Note: we used `img` as the selector, we can decide to give it a `className` and make use of it as the selector.

How to Resize an Image With Inline Styles

We used external styling in the previous example, but just like in traditional HTML, we can use the `style` attribute to add CSS styling. The `style` attribute value must be a JavaScript object with key-value pairs:

```
import Logo from './images/react-logo.png';
const App = () => {
   return (
      <div>
         <img style={{ width: 500, height: 600 }} src={Logo} alt="React Logo" />
      </div>
   );
};
```

By default, the basic unit is in pixels, but suppose we want to make use of other units like `rem`, `%`, `vh`, etc. We will make use of string for the styles' key value:

```
import Logo from './images/react-logo.png';
const App = () => {
   return (
      <div>
         <img style={{ width: "500%", height: "600%" }} src={Logo} alt="React Logo" />
      </div>
   );
};
```

If we have many images that need similar styling and don't want to use external styling, we could create an object to hold these styles objects and then add the object to the `styles` attribute:

```
import Logo from './images/react-logo.png';

const App = () => {
   const myImageStyle = { width: '500px', height: '600px' };

   return (
      <div>
         <img style={myImageStyle} src={Logo} alt="" />
      </div>
   );
};
```

How to Resize an Image With the *width* And *height* Attributes

In traditional HTML, one way to resize images is to make use of the `height` and `width` property with the `img` tag and this also works with React:

```
import Logo from './images/react-logo.png';
const App = () => {
   return (
      <div>
         <img src={Logo} width="500" height="600" alt="" />
         <!-- OR -->
         <img src={Logo} width={500} height={600} alt="" />
      </div>
   );
};
```

The main drawback of this method is that fiddling with the height and width tends to distort images, making them shrink, stretch or otherwise lose their ratio. This can be fixed by using `object-fit: cover;`.

Styling Our Images

When we use the `height`, `width`, `max-height`, and other CSS properties to resize our images, they tend to distort them, making them shrink or stretch.

It's always a good idea to include the `object-fit` property, which specifies how an image should be resized to fit its container. This property can accept a variety of values such as `contain`, `cover`, `fill`, `none` and `scale-down`.

Other CSS properties, such as `max-width`, `min-width`, `max-height`, and `min-height`, can define the maximum and minimum values an image can hit, limitting distortion.