nothing leaks from the component to the app, or vice-versa.

Going back to our previous example:

```
var PonyComponentProto = Object.create(HTMLElement.prototype);

// add some template in the Shadow DOM
PonyComponentProto.createdCallback = function() {
  var shadow = this.createShadowRoot();
  shadow.innerHTML = '<h1>General Soda</h1>';
};

var PonyComponent = document.registerElement('ns-pony', {prototype:
PonyComponentProto});
document.body.appendChild(new PonyComponent());
```

If you try to inspect it now you should see:

```
<ns-pony>
  #shadow-root (open)
    <h1>General Soda</h1>
</ns-pony>
```

Now, even if you try to add some style to the `h1` elements, the visual aspect of the component won't change at all: that's because the Shadow DOM acts like a barrier.

Until now, we just used a string as a template of our web component. But that's usually not the way you do that. Instead, the best practice is to use the <template> element.

6.4. Template

A template specified in a `<template>` element is not displayed in your browser. Its main goal is to be cloned in an element at some point. What you declare inside will be inert: scripts don't run, images don't load, etc. Its content can't be queried by the rest of the page using usual method like `getElementById()` and it can be safely placed anywhere in your page.

To use a template, it needs to be cloned: