FEWD Class 4

Review

- Shorthand Properties
- Classes and IDs
- Floats
- Centering

Centering Things Horizontally

- When placed on the parent element, works for **inline elements**:

```
text-align: center;
```

- When placed on the element, setting the left and right margins to auto works for **block elements**:

```
margin-left: auto;
margin-right: auto;
```

Centering Things Vertically

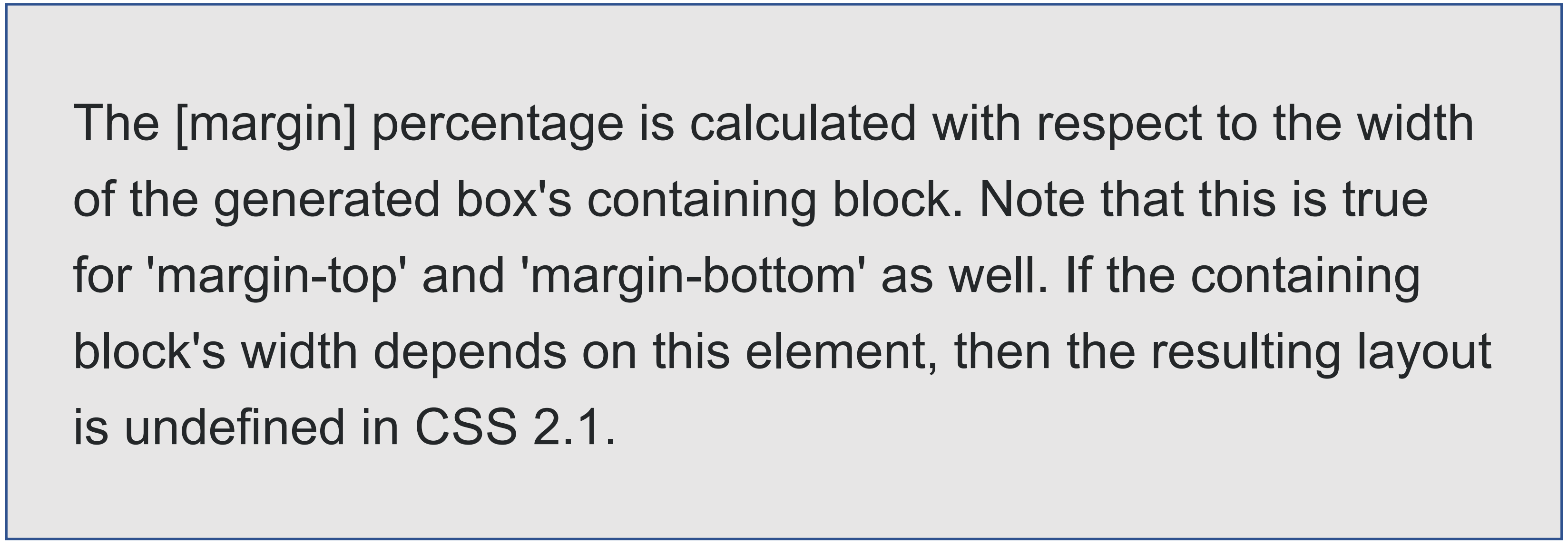
- If you know the **height** of the element you want to center and its parent’s height, you can use margin or padding, such as:

```
margin-top: calc(50vh – 150px);
```

- In this way you can calculate the position in CSS by subtracting the half the height of the element you want to position from half the height of the parent.

Why can’t we just use 50%?

