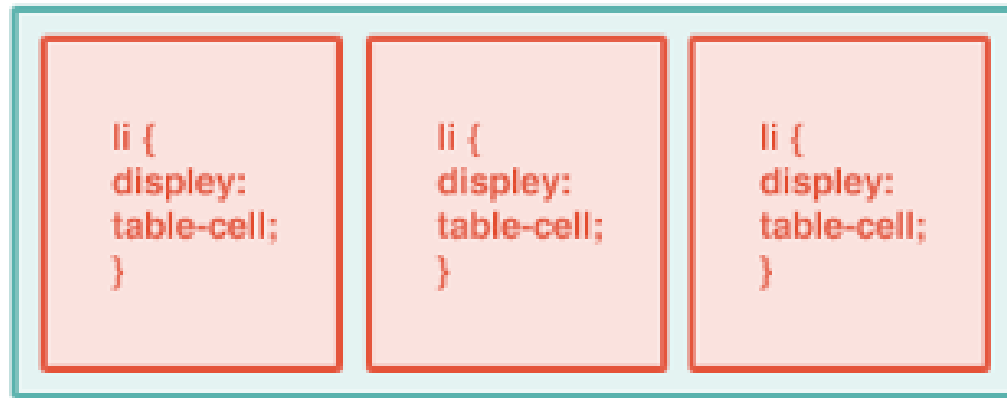
# display

(The element's role when shown.)



# display: table;

```
ul { display: table; }
```



Treat Elements like there are in a **table (like html)**: table-row-group, table-column-group, table cell, table-caption, table-row, table-footer-group, table-header-group, table-column, none,

**line** is like ul or ol in html, **inline** is like li in html, **block** is like a bigger group, **inline-block** is like a row.

**display** is used to specify the relative location of an element to other elements. **block** is a bigger area unit.

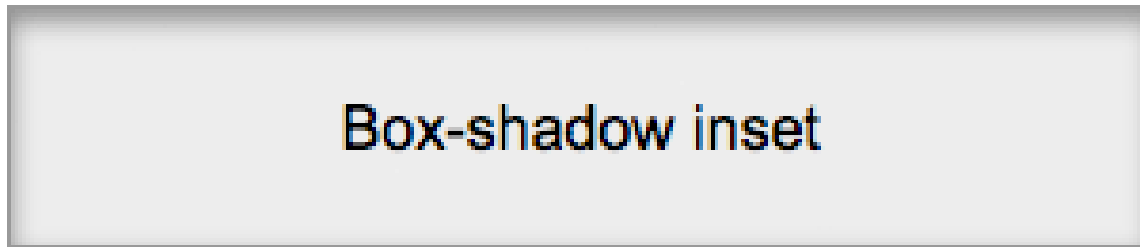
It will not be pushed into a line when it does not fit.

**display** property is only topological not logical.

# box-shadow

box-shadow: x-offset y-offset **blur spread** color inset | none

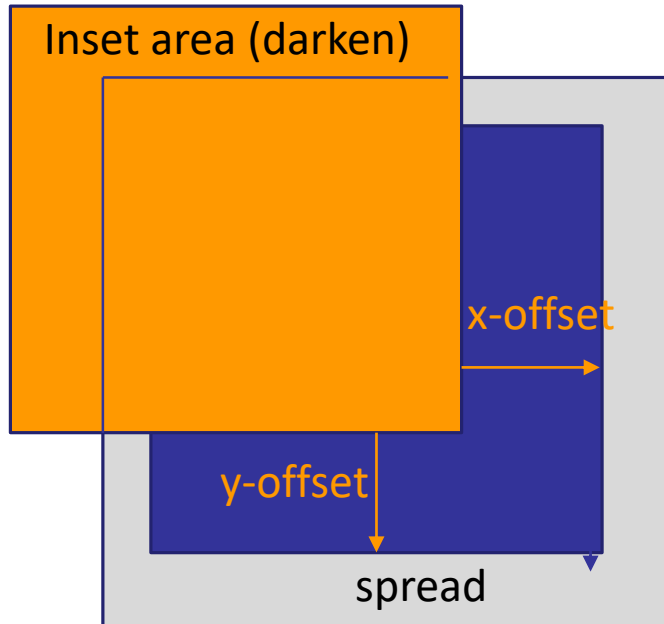
---



```
box-shadow: inset 0 3px 8px #000;
```

                  ↑   ↑   ↑      ↑  
                  X   Y   Blur  Color

# radius and shadow



inset

spread

```
border:5px solid blue;
background-color:orange;
width: 144px;
height: 144px;
```

```
border-radius: 20px;
```

```
border-radius: 0;
```

```
box-shadow:
  rgba(0,0,0,0.4)
  10px 10px;
```

```
box-shadow:
  rgba(0,0,0,0.4)
  10px 10px
  inset
```

```
box-shadow:
  rgba(0,0,0,0.4)
  10px 10px 0
  10px /* spread */
```

```
box-shadow:
  rgba(0,0,0,0.4)
  10px 10px 0
  10px /* spread */
  inset
```



```

.shadow_bottom {
  box-shadow: 0px 8px 8px -6px #333; }
.shadow_top {
  box-shadow: 0px -8px 8px -6px #333; }
.shadow_right {
  box-shadow: 8px 0px 8px -6px #333; }
.shadow_left {
  box-shadow: -8px 0px 8px -6px #333; }
.shadow_bottomright {
  box-shadow: 8px 8px 8px -6px #333; }
.shadow_bottomleft {
  box-shadow: -8px 8px 8px -6px #333; }
.shadow_topright {
  box-shadow: 8px -8px 8px -6px #333; }
.shadow_topleft {
  box-shadow: -8px -8px 8px -6px #333; }
.shadow_all {
  box-shadow: 0px 0px 8px 2px #333; }

```

