WEEKLY OVERVIEW

WEEK 7	Animations / Interactions Lab
WEEK 8	Responsive Design / Final Project Lab
WEEK 9	Interactions Lab / Students' Choice

CSS POSITIONING & ANIMATION

AGENDA

- Review
- CSS Positioning
- CSS Transitions
- CSS Transforms

LEARNING OBJECTIVES

- Differentiate between various CSS Positioning techniques.
- Utilize transitions and transforms to add basic animations on hover

HOMEWORK REVIEW

HOMEWORK — GROUP DISCUSSION

EXERCISE

TYPE OF EXERCISE

- Groups of 2-3

TIMING

10 min

1. Pick someone to take notes for your group.
2. Share 1 thing you're excited about being able to accomplish. Focus on the positives!
3. Have each person in the group note 1 thing they found challenging for the assignment and make note. Discuss as a group how you think you could solve that problem.
4. Discuss the bonus tasks (Best of 3 feature, refactoring the Temp Converter). If anyone in your group was able to tackle one of these, share!

ADVANCED CSS POSITIONING

ACTIVITY — POSITIONING

KEY OBJECTIVE

- Differentiate between various positioning techniques.

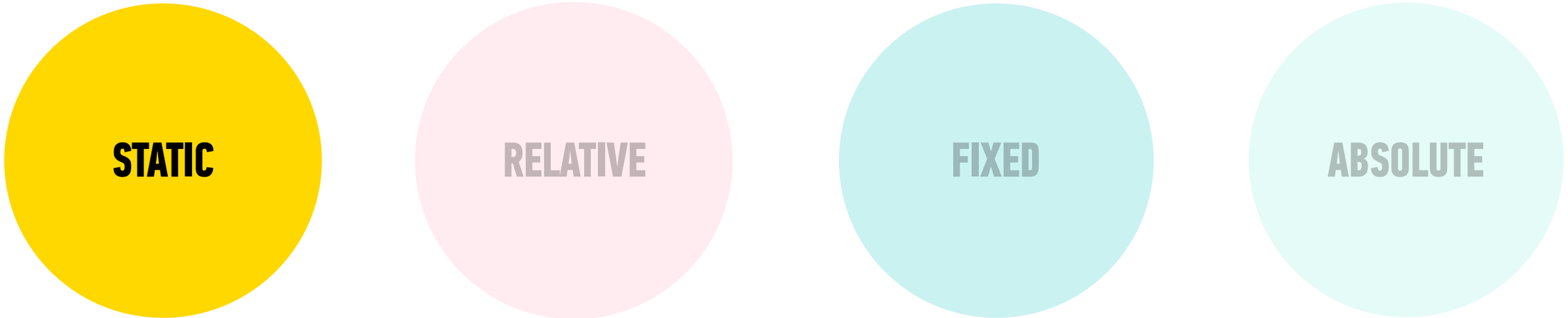
TYPE OF EXERCISE

- Groups of 2 - 3

TIMING

4 min 1. Complete steps 1 - 3 in `positioning_101`

CSS POSITIONING

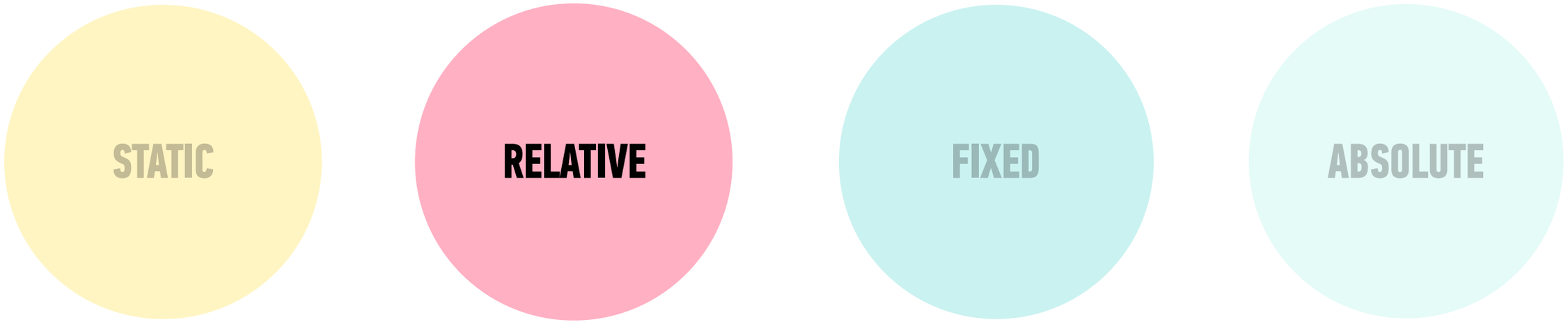


STATIC POSITIONING

- Default positioning
- Normal flow of the document
- Elements render in order, as they appear in the document flow.

```
.my-class {
  position: static;
}
```

