CSS POSITIONING

WEEKLY OVERVIEW

WEEK 4	Responsive Design / CSS Positioning
WEEK 5	Forms / Final Project Lab
WEEK 6	Intro to JS / Functions

AGENDA

- CSS Positioning
- CSS Transitions
- CSS Transforms
- Wireframes

LAB

LEARNING OBJECTIVES

- Use the `position` property to position elements on the page
- Utilize transitions and transforms to add basic animations on hover

REVIEW

SLACK RESPONSE

- What are some common updates we might want to make when making a page responsive?
- What are media queries? What can we change inside media queries?
- Where should we add media queries? Why is this order important?
- I have the two following media queries. How should these be rearranged? Can anything be removed?

```
@media screen and (max-width: 400px) {
     .hamburger {
         display: block;
     }
     section {
         grid-template-columns: repeat(1, 1fr);
     }
}

@media screen and (max-width: 600px) {
     .hamburger {
         display: block;
     }
     section {
         grid-template-columns: repeat(2, 1fr);
     }
}
```

CLASSES

- Classes are used to group elements together so that those elements can be targeted to add styles

```
<div class="alert">Content</div>

 .alert {
    color: red;
    font-size: 20px;
 }
```

CLASSES AND IDS

To add multiple classes to an element, use a space-separated list:

```
<div class="alert success">Content</div>
```

The classes can then be used separately in the CSS file to add styles:

```
.alert {
   border: 2px solid black;
   font-size: 20px;
}
.success {
   background-color: red;
}
```

https://codepen.io/sarahholden/pen/aVdLmG?editors=1100

ADVANCED CSS POSITIONING

ACTIVITY — POSITIONING

EXERCISE

KEY OBJECTIVE

- Differentiate between various positioning techniques.

TYPE OF EXERCISE

- Groups

TIMING

4 min

1. Complete steps 1 - 4B in `positioning_intro`
2. Bonus: If you finish early, look up "z-index CSS". What does this property do? Write a summary in Slack.

STATIC POSITIONING

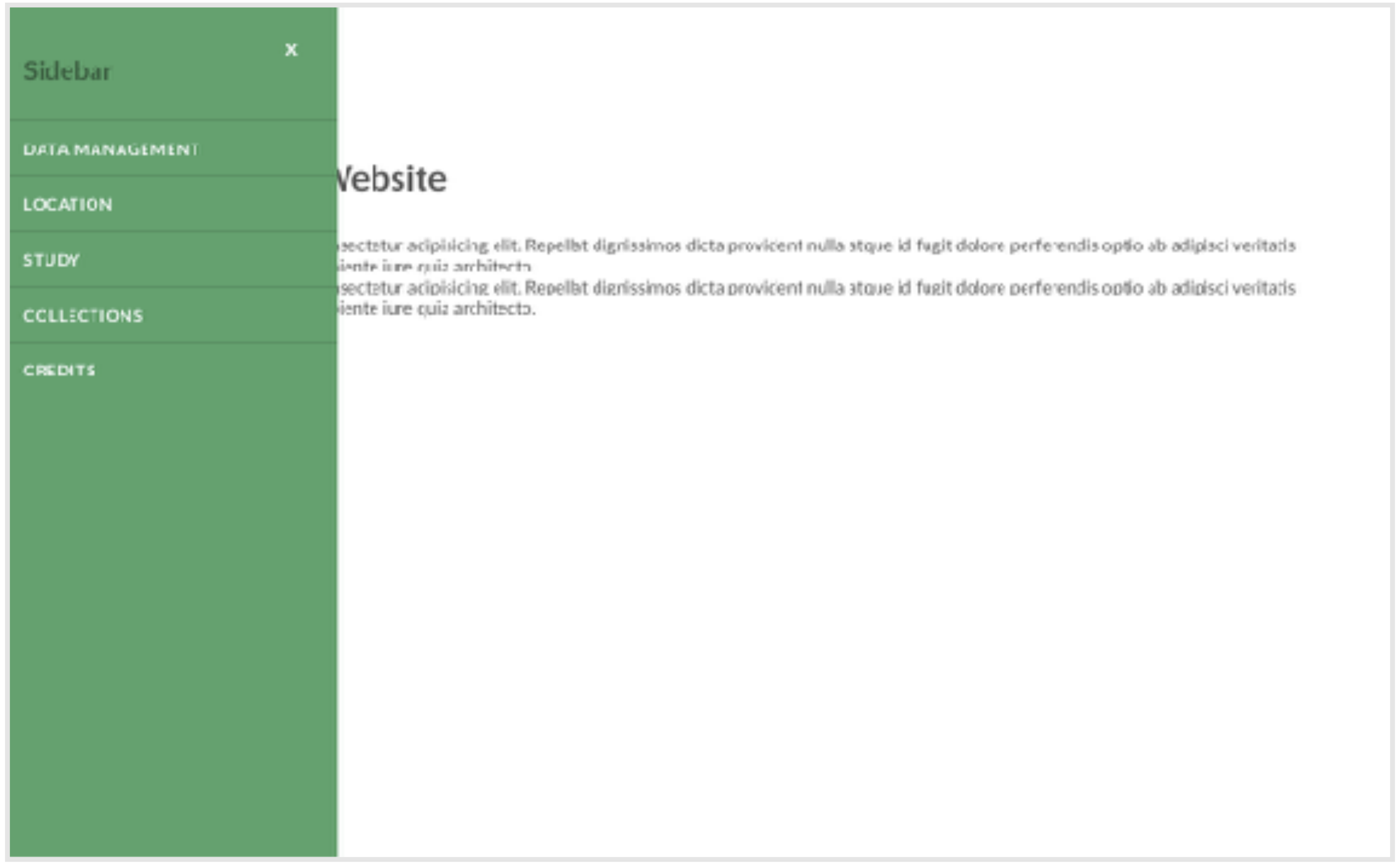
- By default, elements on a page are similar to these wooden blocks.
- They will stack one on top of the other in the same order that they are placed in an HTML file. This is referred to as the "normal flow" of a document.

STATIC POSITIONING

- We can use the `position` property in our CSS to take elements out of the normal flow of the document and specify where they should appear.

```
elementSelector {
   position: yourPositionHere;
}
```

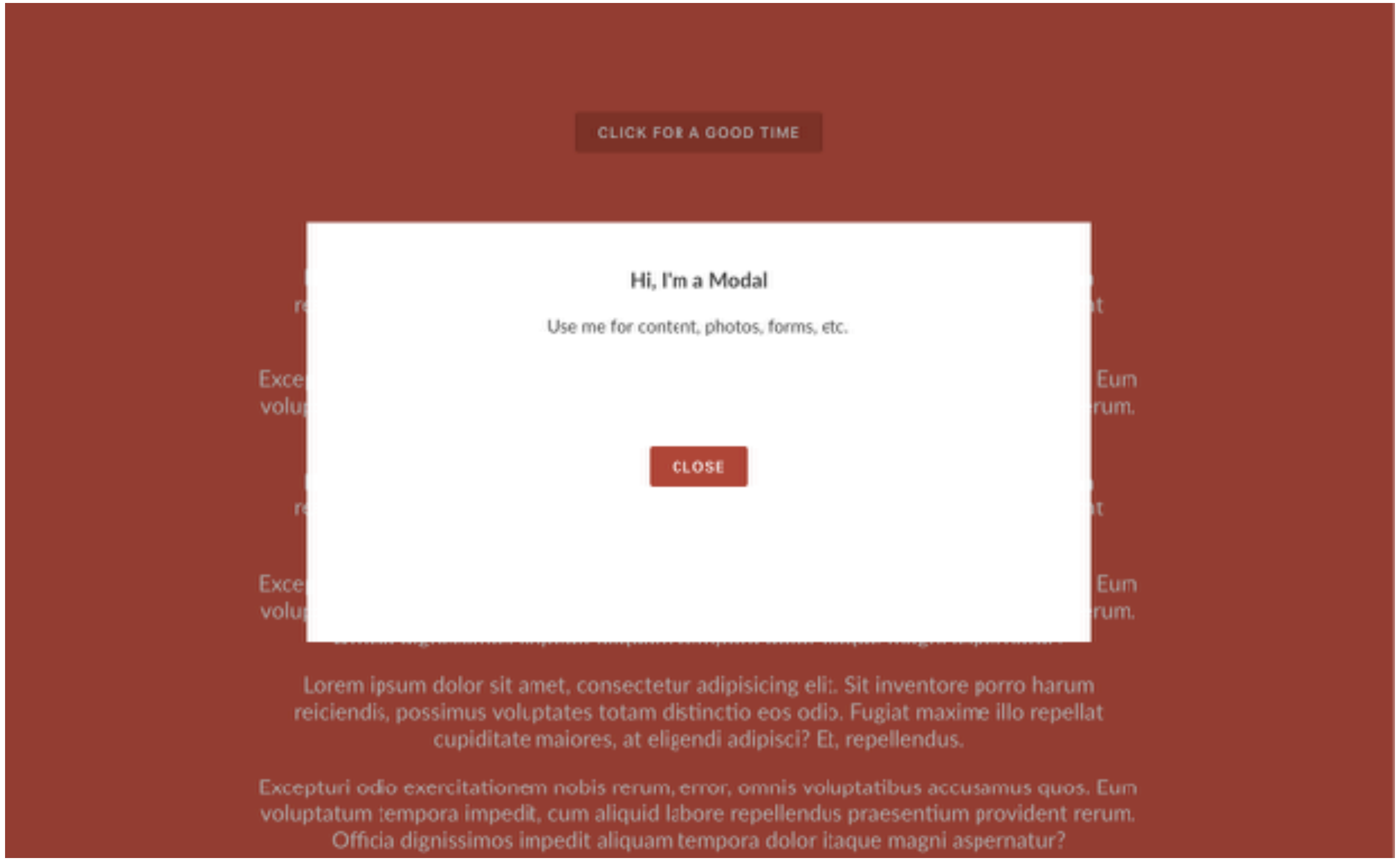
CSS POSITIONING — SIDEBAR



Vebsite

ectetur adipisicing elit. Repellat dignissimos dicta provicent nulla atque id fugit dolore perferendis optio ab adipisci veritatis iente iure quia architecto.
ectetur adipisicing elit. Repellat dignissimos dicta provicent nulla atque id fugit dolore perferendis optio ab adipisci veritatis iente iure quia architecto.

CSS POSITIONING — MODAL WINDOW



CLICK FOR A GOOD TIME

Hi, I'm a Modal

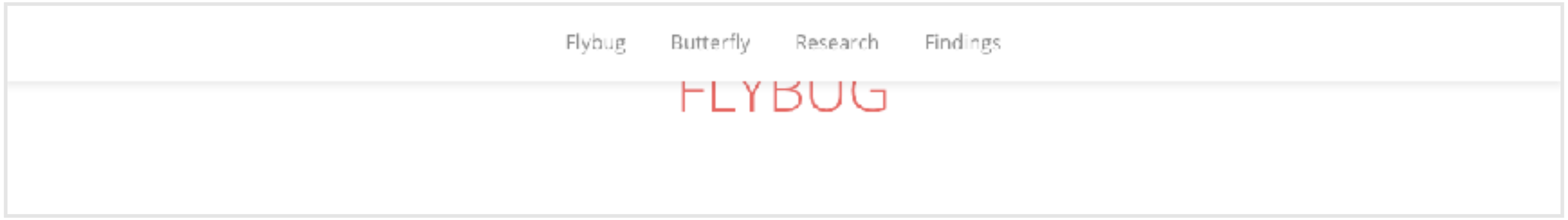
Use me for content, photos, forms, etc.

CLOSE

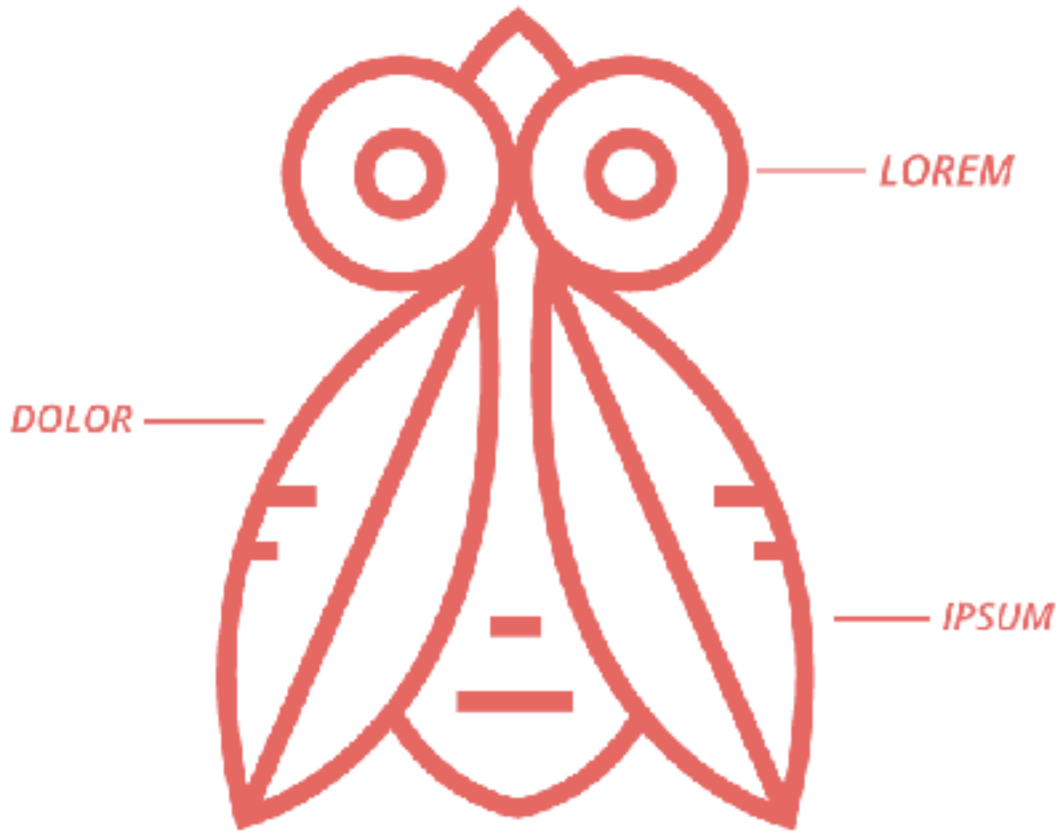
Lorem ipsum dolor sit amet, consectetur adipisicing elit. Sit inventore porro harum reiciendis, possimus voluptates totam distinctio eos odio. Fugiat maxime illo repellat cupiditate maiores, at eligendi adipisci? Et, repellendus.

Excepturi odio exercitationem nobis rerum, error, omnis voluptatibus accusamus quos. Eum voluptatum tempora impedit, cum aliquid labore repellendus praesentium provident rerum. Officia dignissimos impedit aliquam tempora dolor itaque magni aspernatur?

