The `input[type='text']` part is a new type of CSS selector called an "attribute selector". It only matches `<input/>` elements that have a `type` attribute equal to `text`. This lets us specifically target text fields opposed to radio buttons, which are defined by the same HTML element (`<input type='radio'/>`). You can read more about attribute selectors at Mozilla Developer Network.

All of our styles are "namespaced" in a `.form-row` descendant selector. Isolating `<input/>` and `<label>` styles like this makes it easier to create different kinds of forms. We'll see why it's convenient to avoid global `input[type='text']` and `label` selectors once we get to radio buttons.

Finally, let's tweak these base styles to create our desktop layout. Add the following media query to the end of our stylesheet.

```
@media only screen and (min-width: 700px) {
  .speaker-form-header,
  .speaker-form {
    width: 600px;
  }
  .form-row {
    flex-direction: row;
    align-items: flex-start; /* To avoid stretching */
    margin-bottom: 20px;
  }
  .form-row input[type='text'] {
    width: 250px;
    height: initial;
  }
  .form-row label {
    text-align: right;
    width: 120px;
    margin-top: 7px;
    padding-right: 20px;
  }
}
```

Check out that awesome use of the `flex-direction` property to make the `<label>` appear on top of its `<input/>` element in the mobile layout, but to the left of it in the desktop layout.