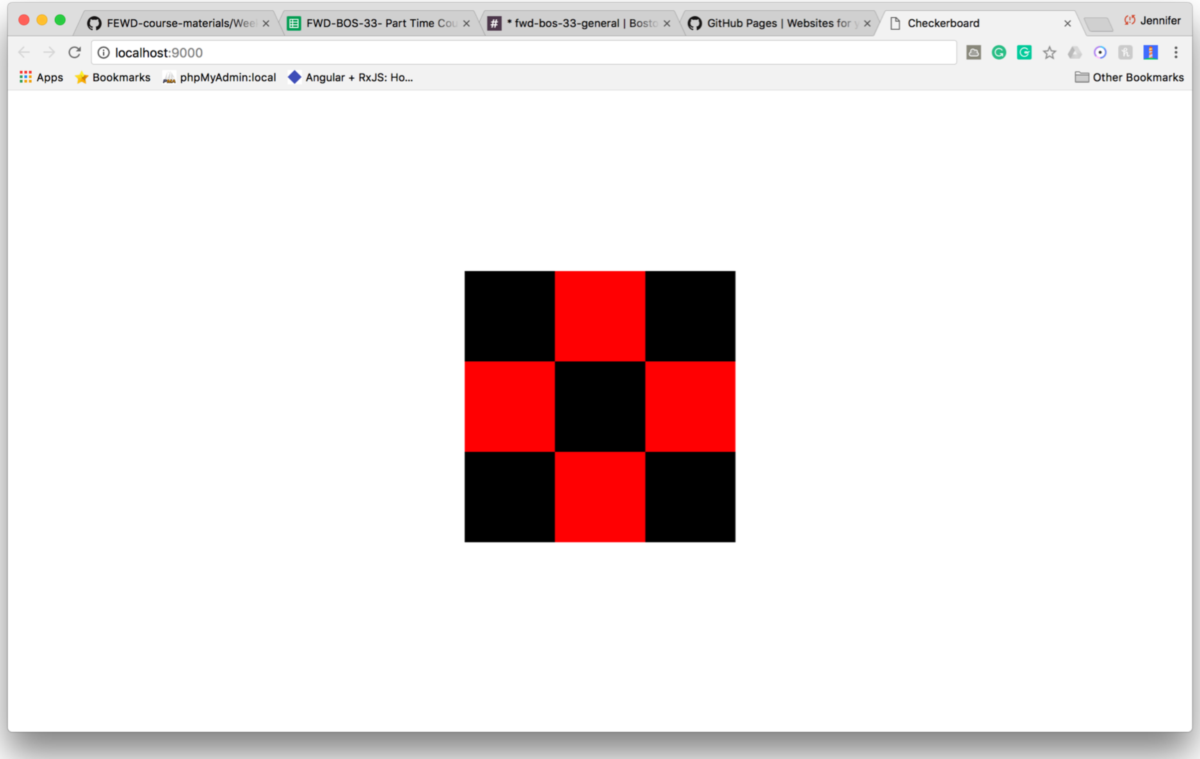
- Because for top and bottom margins and paddings in percentages, the values will be taken as the **fractional width** of the parent element, **not the height**.



The [margin] percentage is calculated with respect to the width of the generated box's containing block. Note that this is true for 'margin-top' and 'margin-bottom' as well. If the containing block's width depends on this element, then the resulting layout is undefined in CSS 2.1.

Lab

- Practice what we learned last class
- Reproduce the following checkerbox design
- It should be 300px x 300px
- Hints:
  - You'll need to use some classes
  - You should center the board on the screen horizontally and vertically
  - You want to use floats
- **BONUS: can you do it with only five black divs on a red board?**


FEWD-course-materials/Wee
FWD-BOS-33- Part Time Cou
* fwd-bos-33-general | Bosto
GitHub Pages | Websites for y
Checkerboard
Jennifer
localhost:9000
Apps
Bookmarks
phpMyAdmin:local
Angular + RxJS: Ho...
Other Bookmarks

Objectives

- Centering Things
- Positioning in CSS
- CSS Pseudo Classes

Positioning

- The position property specifies a method for positioning elements
- The top, right, bottom, and left properties determine the position for the element (except elements with static position).
- There are five different position values:
  - static
  - relative
  - fixed
  - absolute
  - sticky

Position: static

- Static positioning is the default.
- Static positioned elements are in the ***normal flow*** of the document.
- The top, right, bottom and left values and z-index properties have ***no effect***.

Position: relative

- Relative positioned elements are positioned in the ***normal flow*** of the document, but offset by the top, right, bottom and left values.
- You can think of this as ***relative to itself***.
- Because the positioned element **doesn’t** change the position of other elements, it creates a **stacking order**.
- You can specify the stacking order with z-index.