CSS POSITIONING — STICKY NAV



CSS POSITIONING — LABELS FOR IMAGE



FIXED POSITIONING

CSS POSITIONING

FIXED

STICKY

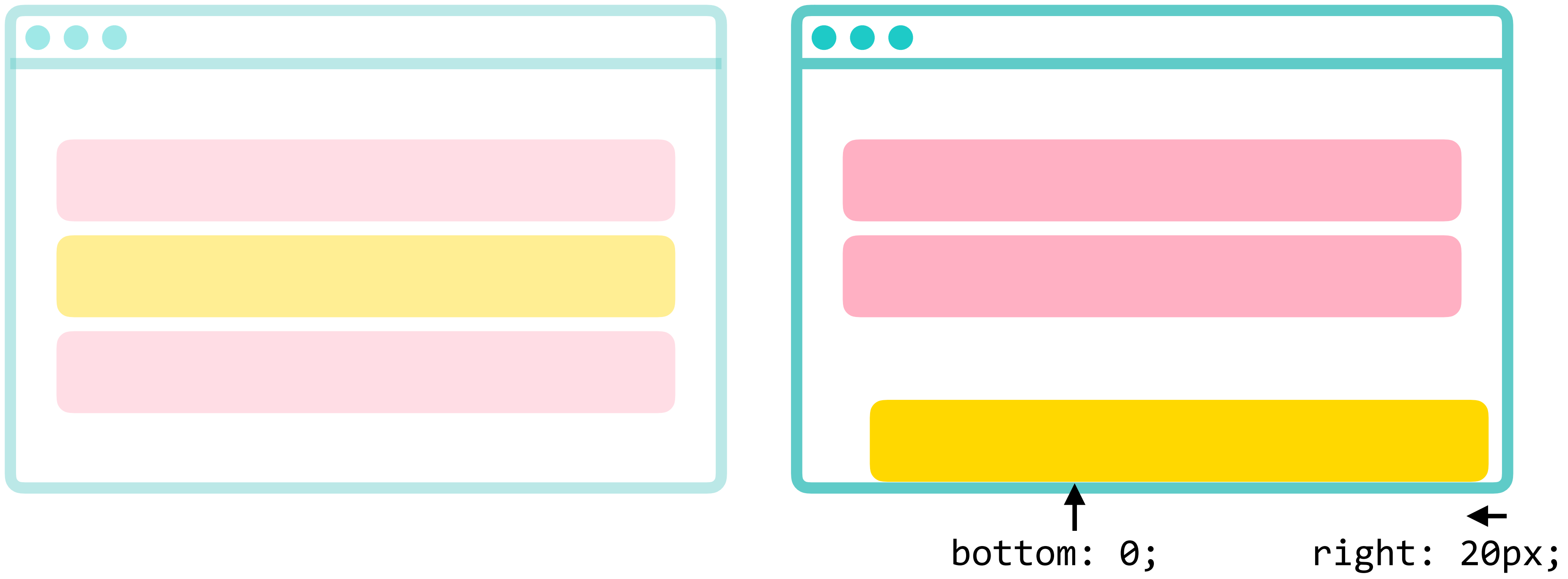
RELATIVE

ABSOLUTE

FIXED POSITIONING

- When the user scrolls, it stays in the same place.
- Use **`right`**, **`top`**, **`left`** and/or **`bottom`** properties to specify where the element should go *in relation to the browser window.*

```
yourSelectorHere {
   position: fixed;
   bottom: 0;
   right: 20px;
}
```



CSS POSITIONING

FIXED

STICKY

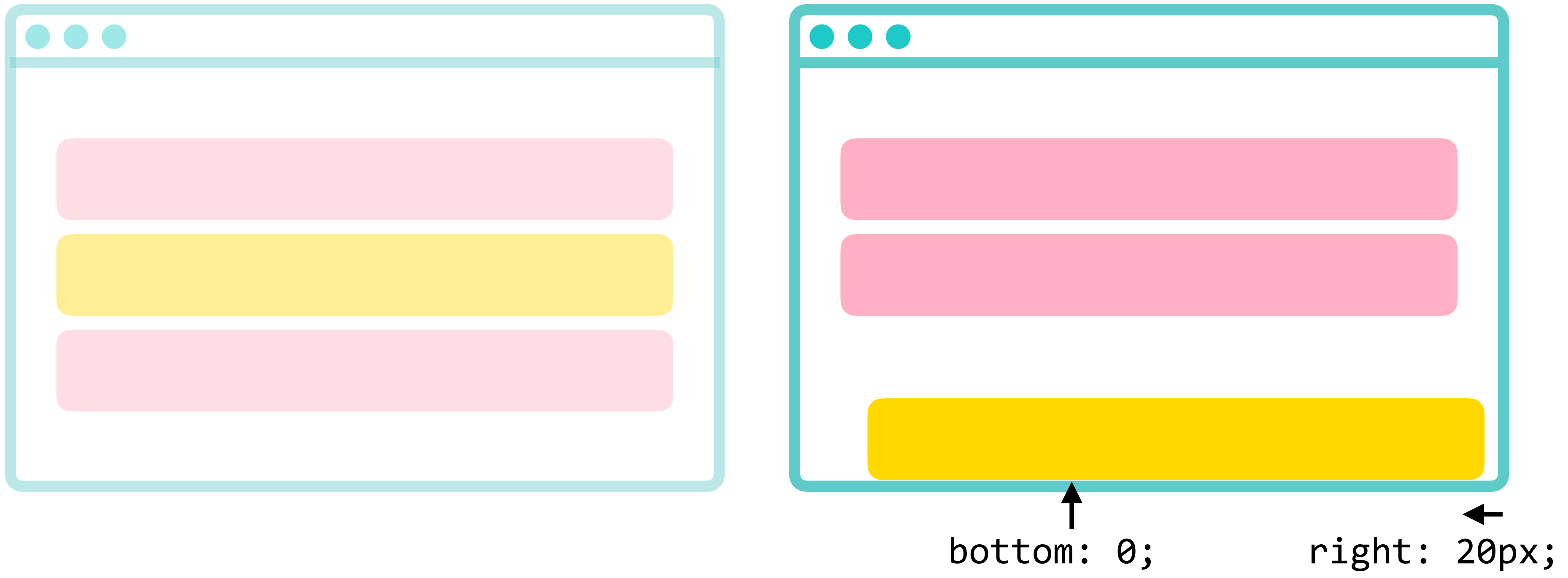
RELATIVE

ABSOLUTE

FIXED POSITIONING

- Similar to fixed position, this time the element will only "stick" when it reaches the specified **top** value.
- A value for **top** is *required* or it won't work!!.

```
yourSelectorHere {
    position: fixed;
    top: 0;
}
```



ANIMATION

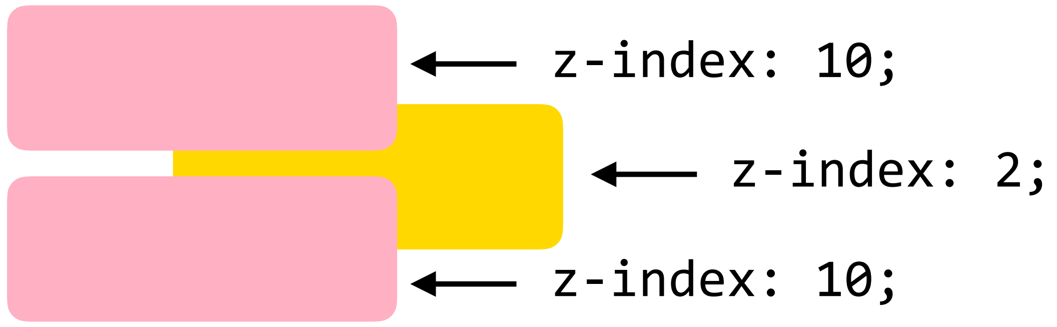
Z-INDEX

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 — FIXED NAV

EXERCISE

KEY OBJECTIVE

- Practice using CSS positioning

LOCATION

- Starter Code > positioning_practice

TIMING

8 min

1. Follow step 1 in main.css
2. After getting fixed position to work, try changing the position from `fixed` to `sticky`!