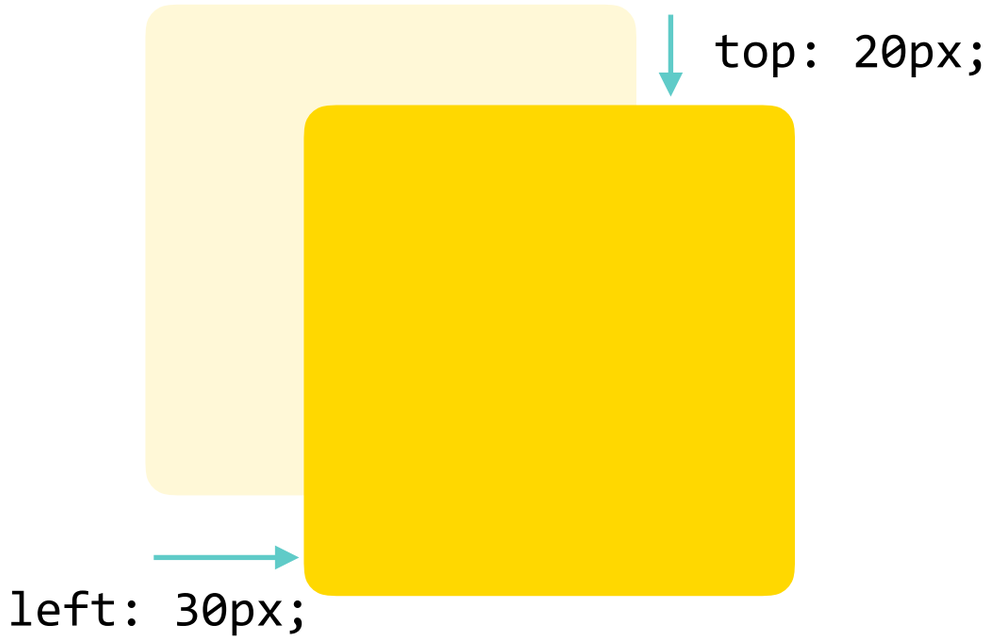
CSS POSITIONING



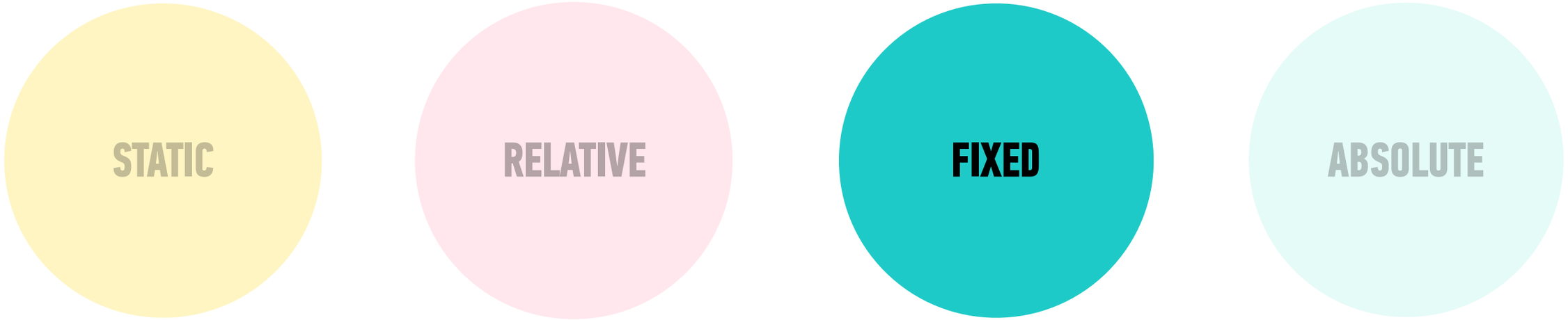
RELATIVE POSITIONING

- Moves an element *relative to where it would have been in normal flow*.
- For example: `left: 30px` adds 30px to an element's **left** position

```
.my-class {
  position: relative;
  top: 20px;
  left: 30px;
}
```



