Box Model

Creating Space for Your Elements

Height and Width

- The default width of inline elements is the content.
- Elements that are not inline can take width and height properties – we saw this in the Display lecture.

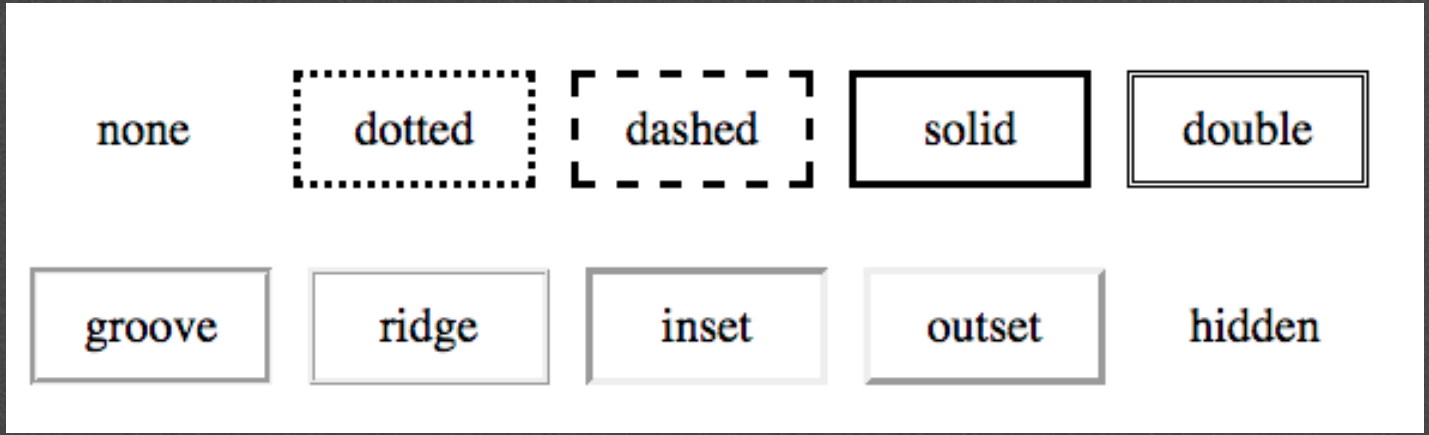
Border

- Any element can have a border around it
- border property specifies *style*, *width*, and *color*
- The border style MUST be specified

```
div {
    border: solid 1px #CC00AA;
}
```

Border-style

- none, dotted, dashed, solid, double, groove, ridge, inset, outset, hidden



Border width and color

- Width
  - Set in pixels or *thin*, *medium*, or *large*
- Color
  - Name - “blue”
  - RGB – rgb(0,0,255)
  - hex - #0000FF
  - transparent

Specifying Individual Sides

```
border-width: 3px;
```


```
border-width: 3px 10px;
```


```
border-width: 3px 10px 20px;
```


```
border-width: 3px 10px 20px 1px;
```