RELATIVE POSITIONING

CSS POSITIONING

FIXED

STICKY

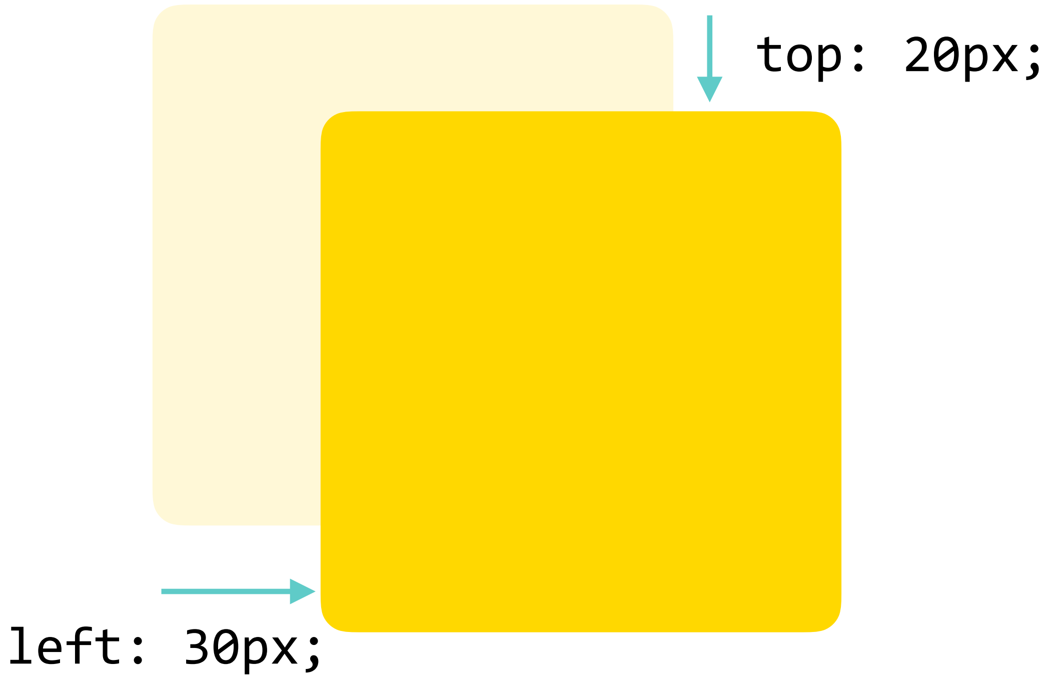
RELATIVE

ABSOLUTE

RELATIVE POSITIONING

- Moves an element *relative to where it would have been in normal flow*.
- For example: `left: 20px` adds 20px to an element's **left** position

```
yourSelectorHere {
    position: relative;
    top: 20px;
    left: 30px;
}
```



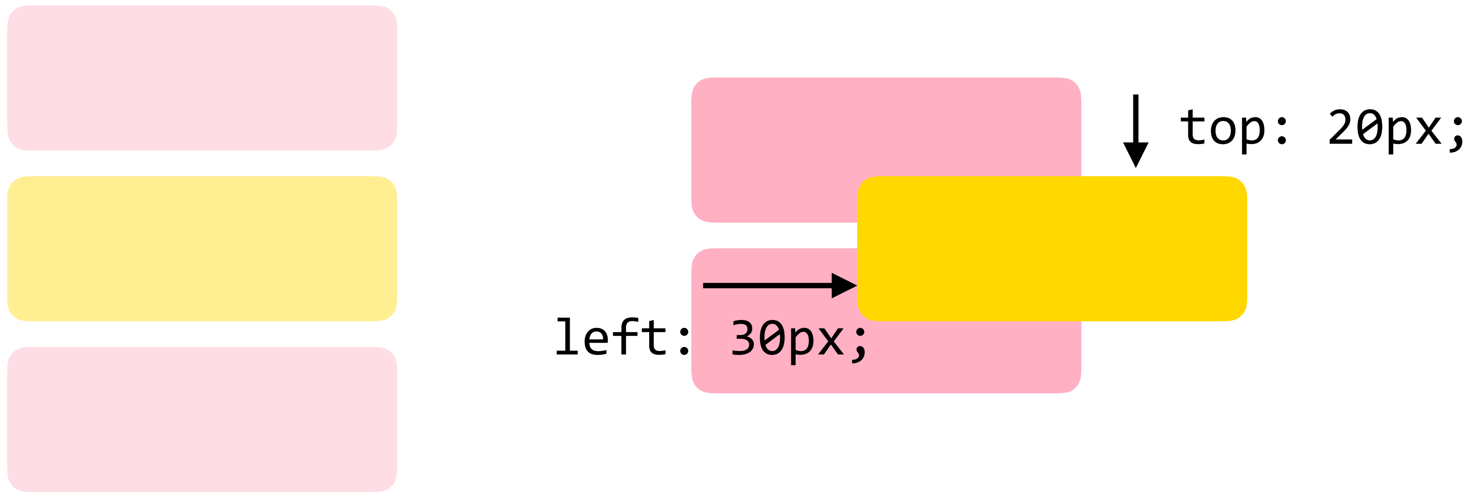
ANIMATION

ABSOLUTE

ABSOLUTE POSITIONING

- Position an element exactly where you want it
- 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

```
yourSelectorHere {
  position: absolute;
  top: 20px;
  left: 30px;
}
```