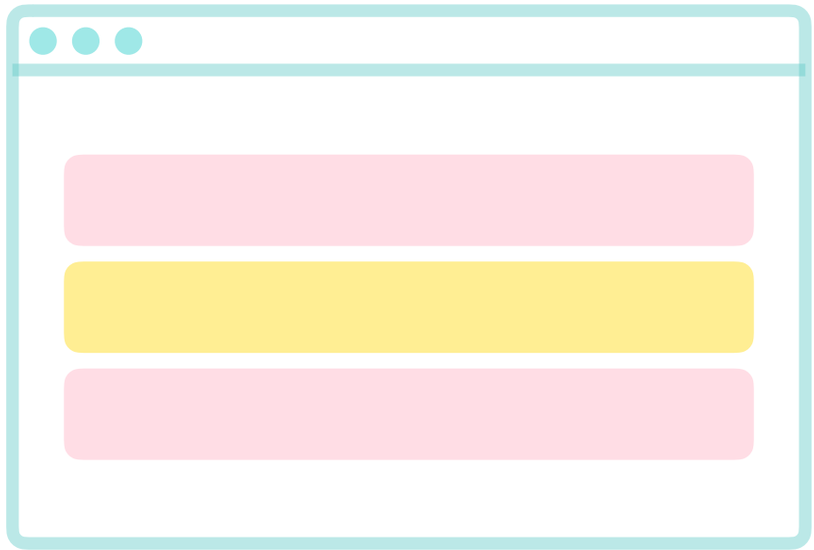
CSS POSITIONING

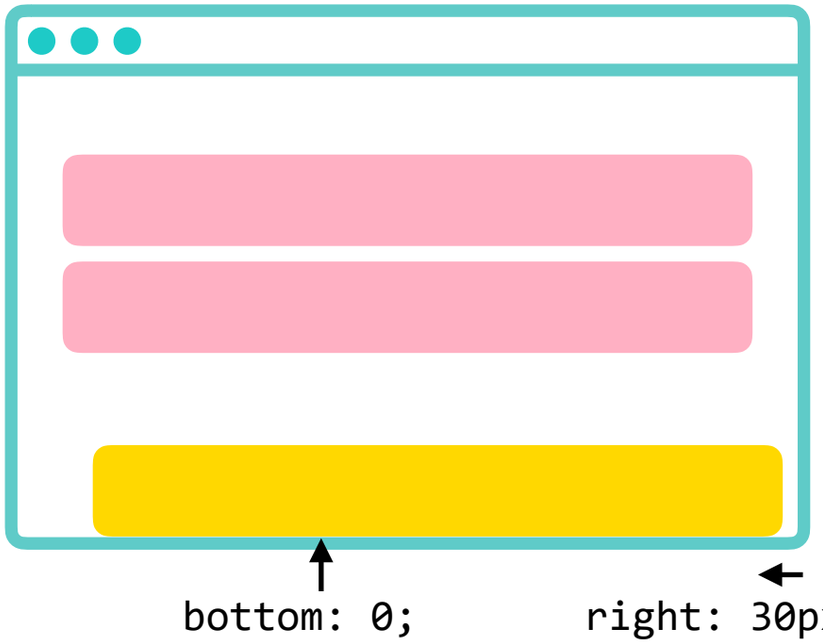


FIXED POSITIONING

- Positioned in relation to *the browser window*
- When the user scrolls, it stays in the same place.
- Use *right*, *top*, *left* and *bottom* properties to position the element in relation to the browser window.

```
.my-class {
  position: fixed;
  bottom: 0;
  right: 30px;
}
```



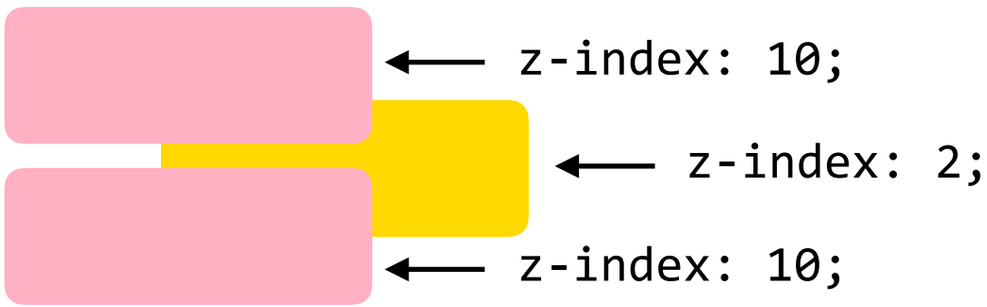


OVERLAPPING ELEMENTS — Z-INDEX

- With relative, absolute, and fixed positioning, elements can overlap.
- We can use `z-index` to control which elements are layered on top of each other.
- This property takes a number — the higher the number the closer that element is to the front.

```
.yellow {
  z-index: 2;
}

.pink {
  z-index: 10;
}
```



Think of this like 'bring to front' and 'send to back' in programs like Adobe Illustrator.

ACTIVITY

EXERCISE

KEY OBJECTIVE

- Practice using CSS positioning

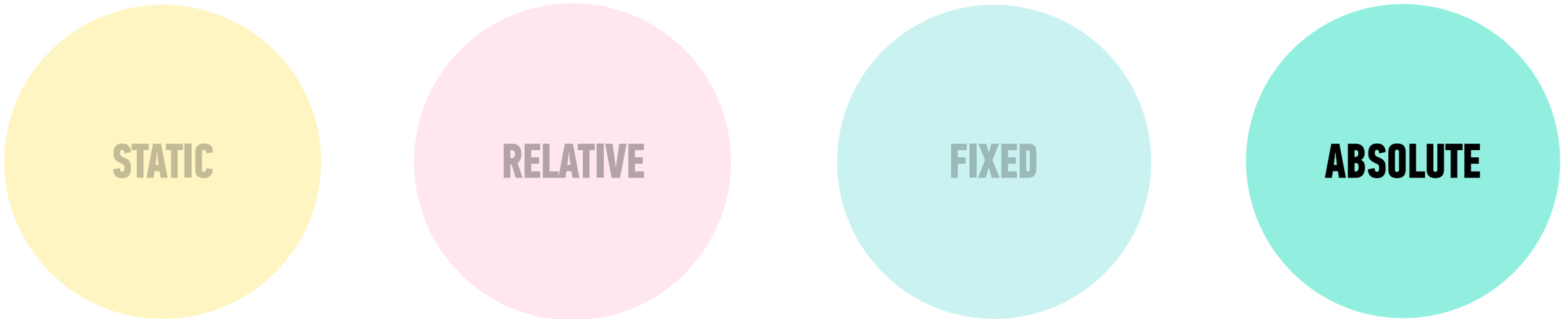
LOCATION

- Starter Code > CSS_Positioning > creepy_crawlers

TIMING

8 min 1. Follow step 1 in main.css

CSS POSITIONING



ACTIVITY — POSITIONING

EXERCISE

KEY OBJECTIVE

- Differentiate between various positioning techniques.

TYPE OF EXERCISE

- Groups of 2 - 3

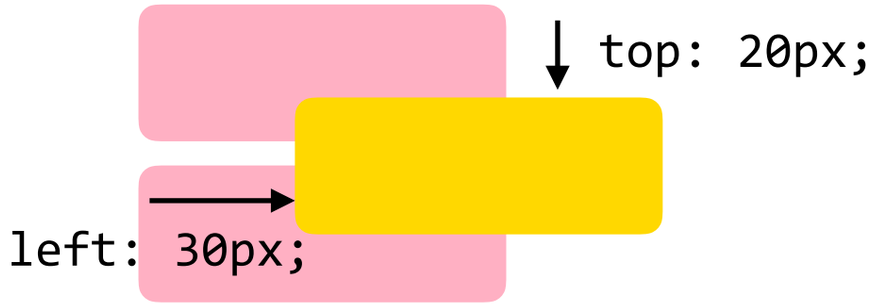
TIMING

4 min 1. Complete steps 4A and 4B in `positioning_101`

ABSOLUTE POSITIONING

- Element is taken out of the normal flow of the document.
- No longer affects the position of other elements on the page (they act like it's not there).
- You can add the *right*, *top*, *left* and *bottom* properties to specify where the element should appear

```
.my-class {
  position: absolute;
  top: 20px;
  left: 30px;
}
```

POSITIONING THINGS ABSOLUTELY

- When using `position: absolute`, *top*, *bottom*, *left* and *right* values will be relative to the element's *closest ancestor that has any position other than static*.