Margin

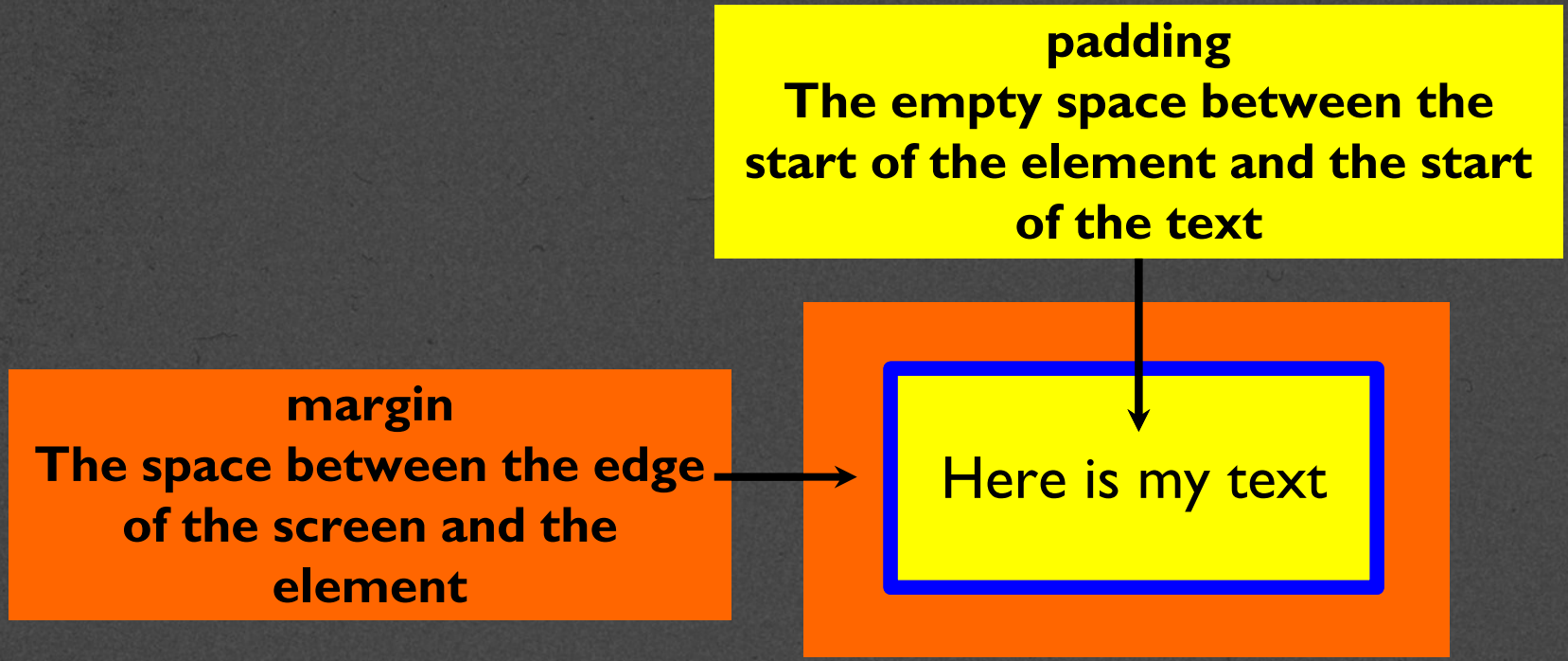
- Margin is additional space *outside* your border – between you and neighbor
- Positive margin
  - element moves right/down
- Negative margin
  - element moves left/upward

Padding

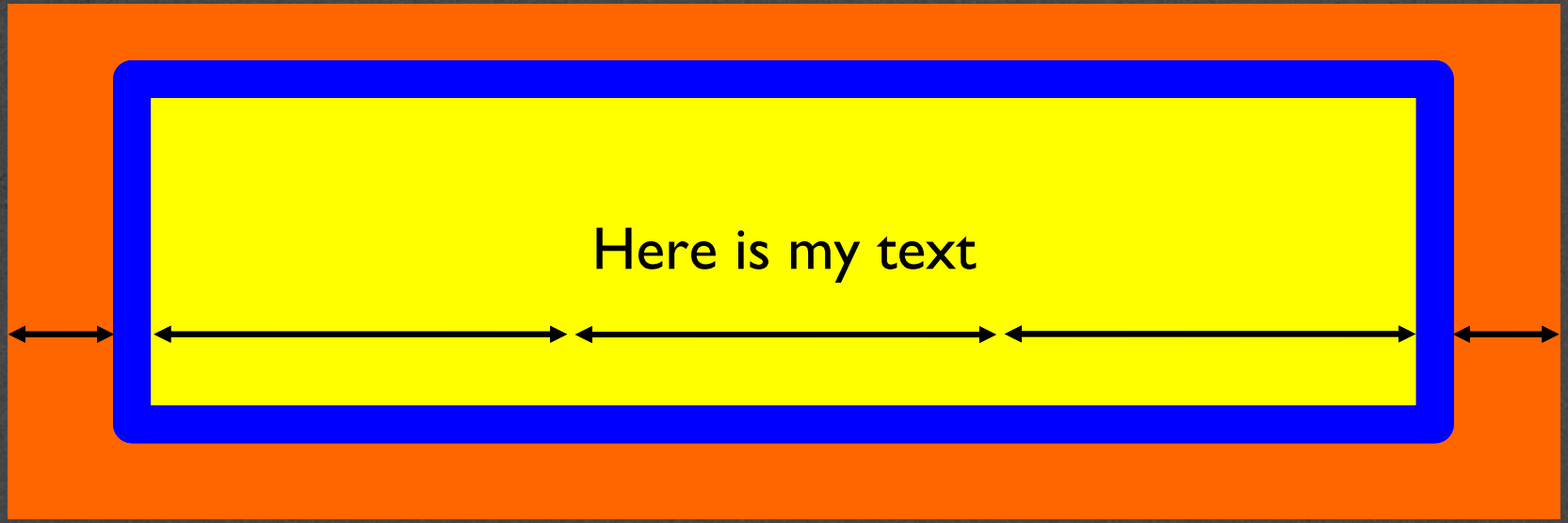
- Padding is additional space *between* the element and its border.
- Positive padding
  - border moves outward from element
- Negative padding
  - border moves over the element

