Coding CSS defensively

When working with CSS technologies, we may be in situations where not all browsers support a property and we need to code around this partial support for the feature we're testing.

To me, this is particularly important when creating examples to publish as part of a blog post or as standalone examples in Codepen.

The idea is as follows:

1. Add a message to the example HTML and style it appropriately
2. Check if the browser supports the feature
    1. If it does, hide the message
    2. If the feature is not supported then make sure the message is displayed

Using `:dir(rtl)` as an example, we add the following `div` with a warning class to the example.

```html
<!-- 1 -->
<div class="warning">
  <h2>Feature not supported</h2>

  <p>Your browser doesn't support the <code>:dir</code> pseudo class</p>
</div>
```

In the CSS we create a class to style our warning `div`.

```css
/* 1 */
.warning {
  color: red;
  text-align: center;
}
```

We then test if the feature is supported using the @supports at rule with the `selector()` syntax so rather than testing for an attribute we're testing if the browser supports a given selector.

If the browser supports the `:dir(rtl)` selector, we hide the warning by setting the div's display to none.

```
/* i */
@supports selector(:dir(rtl)) {
  .warning {
    display: none;
  }
}
```

This may not be totally necessary as the rule will remain visible if the browser doesn't support the rule but I believe in redundancy so I add another `@suports` block with `selector()` syntax and the `not` operator. This time we're testing if the browser **doesn't** support the selector.

If the `:dir(rtl)` selector is not supported then we make sure that the warning is show by setting the `display` attribute on the warning div to `block`.

```
/* ii */
@supports not (selector(:dir(rtl))) {
  .warning {
    display: block;
  }
}
```

We could do other tings as well. In the second support block, if the selector is not supported we could also provide fall backs, polyfills or workarounds to make the feature work closer to the original version.

However, for the purposes of demonstration, just flagging that the feature is not supported is enough for me.