To position an element absolutely:

1. Set `position: relative` on ancestor element
2. Set `position: absolute` on child element and use top, right, bottom and left values to position.

Parent we want child to be positioned relative to

```
<section>
    <div class="info"></div>
</section>
```

Child we want to position

POSITIONING THINGS ABSOLUTELY

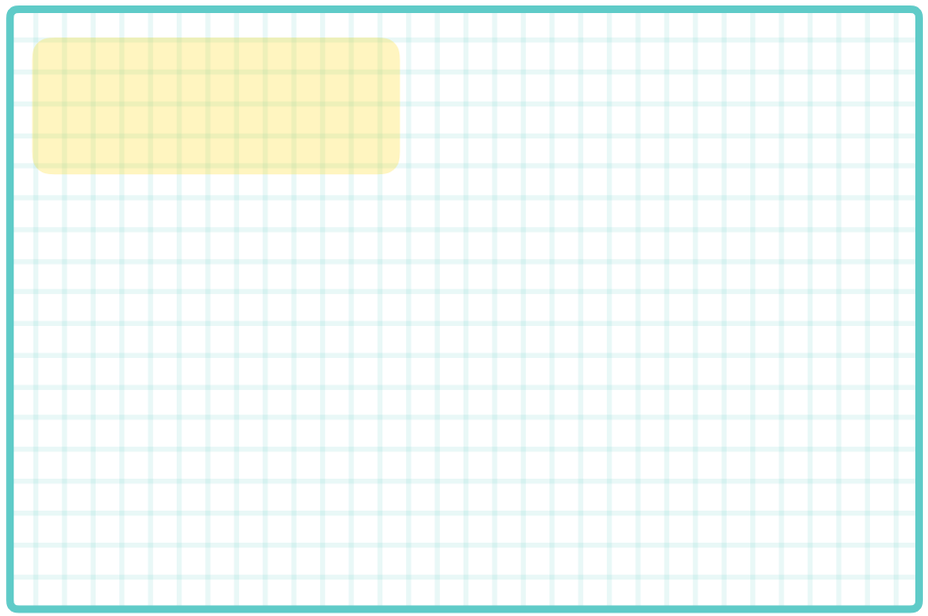
- When using `position: absolute`, *top*, *bottom*, *left* and *right* values will be relative to the element's *closest ancestor that has any position other than static*.

To position an element absolutely:

1. Set `position: relative` on ancestor element
2. Set `position: absolute` on child element and use top, right, bottom and left values to position.

```
<section>
   <div class="info"></div>
</section>
```

```
section {
  position: relative;
}
```



POSITIONING THINGS ABSOLUTELY

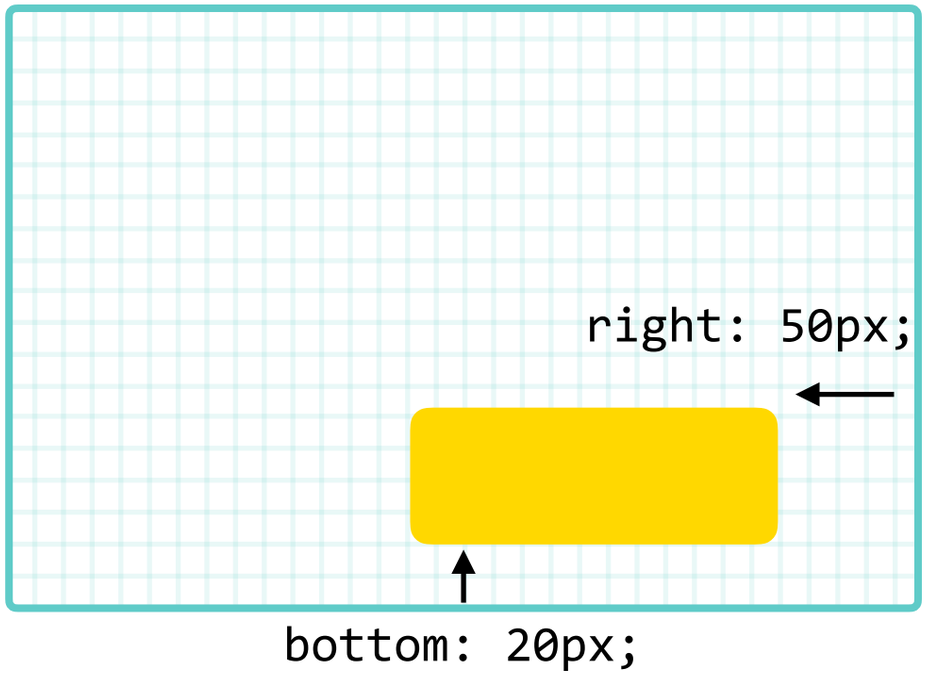
- When using `position: absolute`, *top*, *bottom*, *left* and *right* values will be relative to the element's *closest ancestor that has any position other than static*.