Position: fixed

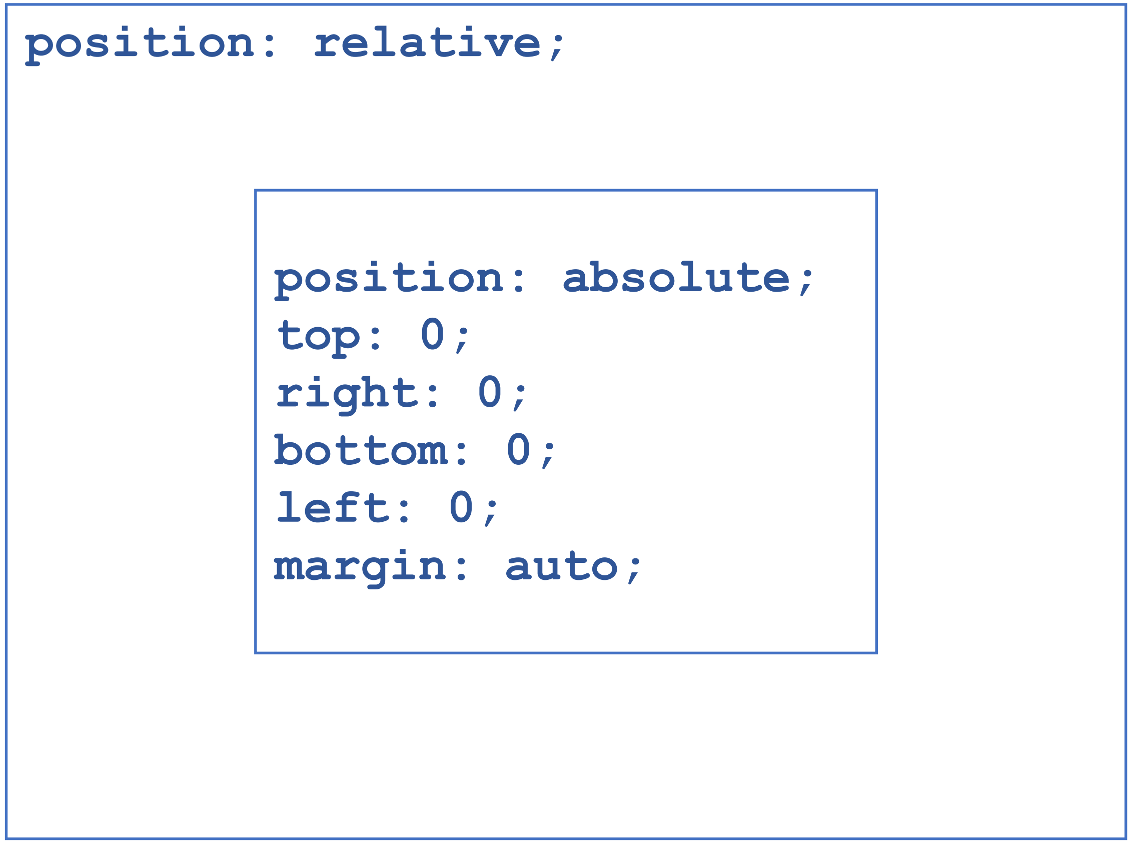
- Fixed positioned are removed from the normal flow of the document.
- Because they are removed from the flow, all of the other elements will reflow to fill the space of the fixed positioned element.
  - It’s like it was never there.
- Fixed positioned elements are positioned based on the viewport window.
- Fixed elements will always stay positioned even if the page is scrolled.

Position: absolute

- Absolute positioned are removed from the normal flow of the document.
- Because they are removed from the flow, all of the other elements will reflow to fill the space of the absolute positioned element.
    - It’s like it was never there.
- Absolute positioned elements are positioned based on their nearest ancestor with positioning specified other than static.
- If no ancestor has positioning, the element is positioned according to the root.

Centering with absolute positioning

- The inner element needs a height set
- Remember to give the outside element a position
- Set the inside element to position: absolute
- Set its top, right, left, and bottom to 0
- Set its margin to auto