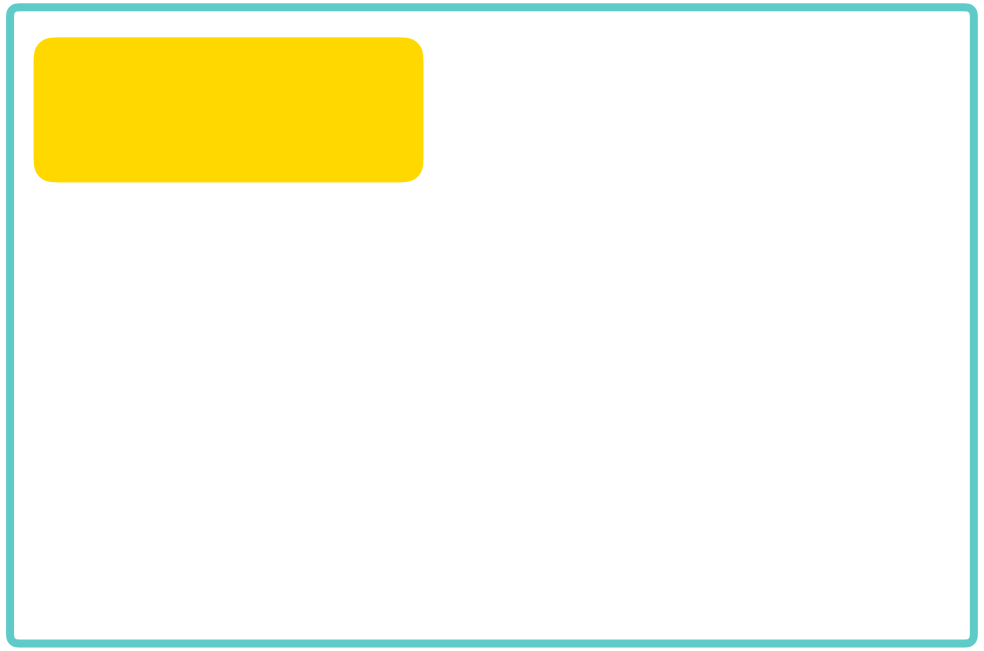


POSITIONING THINGS ABSOLUTELY

Parent we want child to be positioned relative to

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

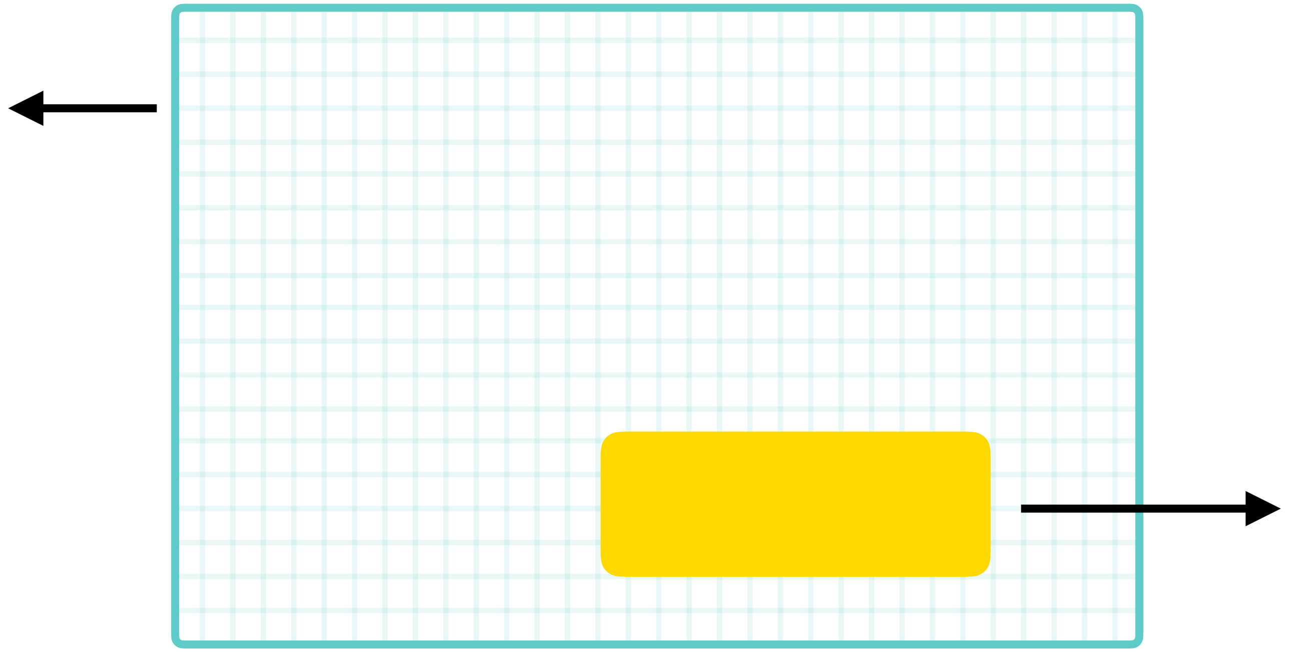
Child we want to position



POSITIONING THINGS ABSOLUTELY

Parent:

`position: relative;`



Child:

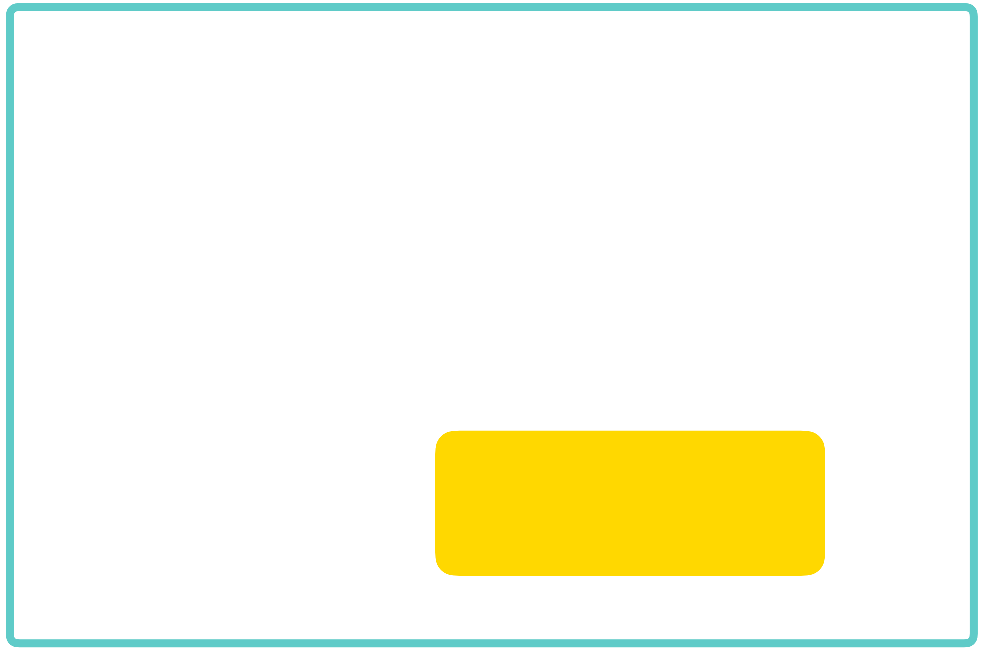
- `position: absolute;`
- `top, bottom, left, or right`