To position an element absolutely:

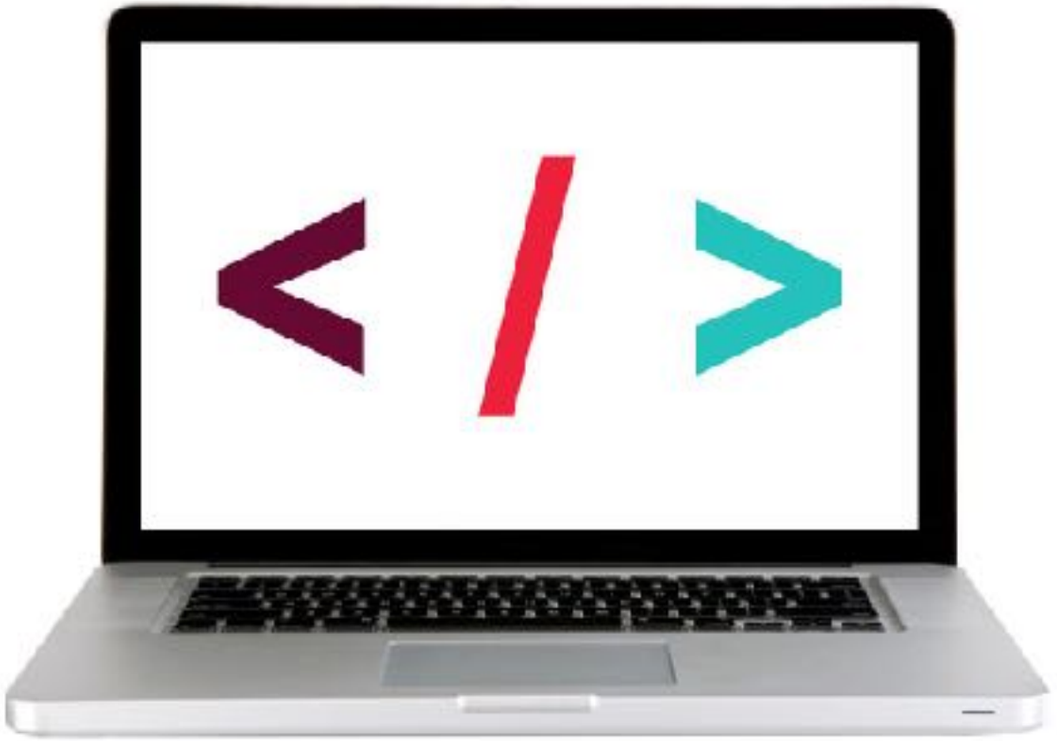
1. Set `position: relative` on ancestor element
2. Set `position: absolute` on child element and use top, right, bottom and left values to position.

```
<section>
   <div class="info"></div>
</section>
```

```
.info {
  position: absolute;
  right: 50px;
  bottom: 20px;
}
```



LET'S TAKE A CLOSER LOOK



starter_code_lesson_13 > positioning_group

WANT TO LEARN MORE?

Resources for more info/examples:

- A List Apart: CSS Positioning 101

ACTIVITY

EXERCISE

KEY OBJECTIVE

- Differentiate between various positioning techniques.

TYPE OF EXERCISE

- Turn and Talk

TIMING

4 min

1. When would you want to use absolute positioning? Can you think of an example?
2. When would you want to use relative positioning?
3. How about fixed?