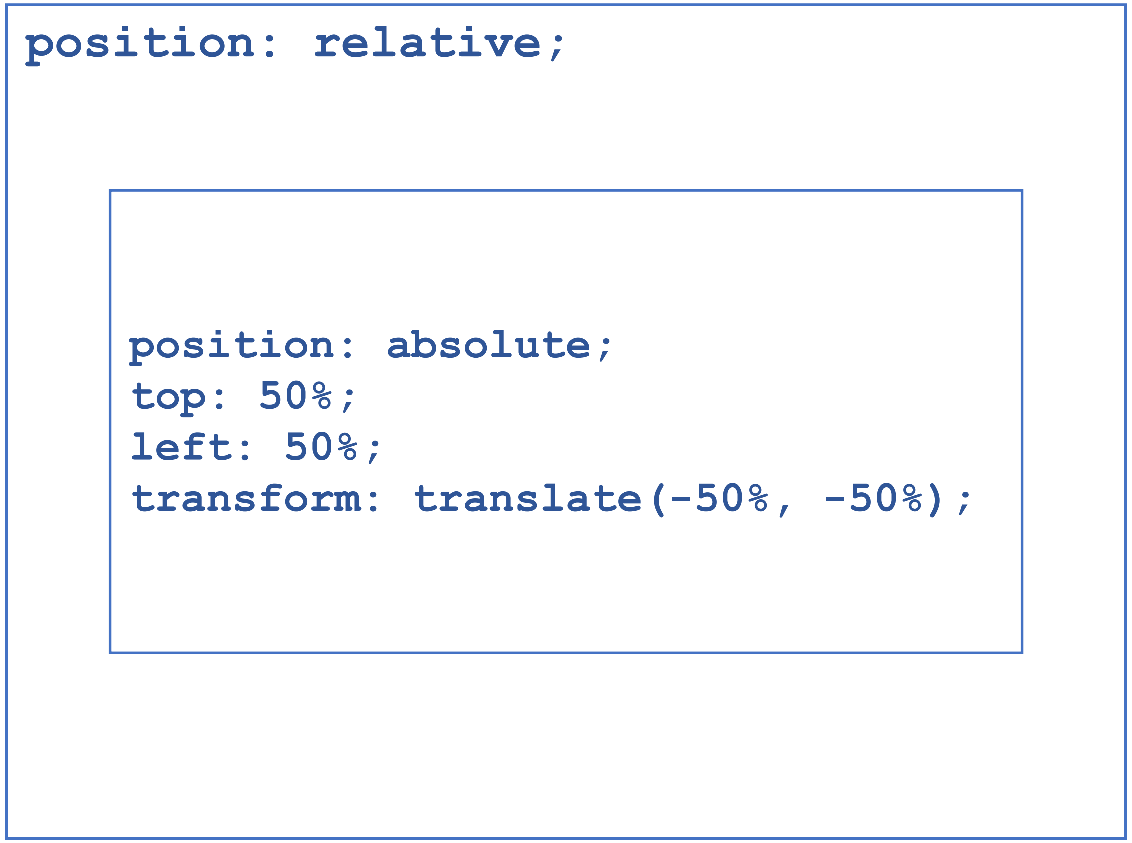


```
position: relative;
```

```
position: absolute;
top: 0;
right: 0;
bottom: 0;
left: 0;
margin: auto;
```

What if I don’t know the height?

- Remember to give the outside element a position
- Set the inside element to position: absolute
- Set its top and left properties to 50%;
- Use transform to shift it 50% of its own width and height up and to the left



```
position: relative;
```

```
position: absolute;
top: 50%;
left: 50%;
transform: translate(-50%, -50%);
```

Position: sticky

- The element is positioned in the normal flow initially.
- It is offset relative to *its flow root and* *containing block*, including table-related elements, based on the values of top, right, bottom, and left.
- The offset does not affect the position of any other elements.

Sticky means what?!?

- Sticky positioned elements are relative positioned until crossing a specified threshold
- After the threshold it acts like a fixed position element until it reaches the boundary of its parent.

```
#sticky { position: sticky; top: 10px; }
```

- This element would scroll along in the normal flow and then when it was 10px from the top of the screen, it would become fixed in place.

Lab

- Build the Relaxr Landing Page

Pseudo Classes

:active

:any

:any-link

:checked

:default

:defined

:dir()

:disabled

:empty

:enabled

:first

:first-child

:first-of-type

:fullscreen

:focus

:focus-visible

:host

:host()

:host-context()

:hover

:indeterminate

:in-range

:invalid

:lang()

:last-child

:last-of-type

:left

:link

:not()

:nth-child()

:nth-last-child()

:nth-last-of-type()

:nth-of-type()

:only-child

:only-of-type

:optional

:out-of-range

:read-only

:read-write

:required

:right

:root

:scope

:target

:valid

:visited

Link Pseudo Classes

```
/* unvisited link */
a:link {
    color: #FF0000;
}

/* visited link */
a:visited {
    color: #00FF00;
}

/* mouse over link */
a:hover {
    color: #FF00FF;
}

/* selected link */
a:active {
    color: #0000FF;
}
```