Web Components

Bringing Reusability to Native HTML

The Problem They Solve

❌ HTML is not reusable

❌ CSS is global

❌ Frameworks don’t play well together

✅ We want **encapsulated**, **reusable** element

What Are Web Components?

Web Components are:

1. Custom Elements

2. Shadow DOM

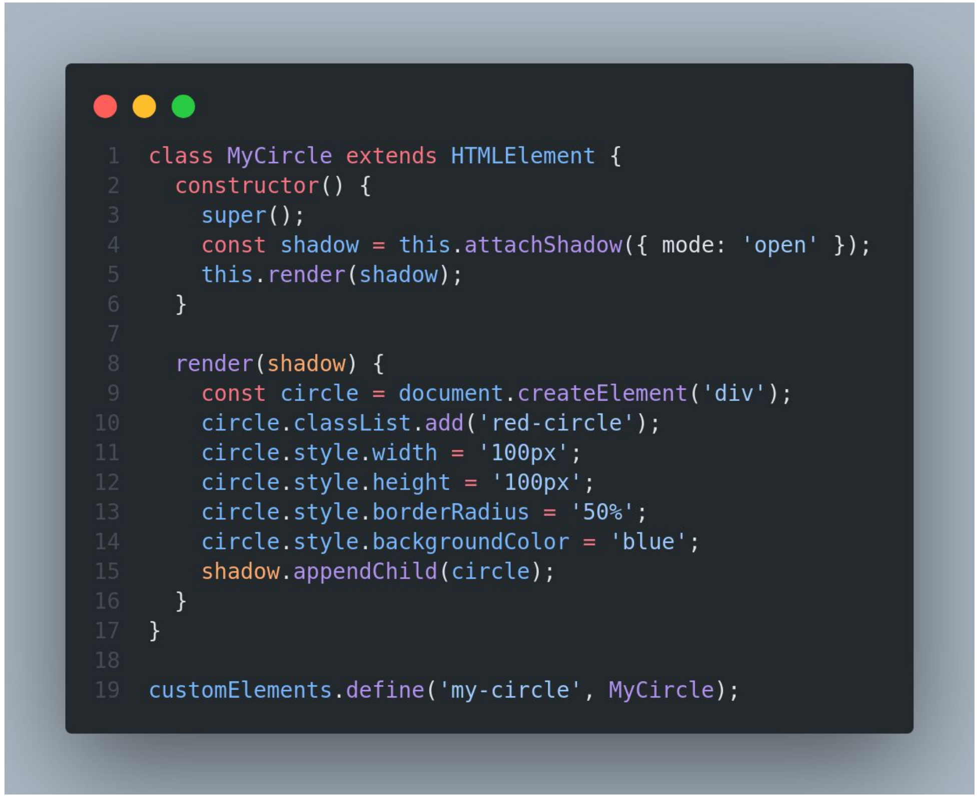
3. HTML Templates

Custom Elements

🧩 Custom HTML tags

📦 With their own behavior & logic

🤯 Like `<video>` or `<input>`, but yours

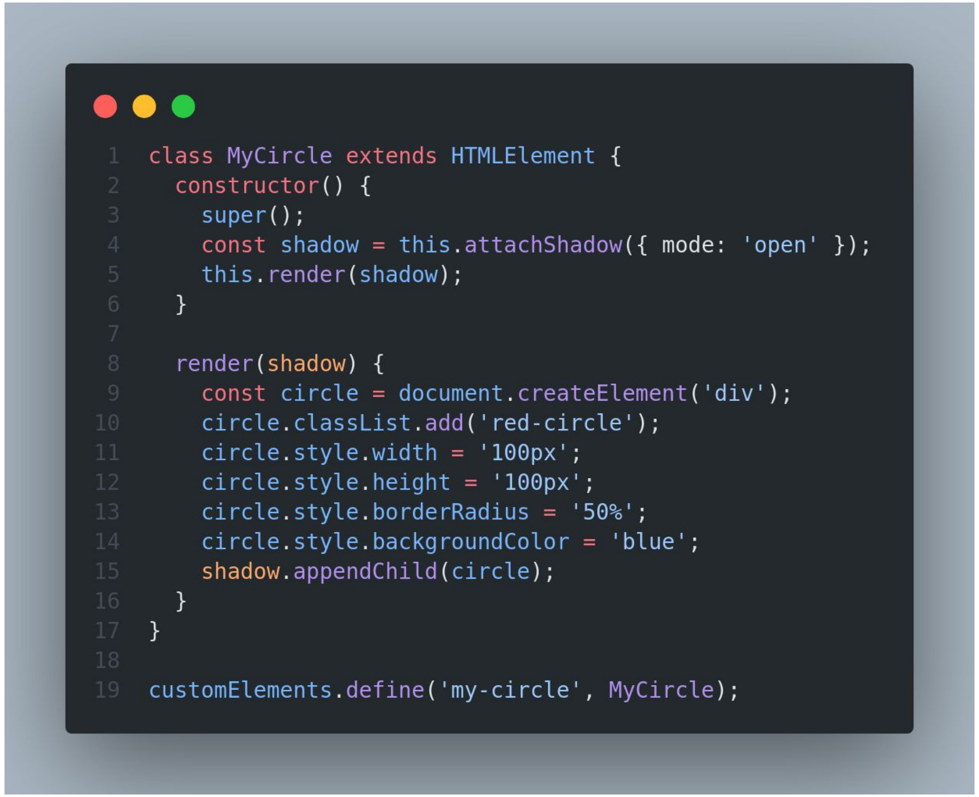


```
class MyCircle extends HTMLElement {
  constructor() {
    super();
    const shadow = this.attachShadow({ mode: 'open' });
    this.render(shadow);
  }

  render(shadow) {
    const circle = document.createElement('div');
    circle.classList.add('red-circle');
    circle.style.width = '100px';
    circle.style.height = '100px';
    circle.style.borderRadius = '50%';
    circle.style.backgroundColor = 'blue';
    shadow.appendChild(circle);
  }
}

customElements.define('my-circle', MyCircle);
```

Code: Creating a Custom Element



```javascript
class MyCircle extends HTMLElement {
  constructor() {
    super();
    const shadow = this.attachShadow({ mode: 'open' });
    this.render(shadow);
  }

  render(shadow) {
    const circle = document.createElement('div');
    circle.classList.add('red-circle');
    circle.style.width = '100px';
    circle.style.height = '100px';
    circle.style.borderRadius = '50%';
    circle.style.backgroundColor = 'blue';
    shadow.appendChild(circle);
  }
}

customElements.define('my-circle', MyCircle);
```

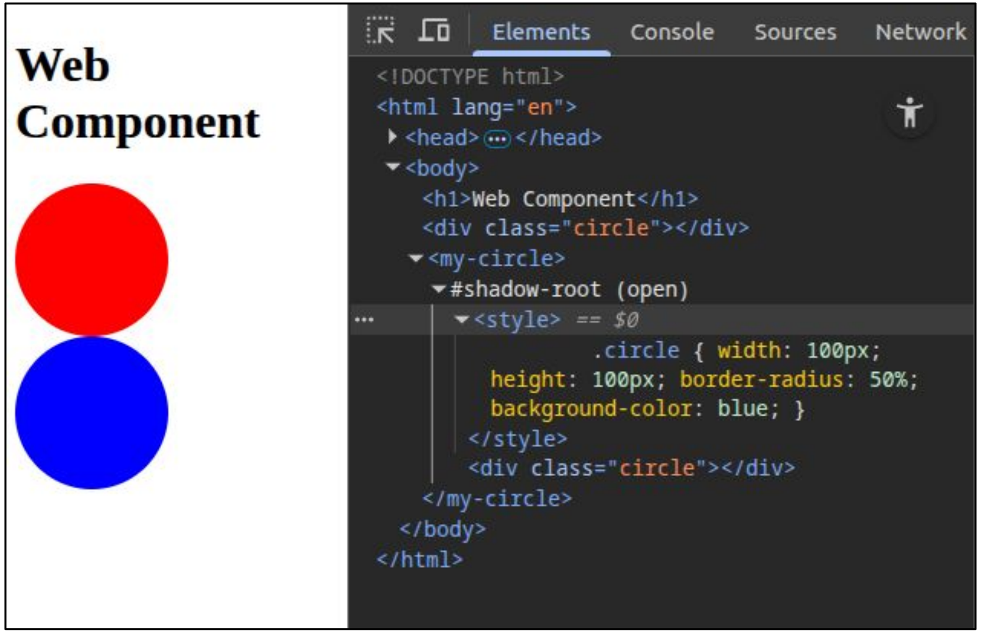
Shadow DOM

🔐 Isolated DOM tree

🎨 Scoped CSS

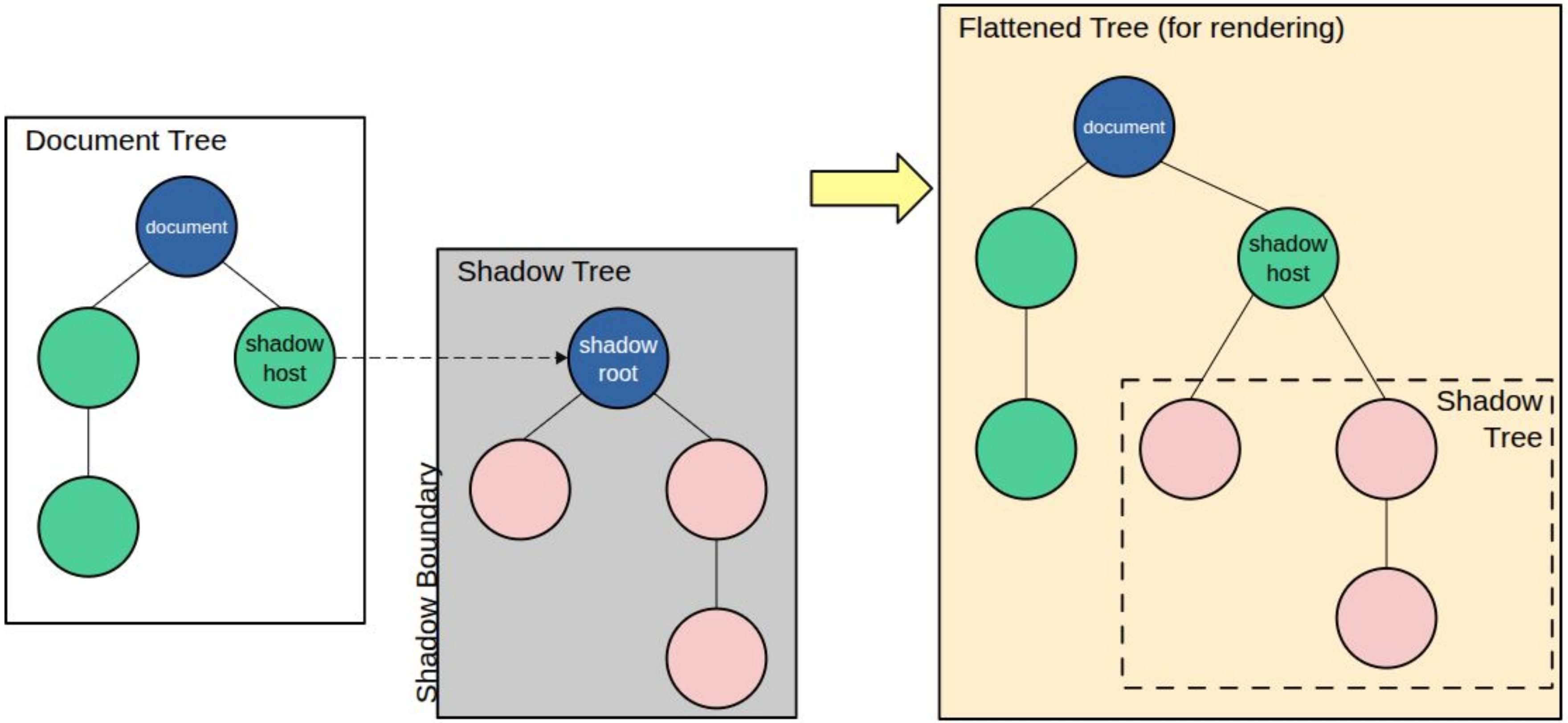
❌ No global styles leak in

✅ No global styles leak out



Web Component

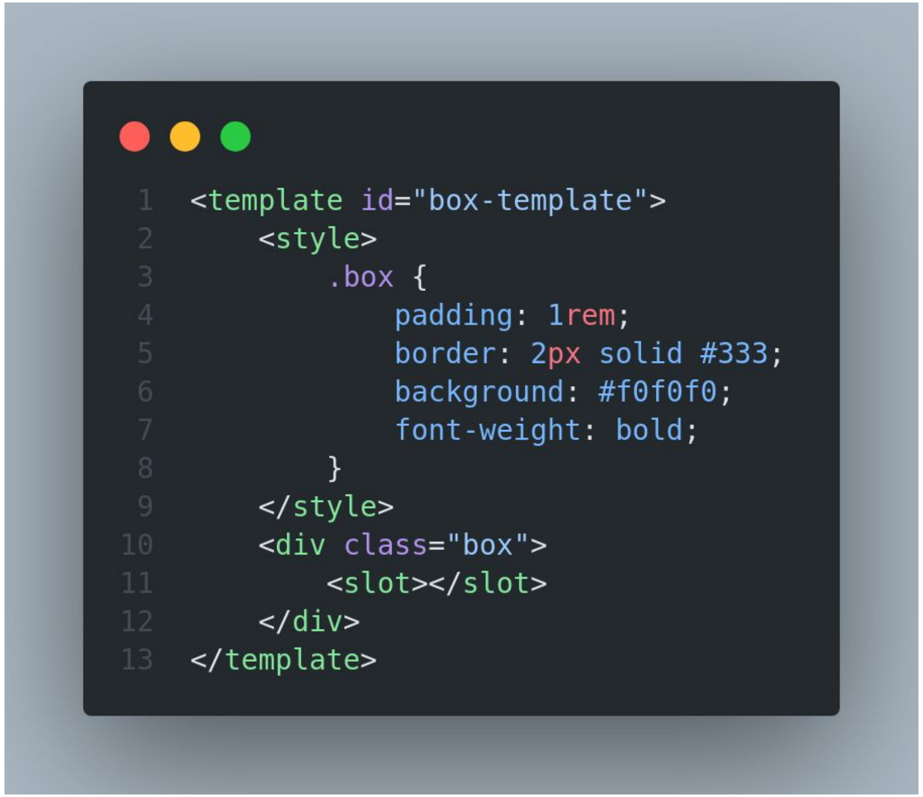
```
<!DOCTYPE html>
<html lang="en">
  ▸<head> ⋯ </head>
  ▾<body>
      <h1>Web Component</h1>
      <div class="circle"></div>
    ▾<my-circle>
      ▾#shadow-root (open)
        ▾<style> == $0
                  .circle { width: 100px;
            height: 100px; border-radius: 50%;
            background-color: blue; }
          </style>
          <div class="circle"></div>
      </my-circle>
    </body>
</html>
```



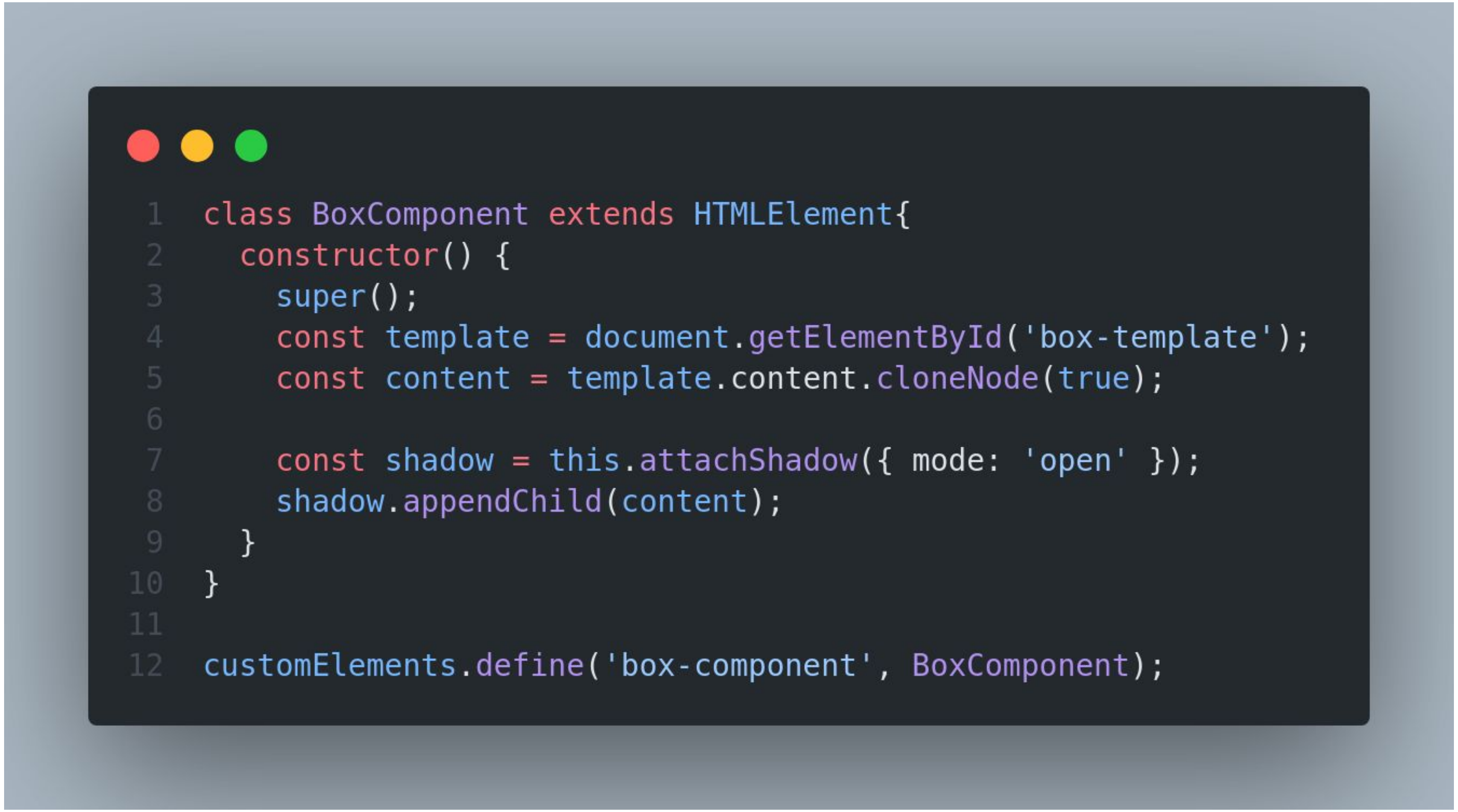
Document Tree
document
shadow host
Shadow Tree
shadow root
Shadow Boundary
Flattened Tree (for rendering)
document
shadow host
Shadow Tree

HTML Templates

- 📄 Define markup without rendering
- 🧬 Clone it into shadow DOM
- 🌀 Great for reusability



```html
<template id="box-template">
    <style>
        .box {
            padding: 1rem;
            border: 2px solid #333;
            background: #f0f0f0;
            font-weight: bold;
        }
    </style>
    <div class="box">
        <slot></slot>
    </div>
</template>
```



```javascript
class BoxComponent extends HTMLElement{
  constructor() {
    super();
    const template = document.getElementById('box-template');
    const content = template.content.cloneNode(true);

    const shadow = this.attachShadow({ mode: 'open' });
    shadow.appendChild(content);
  }
}

customElements.define('box-component', BoxComponent);
```

So Why Aren’t Web Components Everywhere?

🚧 Developer Experience

❌ No built-in state management

❌ No JSX / Reactivity

⚠️ Poor SSR

🔌 Weak framework integration

Final Thoughts

✅ Native component model

✅ Encapsulation, reusability

🚀 Framework-agnostic

🧰 Ideal for design systems

🧪 Still evolving