Margin and Padding

- Neither takes a color (transparent)
- Can also be defined in 1 - 4 values like border


padding
The empty space between the start of the element and the start of the text
margin
The space between the edge of the screen and the element
Here is my text

Additive Height and Width



margin + border + padding + width = actual width

What is the width and height?

```
div {
    width: 100px;
    height: 50px;
    padding: 10px;
    margin: 5px;
    border: 1px solid black;
}
```

Centering an Element

- To horizontally center an element use:
  - margin: 0 auto;
- But…
  - The element must display: block
  - The element must not float
  - *The element must not have a fixed or absolute position*
  - The element must have a width that is not auto

box-sizing

- box-sizing takes some of the “math” out
- Options:
  - content-box: default additive
  - border-box: width takes content, padding, and border into consideration

Measurements

- Absolute – set to a specific size
  - px, mm, cm, pt, ….
- Fluid – sets size relative to surrounding elements
  - %, vw, vh
  - em (for font): 1em is current size, .75 is 75% of the current size
  - rem (for font): 1rem is current size of root element

Review

- Design sketches should be done with box model (margin, border, padding, content) in mind.
- Use box-model to reduce complexity
- Margin must always be considered
- Use fluid sizes for best viewing

© Colleen van Lent
University of Michigan
School of Information

Fluid Units and Overflow

Sizing and fitting your content

CSS Units

- Some measurements are fixed:
  - font-size:100pt
  - width: 500px
- Some measurements are fluid:
  - grid-template-columns: 45% 45%
  - height: 50vh
  - font-size: 2em

Overflow

- Sometimes measurements still don't work or are out of our control.
- You can use the overflow: property to control what should happen

Overflow options

- visible - Show all of the content, even if it doesn't fit.
- hidden - The content that doesn't fit will be invisible.
- scroll - Only the content that fits is visible, and a scrollbar is added to see the rest of the content
- auto - Similar to scroll, but it adds scrollbars only when necessary

Example

- Replit: Simple Grid With Overflow

Accessibility Issues

- Avoid using hidden if there is any text or important visual imagery. Just because you "think" it fits doesn't mean it will for everyone – some people magnify the page
- If you use scroll or auto you should add tabindex="0" to the html of the element you are styling so that it is still keyboard accessible.

Visibility

- Specifies whether or not element is visible
- Options include:
  - visible
  - hidden
  - collapse (only for table elements)
- Unlike display:none a hidden element is still part of the DOM and still takes up space

Review

- When setting the size, overflow and visibility of your elements, remember that not everyone accesses your site the same.
- Think about what parts of your site have content, vs which parts are just decorative.

University of Michigan
School of Information

Positioning

Positioning!

- Putting elements where you want them can be time-consuming and frustrating
- Why not tables?

Position Properties

- The four position properties are:
  - static
  - relative
  - absolute
  - fixed
- Position properties are modified by the properties: top, right, bottom, left

Static

- Default value for elements
- Place in the next available position
- Not affected by the top, bottom, left, and right properties.

Relative

- Positioned "relative to itself"
- Take the static position, but add offsets.
- The new positioning does NOT affect any other element. It is possible to move an element and leave a big hole where it would have been.
- Relatively positioned elements are often used as container blocks for absolutely positioned elements.

Absolute

- Element is removed from the document flow and positioned relative to its *nearest ancestor* (or the root)
- Other elements behave as if element does not exist
- Can end up on top of another element

Fixed Position

- Positioned relative to the *browser window*
- Will not move, even if the window is scrolled
  - IE7 and IE8 support the fixed value only if a !DOCTYPE is specified
- Think of popup boxes that won't go away!!!
- Or a navigation bar that is always visible on the top

Z-index

- Multiple elements may be placed in the same position.
- z-index is a numeric value, positive or negative that dictates stacking order

Review

- Positioning elements is key to achieving desired layouts
- Proper planning will make this easier

© Colleen van Lent
University of Michigan
School of Information