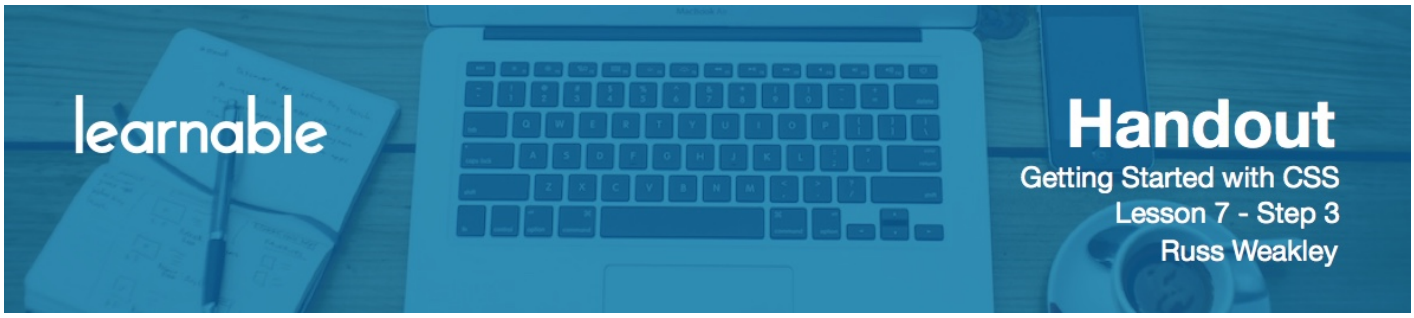

Relative Positioning

Now let's look at relative positioning. This can be written with

```
div {position: relative}
```

When given relative positioning, an element will move out of flow or up off the page. In flow content that comes after the relatively positioned element will behave as if that element was still in normal flow, so it will leave a gap for it. Elements with relative positioning will behave as though not positioned in terms of their width, so they will maintain their block level, their inline, or their inline-block behavior.

You can assign positioning to a relatively positioned element using the `top`, `right`, `bottom`, and `left` properties. A relatively positioned element will be positioned relative to its original position in normal flow.

Exercise

Let's assign positioning for the two elements:

```
h2, strong { position: relative; }
```

If you reload the page, nothing will visually change. We need to give those elements some `left` or `right` values:

```
h2, strong { left: 20px; }
```

These two elements will be shifted to the right by 20px, but the interesting thing is that they will be positioned relative to their original position. They're not being positioned relative to the parent or the viewport like absolutely positioned elements.

```
h2, strong { top: 20px; }
```

The same thing is going to happen: they will be shifted down now, positioned relative to their original position.

```
h2, strong { width: 300px; }
```

The `strong` element, which is an inline element, won't change while the `h2` element, which is a block level element, will have its width changed.