TRANSITIONS

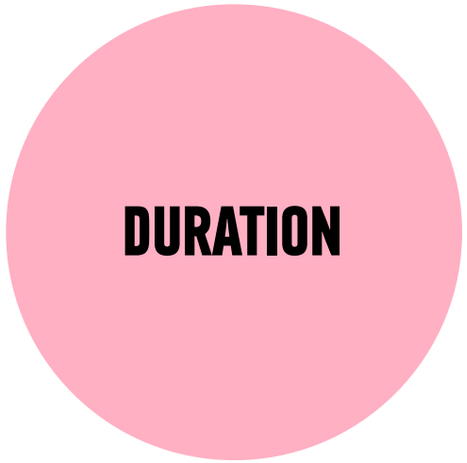
TRANSITIONS

- Provide a way to control animation speed when changing properties
- Instead of having property changes take effect immediately, you can have them take place over a period of time.

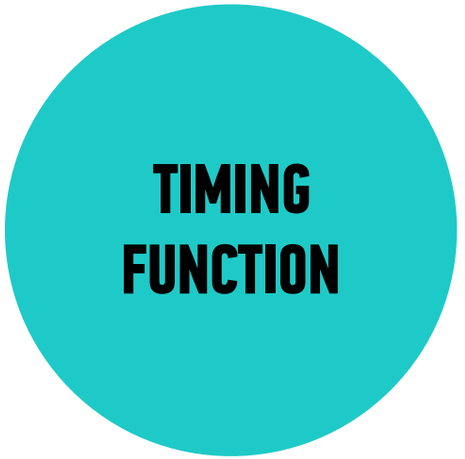
```
.example {
  transition: [transition-property] [transition-duration] [transition-timing-function] [transition-delay];
}
```

TRANSITIONS

Which properties to animate



How long the transition will last



How the transition will run

When the animation will start

TRANSITIONS - TRANSITION-PROPERTY

PROPERTIES TO ANIMATE

- Can specify a specific property to transition or "all" to transition all properties
- *Default:* all

```
div {
 transition: opacity 0.5s;
}
```

```
div {
 transition: all 0.5s;
}
```

```
div {
 transition: height 0.5s;
}
```

```
.example {
 transition: [transition-property] [transition-duration] [timing-function] [transition-delay];
}
```

TRANSITIONS - TRANSITION-DURATION

DURATION

- A time value, defined in seconds or milliseconds

```
div {
 transition: all 0.5s;
}
```

```
div {
 transition: all 350ms;
}
```

```
div {
 transition: all 3s;
}
```

```
.example {
 transition: [transition-property] [transition-duration] [timing-function] [transition-delay];
}
```

TRANSITIONS

TIMING FUNCTION

- Describes how a transition will proceed over its duration, allowing a transition to change speed during its course.
- Timing functions: ease, linear, ease-in, ease-out, ease-in-out

```
div {
 transition: opacity 0.5s ease;
}
```

```
div {
 transition: opacity 0.5s ease-in-out;
}
```

```
.example {
 transition: [transition-property] [transition-duration] [timing-function] [transition-delay];
}
```

Interactive graph of transition timing functions

TRANSITIONS

DELAY

- Length of time before the transition starts

```
div {
 transition: background-color 0.5s ease 2s;
}
```

```
.example {
 transition: [transition-property] [transition-duration] [timing-function] [transition-delay];
}
```

