



lit: another look at web components

[lit](#) is the spiritual successor to the [Polymer](#) Project and its ecosystem.

Like Polymer, Lit provides abstractions over the web component APIs and provides functionality that have no WebComponent API equivalent (like the import/export capabilities we lost when HTML Imports went away).

This post will cover the basics of Lit and how to use it to build a web component.

We will also look at [Custom Elements Everywhere](#), a website that provides compatibility information for several different frameworks and component libraries.

Using Lit

```
import {html, css, LitElement} from 'lit';

export class SimpleGreeting extends LitElement {
  static styles = css`p { color: blue }`;

  static properties = {
    name: {type: String},
  };

  constructor() {
    super();
    this.name = 'Somebody';
  }

  render() {
    return html`<p>Hello, ${this.name}</p>`;
  }
}
```

```
customElements.define('simple-greeting', SimpleGree'Somebody'`<p>Hello, $
```

```
<!DOCTYPE html>
<head>
  <scrip<head>="module" src="./simple-greeting.js"></script>
</head>
<body>
  <simple-greeting name="World"></simple-greeting</script>
```

Rendering

```
import {LitElement, html} from 'lit';

class MyElement extends LitElement {
  render() {
    return html`<p>Hello from my template.</p>`;
  }
}
customElements.define('my-element', MyElement);
'my-element'
```

[Rendering](#)

Scoped Styles

```
import {LitElement, html, css} from 'lit';

export class MyElement extends LitElement {
  static styles = css`
    p {
      color: green;
    }
  `;
  render() {
    return html`<p>I am green!</p>`;
  }
}
```

```
}  
}  
customElements.define('my-element', MyElement);  
'my-element'`<p>I am green!</p>``<p>I am green!</p>`
```

[Styles](#) and [Scoped Styles](#) provide ways to style the custom element's content without fear that they will spill out of the components and where the parent styles will not override the component's

Lifecycle Events

[Lifecycle events](#)

Decorators

[Decorators](#) documentation

[Enabling decorators](#)

Reactivity

[Reactive properties](#)

Templates

[Templates](#)

Directives

[Built-in](#) and [custom](#) directives

Composition Strategies

[Composition strategies](#)

What can we use Lit for?

The project suggest three main use cases for Lit:

- Shareable components
- Design Systems
- Apps and sites

I will concentrate on the Design System because that's the one most intriguing to me.

[Custom Elements Everywhere](#)