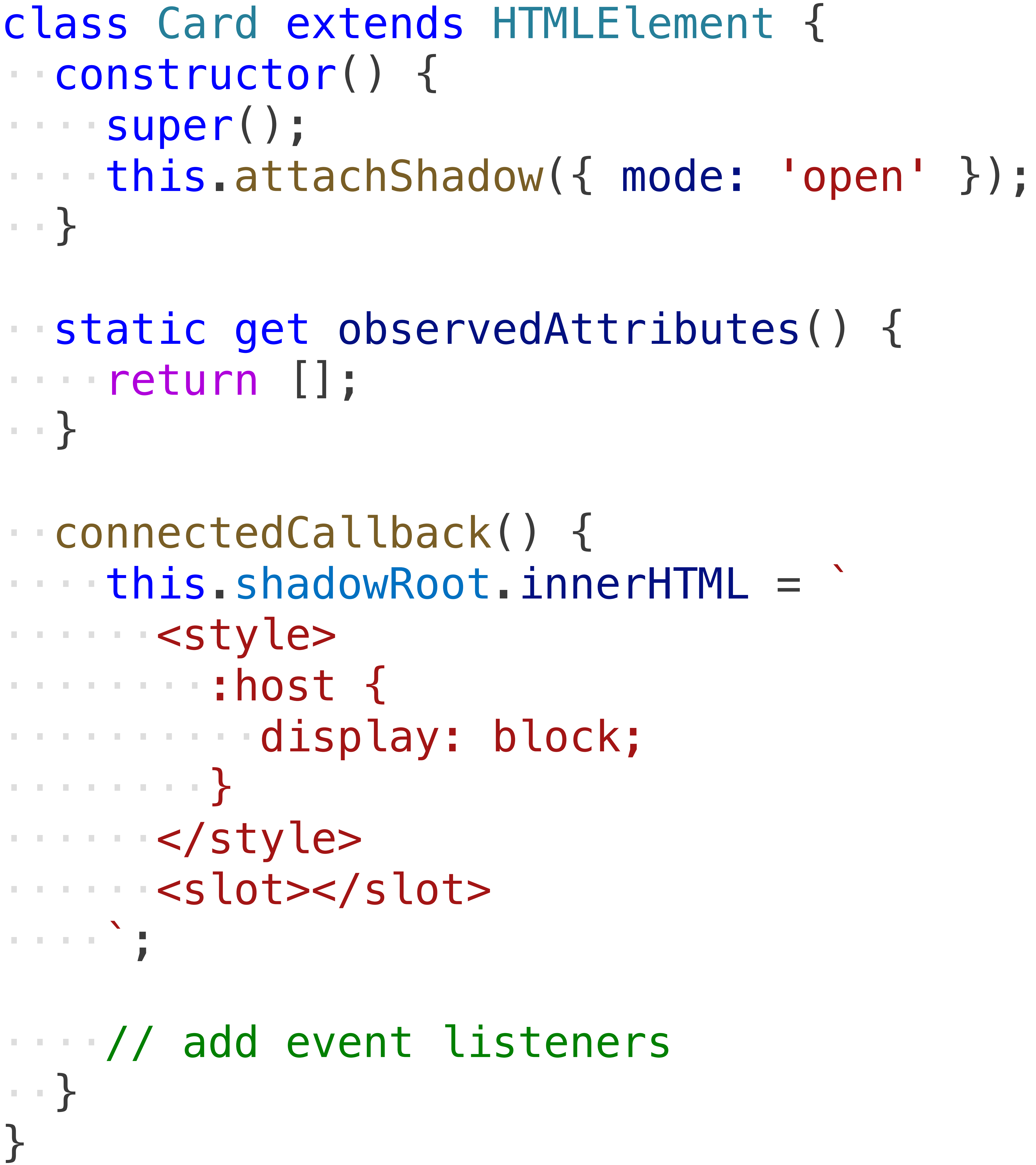


```
class Card extends HTMLElement {
  constructor() {
    super();
    this.attachShadow({ mode: 'open' });
  }

  static get observedAttributes() {
    return [];
  }

  connectedCallback() {
    this.shadowRoot.innerHTML = `
      <style>
        :host {
          display: block;
        }
      </style>
      <slot></slot>
    `;

    // add event listeners
  }
}
```

We will focus today only on extending HTMLElement though you can extend specific elements as well.

I’ll use **vanilla custom element** syntax today, *however*…

in production, I’d strongly recommend leveraging a library such as Lit to streamline development.

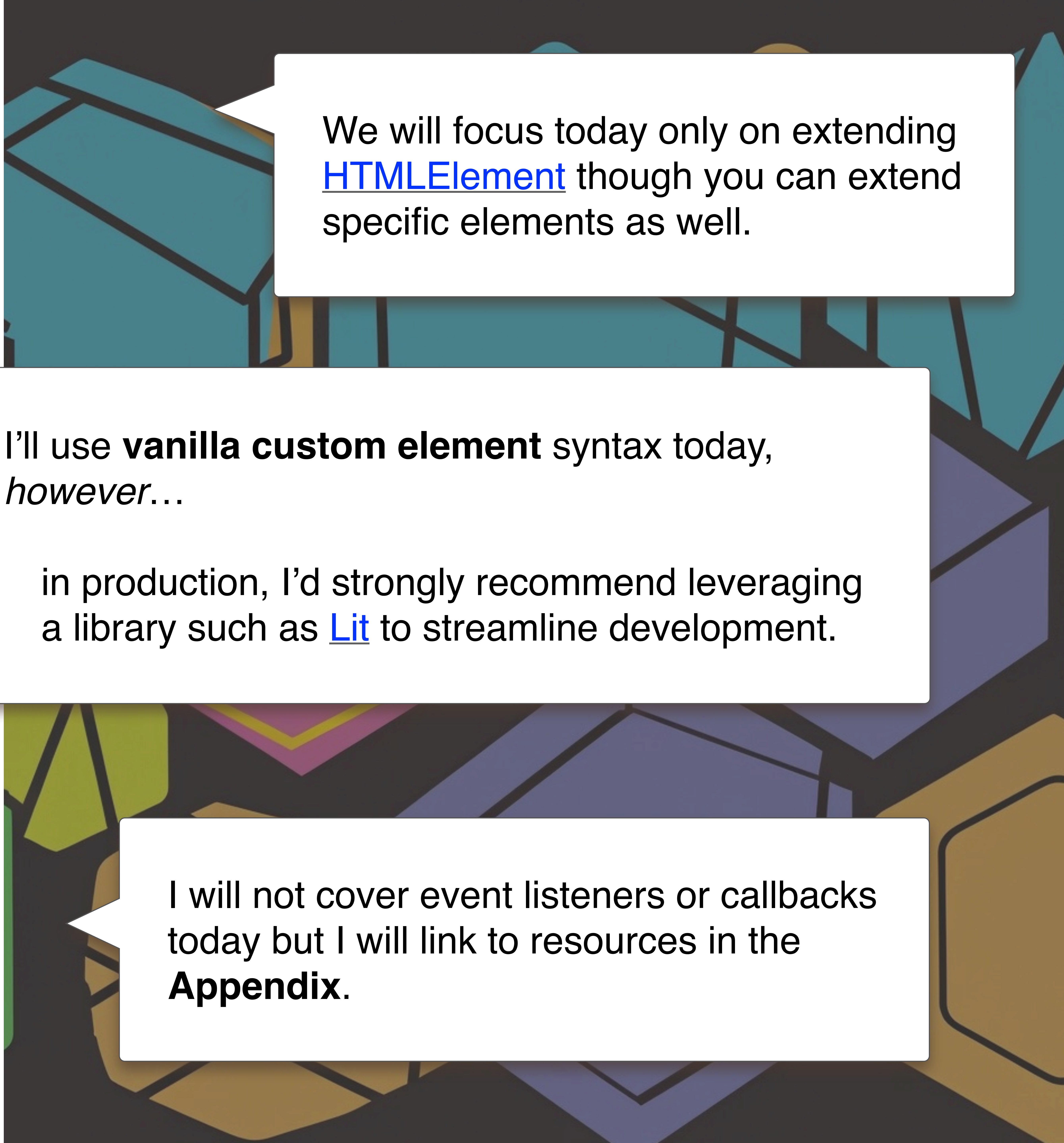
I will not cover event listeners or callbacks today but I will link to resources in the **Appendix**.

```
class Card extends HTMLElement {
  constructor() {
    super();
    this.attachShadow({ mode: 'open' });
  }

  static get observedAttributes() {
    return [];
  }

  connectedCallback() {
    this.shadowRoot.innerHTML = `
      <style>
        :host {
          display: block;
        }
      </style>
      <slot></slot>
    `;

    // add event listeners
  }
}
```



We will focus today only on extending HTMLElement though you can extend specific elements as well.

I'll use **vanilla custom element** syntax today, *however…*

in production, I'd strongly recommend leveraging a library such as Lit to streamline development.

I will not cover event listeners or callbacks today but I will link to resources in the **Appendix**.

Custom Element

How we register elements to the page.

```
window.customElements.define("the-card", Card);
```

Custom element registry: Instance methods