POSITIONING THINGS ABSOLUTELY

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

```
section {
  position: relative;
}

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



ACTIVITY — ABSOLUTE POSITIONING

EXERCISE

KEY OBJECTIVE

- Practice using CSS positioning

LOCATION

- Starter Code > positioning practice

TIMING

8 min	1. Follow step 2 in main.css

WANT TO LEARN MORE?

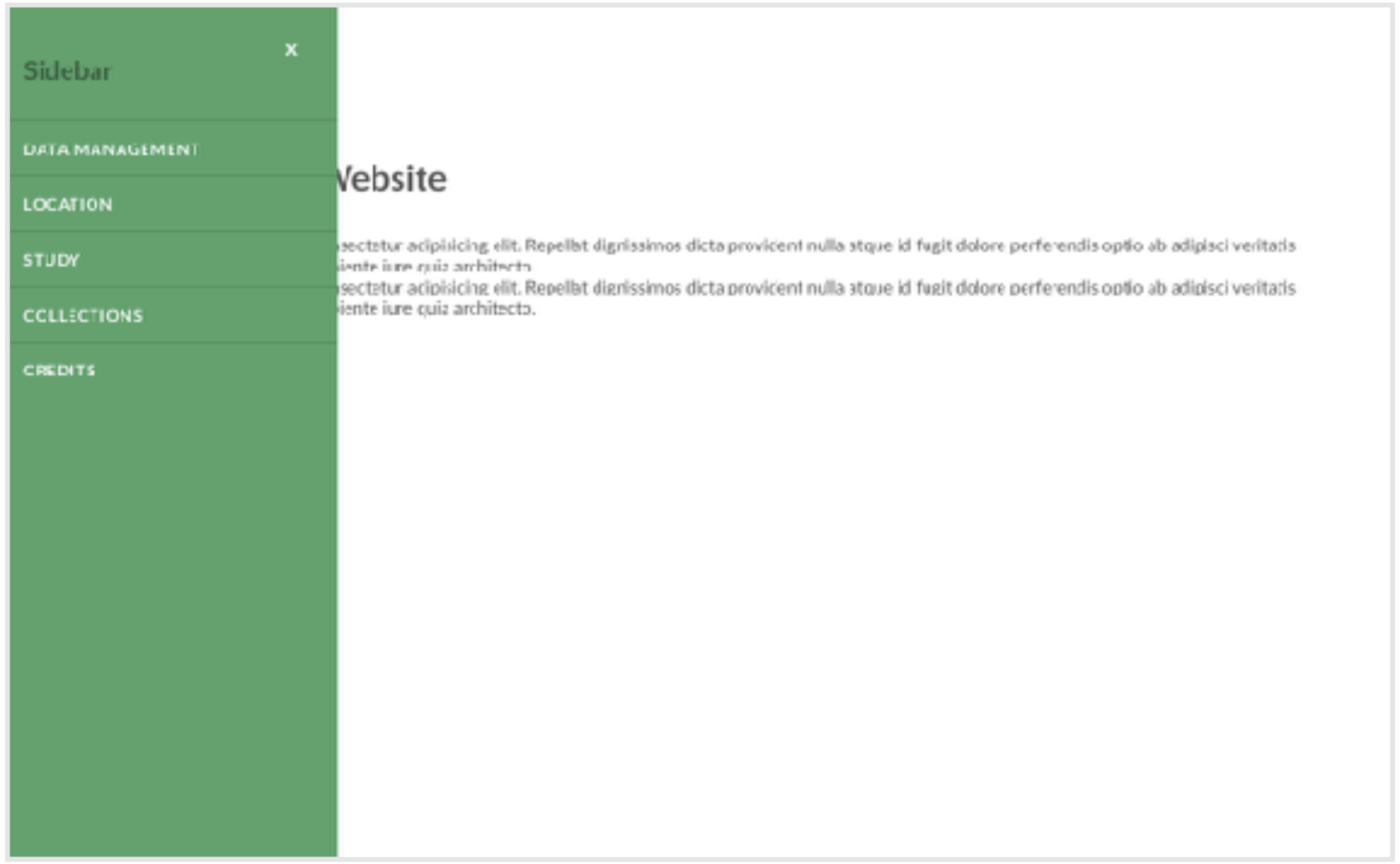
Resources for more info/examples:

- A List Apart: CSS Positioning 101

ACTIVITY — SCROLL ANIMATIONS

1. We want to position an element inside another element. One is a parent and the other is the child. Which would we set a position of relative and which would we give a position of absolute? Do we need to add anything else?
2. Take a look at the following four slides. Which positioning would we need to use for each?

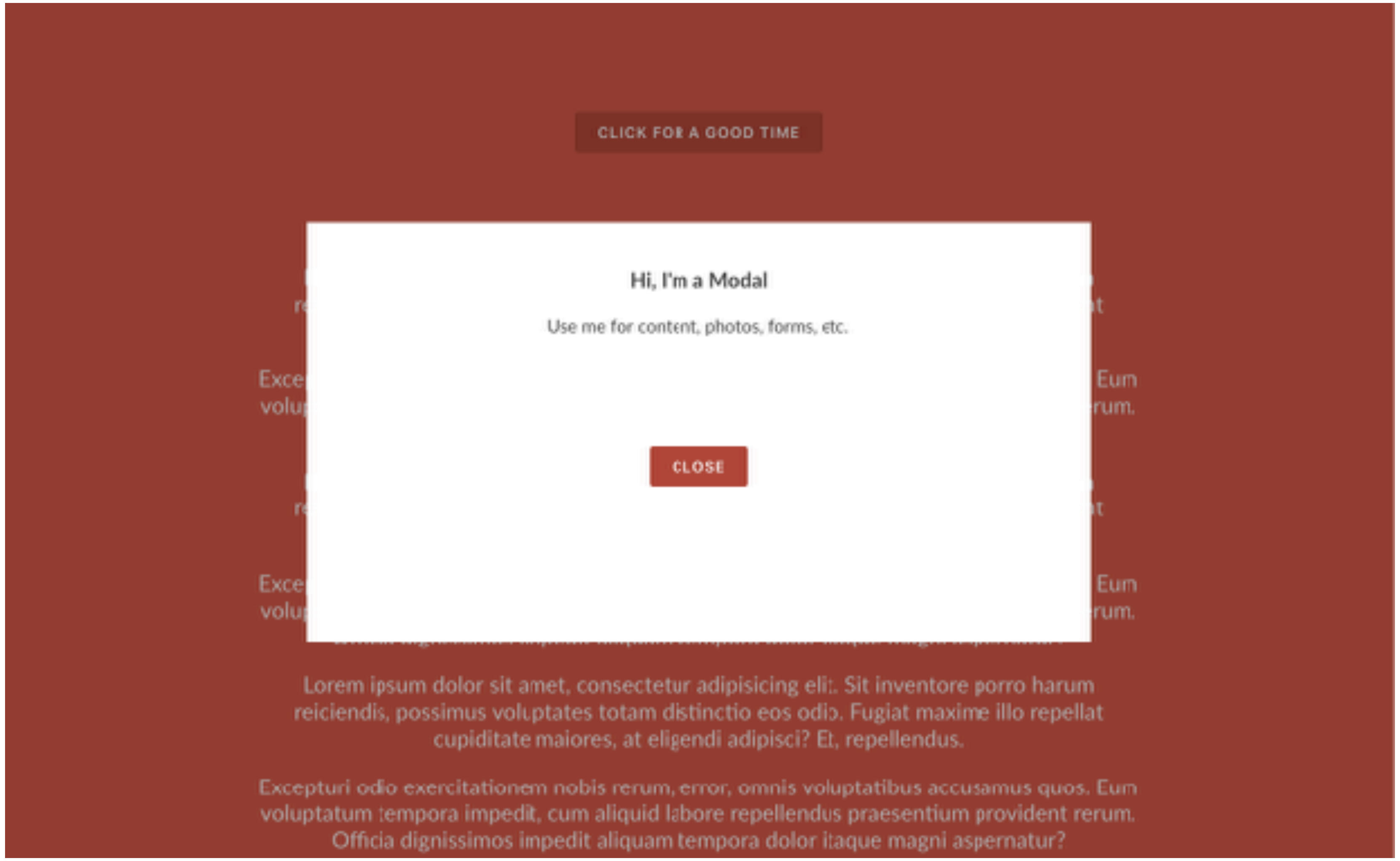
CSS POSITIONING — SIDEBAR



Vebsite

sectetur adipisicing elit. Repellat dignissimos dicta provicent nulla atque id fugit dolore perferendis optio ab adipisci veritatis
iente iure quia architecto.
sectetur adipisicing elit. Repellat dignissimos dicta provicent nulla atque id fugit dolore perferendis optio ab adipisci veritatis
iente iure quia architecto.

CSS POSITIONING — MODAL WINDOW



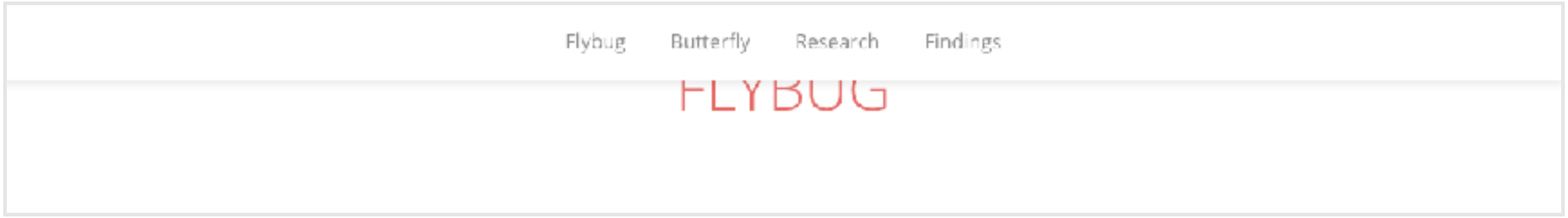
Hi, I'm a Modal

Use me for content, photos, forms, etc.

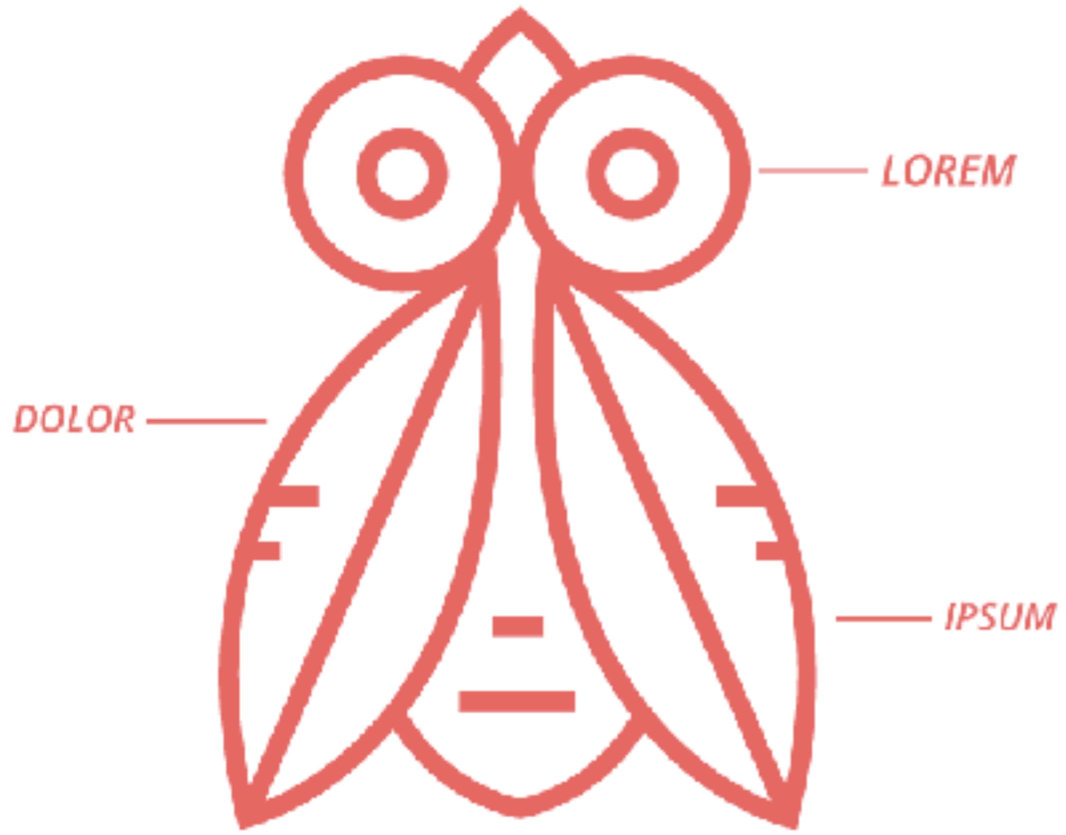
Lorem ipsum dolor sit amet, consectetur adipisicing elit. Sit inventore porro harum reiciendis, possimus voluptates totam distinctio eos odio. Fugiat maxime illo repellat cupiditate maiores, at eligendi adipisci? Et, repellendus.

Excepturi odio exercitationem nobis rerum, error, omnis voluptatibus accusamus quos. Eum voluptatum tempora impedit, cum aliquid labore repellendus praesentium provident rerum. Officia dignissimos impedit aliquam tempora dolor itaque magni aspernatur?

CSS POSITIONING — STICKY NAV



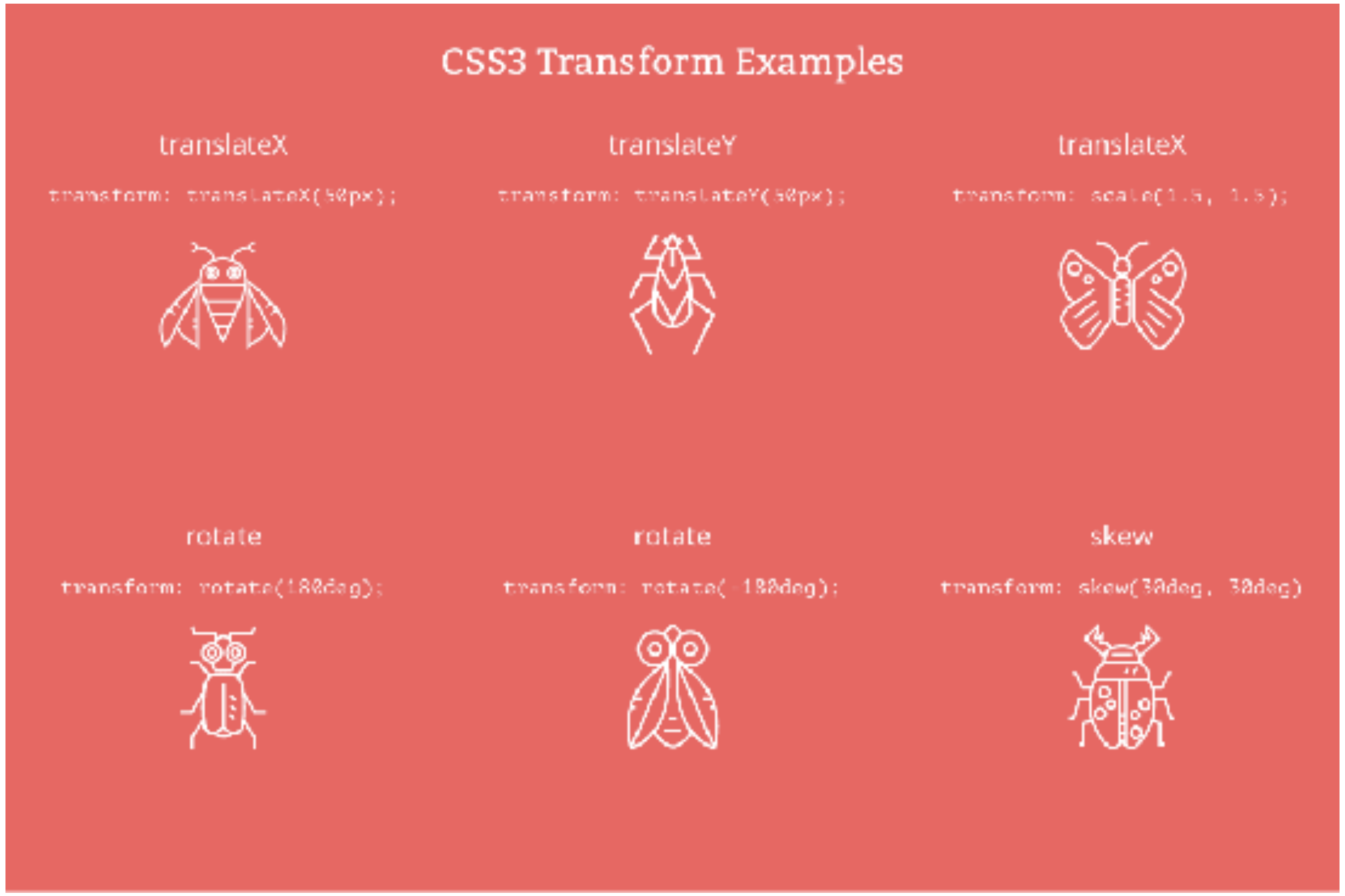
CSS POSITIONING — LABELS FOR IMAGE



ANIMATION

TRANSITIONS

LET'S TAKE A CLOSER LOOK — TRANSFORM



CSS3 Transform Examples

translateX

```
transform: translateX(50px);
```

translateY

```
transform: translateY(50px);
```

translateX

```
transform: scale(1.5, 1.5);
```

rotate

```
transform: rotate(180deg);
```

rotate

```
transform: rotate(-180deg);
```

skew

```
transform: skew(30deg, 30deg)
```

Syntax: W3 Schools

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.

```
yourSelectorHere {
    transition: what-to-transition animation-duration timing-function;
}
```

EXAMPLE:

```
transition: all 350ms ease-in-out;
```

TRANSITIONS - TRANSITION-PROPERTY

PROPERTIES TO ANIMATE

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

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

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

TRANSITIONS - TRANSITION-DURATION

DURATION

- A time value, defined in seconds or milliseconds

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

```
yourSelectorHere {
    transition: height 300ms;
}
```

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`**

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

Examples for each

ACTIVITY — BUTTON LAB

EXERCISE

KEY OBJECTIVE

- Practice using CSS transitions

TYPE OF EXERCISE

- Individual/Partner Lab

TIMING

6 min	1. Add :hover styles and transition to the button

BORDER SHORTHAND

For a border on all sides:

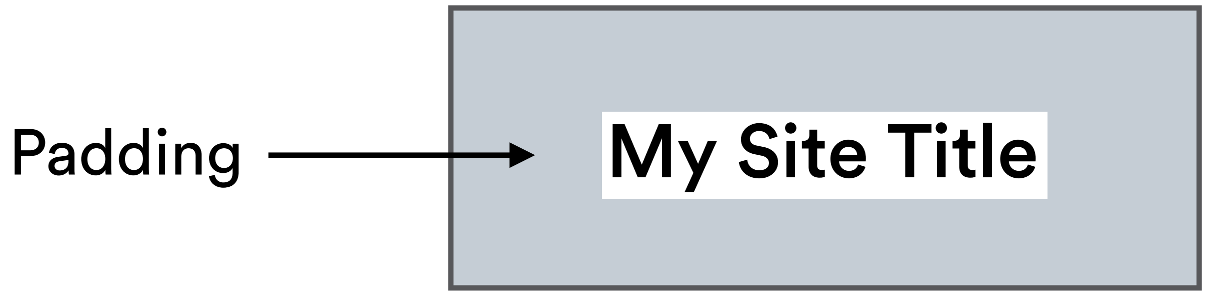
```
border: width style color;           border: 1px solid red;
```

For a border on one side:

```
border-top: width style color;       border-top: 1px solid red;
border-right: width style color;     border-right: 1px solid red;
border-bottom: width style color;    border-bottom: 1px solid red;
border-left: width style color;      border-left: 1px solid red;
```

PADDING

We can add `padding` to get space between our content and the border



MORE FUN WITH TRANSITIONS — CODROPS

Fun CSS button styles: Creative buttons

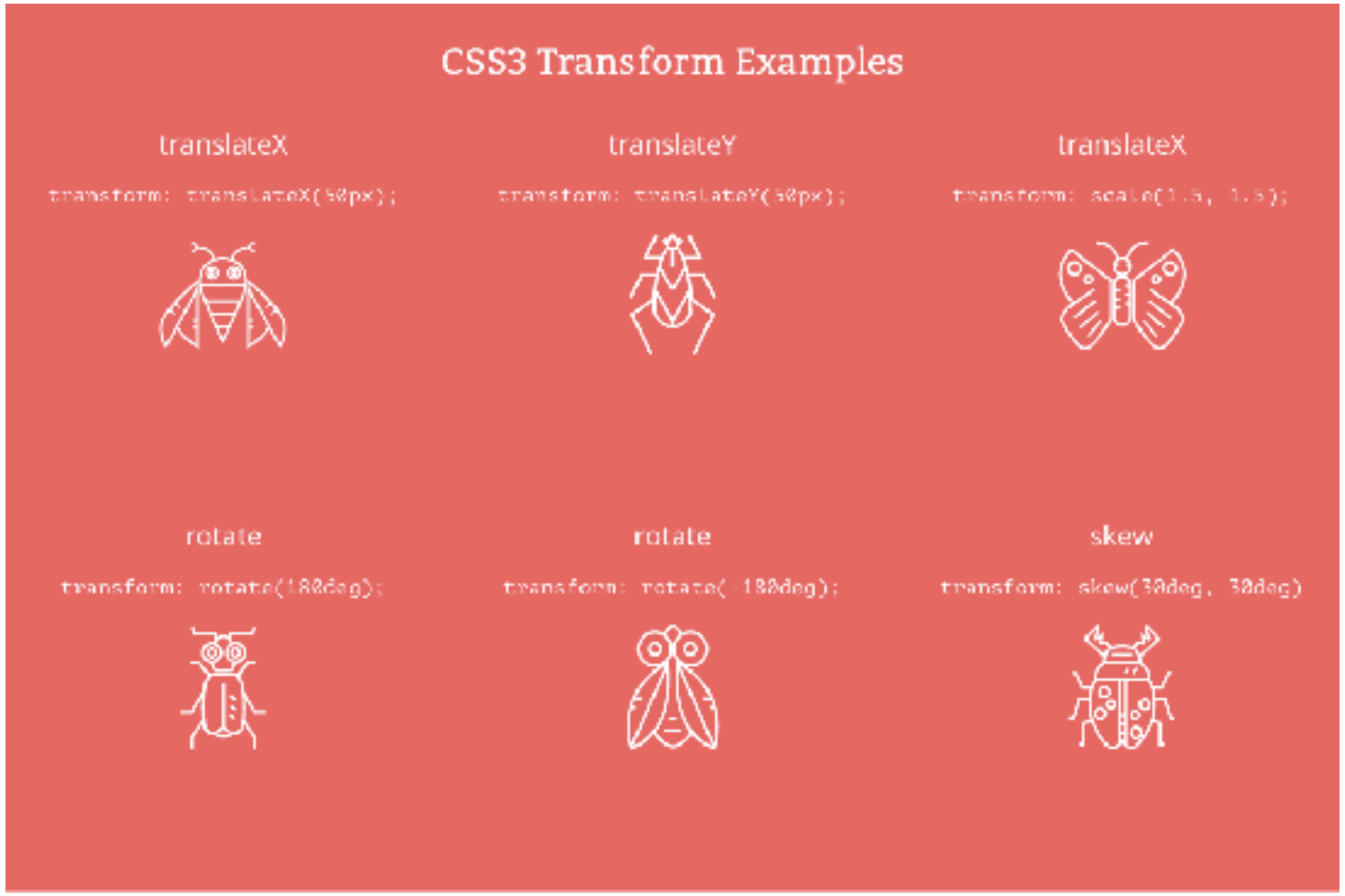
Icon hover effects: Icon Hover Effects

Modal dialogue effects (advanced): Dialogue Effects

ANIMATION

TRANSFORMATIONS

LET'S TAKE A CLOSER LOOK — TRANSFORM



CSS3 Transform Examples

translateX

```
transform: translateX(50px);
```

translateY

```
transform: translateY(50px);
```

translateX

```
transform: scale(1.5, 1.5);
```

rotate

```
transform: rotate(180deg);
```

rotate

```
transform: rotate(-180deg);
```

skew

```
transform: skew(30deg, 30deg)
```

Syntax: [W3 Schools]

ACTIVITY — TRANSFORM ON TIMER

EXERCISE

KEY OBJECTIVE

- Practice using CSS transitions

TYPE OF EXERCISE

- Individual/Partner Lab

TIMING

10 min

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

IMAGE OVERLAY LAB

ACTIVITY

EXERCISE

LOCATION

- starter code > image_overlay

KEY OBJECTIVE

- Practice working through common interactions

TIMING

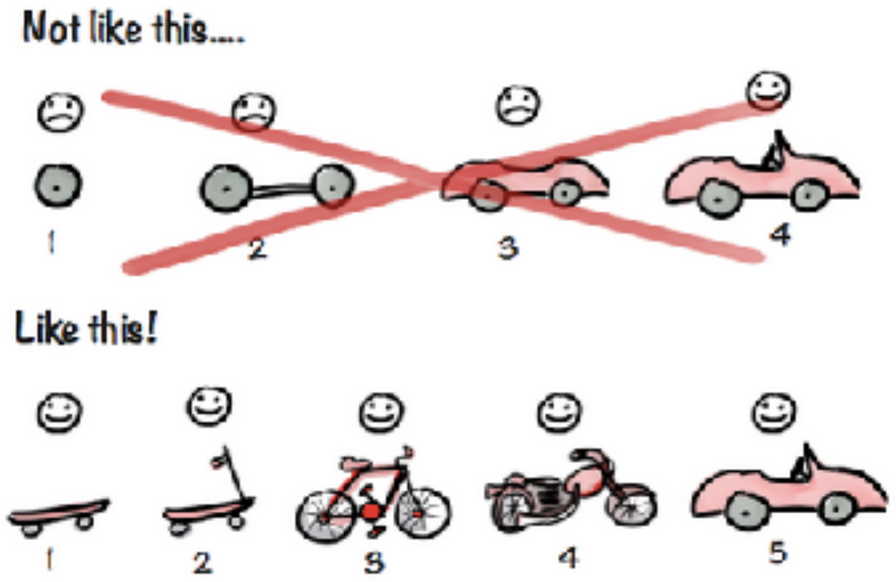
20 min	1. Work through the steps in your CSS.

FINAL PROJECT

WIREFRAMES

PLANNING OUT YOUR PROJECT

- In software development, you'll hear this advice over and over: start with the simplest possible thing you can build to reach your goal, then add on!
- You probably already have an awesome project idea. Most people don't have time during this class to get to the "racecar" version of their projects, but you should learn a lot and get a working site.



TECHNICAL REQUIREMENTS — SCOPE

- Ensure projects are feasible and appropriately scoped
- Save development time by planning thoroughly
- Figure out the simplest possible implementation of your idea. What is the "skateboard" version of your site?
- Keep in mind that because this is a front-end course, we won't go over how to add back-end features like logging in users or storing user data.
- Write down a few sentences that describe what your vision is and what the most important parts of your site are.

TECHNICAL REQUIREMENTS — WIREFRAMES

- Sketch out the core pages of your app.
- Start simple: draw some boxes. Add some text to the boxes to show what part of the page they are, like the header, sidebar, images, titles, articles, and so on.

SUBMISSION

Create a folder wireframes and save any images, pdfs, wireframes in that folder

Then drag and drop to your GitHub homework folder

WEEKLY OVERVIEW

WEEK 4	Responsive Design / CSS Positioning
WEEK 5	Forms / Final Project Lab
WEEK 6	Intro to JS / Functions

LEARNING OBJECTIVES

- Use the `position` property to position elements on the page
- Utilize transitions and transforms to add basic animations on hover

EXIT TICKETS