MORE FUN WITH TRANSITIONS — CODROPS

Fun CSS button styles: Creative buttons

Icon hover effects: Icon Hover Effects

Modal dialogue effects (advanced): Dialogue Effects

ACTIVITY — BUTTON LAB

EXERCISE

KEY OBJECTIVE

- Practice using CSS transitions

TYPE OF EXERCISE

- Individual/Partner Lab

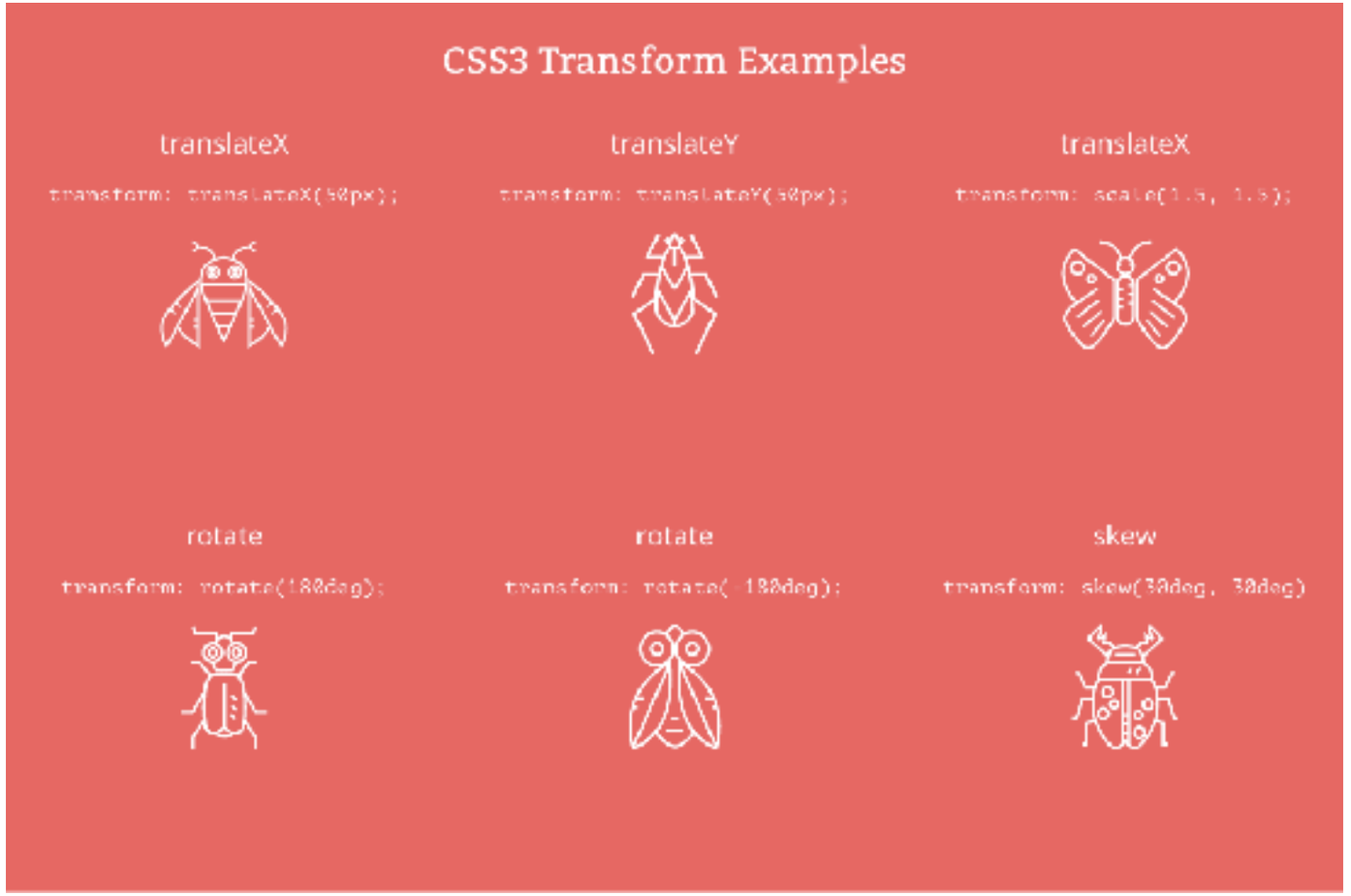
TIMING

6 min

1. In `transition_button_lab`, add :hover styles and transition to the button

TRANSFORMS

LET'S TAKE A CLOSER LOOK — TRANSFORM



Syntax: [CSS Tricks]

ACTIVITY — TRANSFORM ON TIMER

KEY OBJECTIVE

- Practice using CSS transitions

TYPE OF EXERCISE

- Individual/Partner Lab

TIMING

10 min

- Follow the instructions in starter code > transform_bug > css > style.css
- You'll want to use CHROME to test this!

LEARNING OBJECTIVES

- Differentiate between various CSS Positioning techniques.
- Utilize transitions and transforms to add basic animations on hover

WEEKLY OVERVIEW

WEEK 7	Animations / Interactions Lab
WEEK 8	Responsive Design / Final Project Lab
WEEK 9	Interactions Lab / Students' Choice

EXIT TICKETS!