
A RESOURCE FOR DESIGN SYSTEM AND ENGINEERING TEAMS

Lit and Storybook starter.



One button, every seam tested.

Install, run one command, read the output.

25

checks, each verified in the starter

19

browser tests: 12 unit, 7 stories

01

command runs all of it: `npm run check`

A starter is only useful if it runs.

Most web component starters stop at a counter that renders. The hard parts come later: a button inside a shadow root that does not submit its form, a manifest that lists private fields, a Storybook build that silently drops the element, tests in a simulated DOM that cannot see computed styles. This starter is small on purpose, one component, and it covers those seams with tests that run in a real browser.

The component is `ds-button`, the same one Field Guides 13 and 14 use. It reads Field Guide 03's semantic tokens through the shadow boundary, follows Field Guide 14's scoping rules (the checker reports zero findings on its styles), and ships the `@lit/react` wrapper that Field Guide 13's matrix recommends for React 18 consumers.

Every version and output in this guide comes from copying the kit to a clean directory, running `npm install` against the public registry and then `npm run check`: build, 19 tests, server render, Storybook build and a guard on the static build. It exited 0 on 24 September 2026.

Version 1.0 / Sources checked 24 September 2026

Field Guide 15 of the Design × AI series; pairs with 13 and 14. Verified 24 September 2026 on Node 22.22.3 and npm 10.9.8: Lit 3.3.3, Storybook 10.6.0, Vitest 4.1.11, Playwright 1.63.0 (Chromium 153), Vite 8.3.0, TypeScript 7.0.2, React 19.3.0.

A starting point, not a framework. One component, one theme pair, one browser in CI; add Firefox and WebKit instances before you rely on it. `@lit-labs/ssr` is a Lit Labs package. Prepared with AI assistance and edited by hand.

START HERE

What is in the box, and how to run it.

Copy the kit's `lit-storybook-starter` folder, then install and check. The first run may need `npx playwright install chromium`.

File	What it does
<code>src/ds-button.ts</code>	The Lit 3 component: form-associated, focus-delegating, token-driven.
<code>src/ds-button.styles.ts</code>	Shadow styles in three <code>@layer</code> s; semantic tokens and <code>--_</code> state only.
<code>src/tokens/semantic.css</code>	Field Guide 03's semantic tokens, light and dark. Loaded once, in the document.
<code>src/react/index.ts</code>	The <code>@lit/react</code> wrapper, 17 lines.
<code>src/ds-button.stories.ts</code>	Seven stories, one with a play function; every story is also a test.
<code>test/ds-button.test.ts</code>	Twelve browser tests: tokens, theme, forms, focus, React.
<code>custom-elements-manifest.config.mjs</code>	Analyzer config with a public-API-only plugin.
<code>scripts/ssr-render.mjs</code>	Renders the button with <code>@lit-labs/ssr</code> and measures the HTML.
<code>scripts/verify-storybook.mjs</code>	Fails if the static Storybook build lost an element registration.

terminal

```
npm install      # 222 packages, 15 s on the verification run
npm run check    # build, analyze, test, ssr, build-storybook, verify
npm run storybook # http://localhost:6006
```

Label	Meaning
<code>KIT TEST</code>	Proven by a test in <code>test/</code> or a story, in Chromium 153.
<code>BUILD</code>	Proven by a build step in <code>npm run check</code> exiting 0.
<code>SPEC</code>	Follows from a platform specification or a tool's own documentation.
<code>PRACTICE</code>	A working method with a review signal rather than a hard check.

SECTION 01

The component: small, native, token-driven.

What `ds-button` does that a first-draft Lit button usually does not.

Suggested owners: Component owner

☐ Tokens come from the document

L01 **KIT TEST**

The component reads `--ds-color-bg-brand` and friends; `semantic.css` sets them on `:root` and `[data-theme="dark"]`. No token import inside the shadow root.

Evidence: Tests read the computed background: `rgb(18, 91, 208)` in light, `rgb(128, 173, 255)` after `data-theme="dark"`, with no re-render.

Field Guide 14, rules T01 and T02.

☐ Variants switch private properties on `:host`

L02 **KIT TEST** **BUILD**

`tone` and `size` reflect to attributes, and `:host([tone="destructive"])` sets `--_bg` and `--_fg`. The inner button reads only `--_*`.

Evidence: Test: `tone="destructive"` computes `rgb(196, 43, 43)`. The Field Guide 14 checker reports 0 findings on `ds-button.styles.ts`.

Field Guide 14, T04.

☐ It is form-associated

L03 **KIT TEST** **SPEC**

`static formAssociated = true` and `attachInternals()`. Activation calls `form.requestSubmit()` or `form.reset()`; `formDisabledCallback` honours a disabled fieldset. `requestSubmit()` runs without a submitter, so the button adds no name or value to the form data.

Evidence: Tests: submits its light-DOM form; does not submit while loading; disabled by a disabled fieldset; `e1.form` returns the form.

web.dev, more capable form controls; MDN ElementInternals (S11, S12).

☐ A control test proves why

L04 **KIT TEST**

A naive element with a native `<button type="submit">` in its shadow root does not submit the outer form. Keep that test: it documents the reason for L03 and fails loudly if the platform ever changes.

Evidence: Test "a submit button inside a shadow root does not submit the outer form" passes.

Shoelace discussion 1057 (S16).

☐ **Focus goes to the real control**L05 **KIT TEST**

`shadowRootOptions` adds `delegatesFocus: true`, so `el.focus()`, labels and the Tab key land on the inner button, and `:focus-visible` styles apply there.

Evidence: Test: after `el.focus()`, `shadowRoot.activeElement` is the inner button. SSR output carries `shadowrootdelegatesfocus`.

Lit shadow DOM docs (S02).

☐ **Loading keeps the name and says it is busy**L06 **KIT TEST**

The spinner is `aria-hidden` and sits beside the slot, not in place of it, so the label stays the accessible name. The inner button gets `aria-busy` and `aria-disabled`, and activation is blocked.

Evidence: Test: loading sets `aria-busy="true"` and `aria-disabled="true"` and submits nothing.

Field Guide 02, Button contract: loading keeps the label for the accessible name.

☐ **No decorators, so any toolchain compiles it**L07 **BUILD** **SPEC**

`static properties` plus `declare` fields initialised in the constructor. No `experimentalDecorators`, no dependency on how a compiler implements decorators, and `useDefineForClassFields: false` keeps the fields reactive.

Evidence: `tsc` from TypeScript 7.0.2 and Vite 8.3.0 both compile the component with no decorator settings.

Lit reactive properties docs (S01).

The element carries the behaviour. Everything else is packaging.

SECTION 02

The manifest: the API that tools read.

`custom-elements.json` feeds Storybook's docs table, IDEs, Field Guide 13's JSX type generator and any agent reading your library. Make it describe the public API and nothing else.

Suggested owners: Component owner + design-system lead

☐ Generate it on every build

M01 **BUILD**

`cem analyze` with `litelement: true` in the config, run by `build`, `test`, `storybook` and `build-storybook`. Point `customElements` in `package.json` at it so tools find it.

Evidence: `@custom-elements-manifest/analyzer: Created new manifest.` in every `npm run check`.

CEM analyzer docs (S07).

☐ Write unions inline

M02 **BUILD**

With `declare tone: ButtonTone`, the manifest records the text `ButtonTone` and every consumer loses the allowed values. Written inline, the manifest carries `"primary" | "secondary" | "ghost" | "destructive"`; export aliases from the class for TypeScript users.

Evidence: The starter's first draft produced `ButtonTone`; the manifest and Storybook's docs table now list the four values.

Observed in the kit run.

☐ Publish the public API only

M03 **BUILD**

The analyzer lists `#internals`, `#onClick` and static fields, and Storybook shows them as controls. An 11-line `packageLinkPhase` plugin drops `#private` and `private` members.

Evidence: Manifest members after the plugin: `tone`, `size`, `type`, `disabled`, `loading`, `form`, `formDisabledCallback`.

Observed in the kit run; plugin API (S07).

☐ Document slots and parts in JSDoc

M04 **BUILD**

`@summary`, `@slot` and `@csspart control` on the class. They become the docs page description, the slots table and the parts table, and Field Guide 14's rule T16.

Evidence: Storybook's docs page lists one part, `control`, with its description.

S07; Custom Elements Manifest schema (S08).

If the manifest is wrong, every tool that reads it is wrong in the same way.

SECTION 03

Storybook 10: docs and tests in one.

Storybook 10 is ESM-only and builds web components with Vite. These checks are what it took to trust its output.

Suggested owners: Design-system lead + component owner

☐ Use the web-components-vite framework

SB01 **BUILD**

`framework: "@storybook/web-components-vite"`, stories in CSF with Lit `html` templates, `tags: ["autodocs"]` for the docs page.

Evidence: `Storybook build completed successfully`: 8 index entries (7 stories and a docs page).

Storybook web components docs (S03).

☐ Load the manifest in preview

SB02 **BUILD**

`setCustomElementsManifest(manifest)` in `.storybook/preview.ts`. In 10.6.0 that alone fills the docs table; issue 33038 reported a regression in 10.0.7.

Evidence: The static docs page shows descriptions, tone values, defaults and the part, all from the manifest.

Kit run; Storybook issue 33038 (S06).

☐ Make axe a failing test

SB03 **KIT TEST**

`a11y: { test: "error" }` in preview parameters, with `@storybook/addon-a11y`. axe sees inside open shadow roots.

Evidence: A temporary story with an empty `<ds-button>` failed with `Buttons must have discernible text (button-name)` (axe 4.13); removed afterwards.

Storybook accessibility testing (S05).

☐ Run every story as a test, on Vitest 4

SB04 **KIT TEST** **BUILD**

`@storybook/addon-vitest` turns each story and play function into a browser test. Its 10.6.0 release peers Vitest `^3 || ^4`, so npm refuses Vitest 5.0.1 (ERESOLVE). The starter pins 4.1.11.

Evidence: `stories (chromium)`: 7 of 7 pass, including the form-submit play function.

Storybook Vitest addon docs (S04); npm peer dependency metadata.

☐ Pre-bundle Lit once

SB05 **BUILD**

Without it, story tests log `Multiple versions of Lit loaded`. Adding `lit` and `lit/directive-helpers.js` to `optimizeDeps.include` in `viteFinal` removes the warning.

Evidence: No multiple-versions warning on two clean runs with the setting.

Lit multiple-versions message (S13).

Docs, controls and tests come from the same stories. Keep it that way.

SECTION 04

Tests: a real browser, and the seams.

The shadow boundary changes styling, focus and forms. Those are exactly the things a simulated DOM cannot compute, so the tests run in Chromium.

Suggested owners: Component owner

☐ Run component tests in a real browser

V01 **KIT TEST**

Vitest browser mode with the Playwright provider, headless Chromium. Two projects: `unit` for `test/**` and `stories` for the story files.

Evidence: `Test Files 2 passed (2)`, `Tests 19 passed (19)`.

Vitest browser mode (S09).

☐ Assert computed styles, not class names

V02 **KIT TEST**

Read `getComputedStyle(control).backgroundColor` after mounting in the document with the token sheet loaded. That proves the token reached the shadow root, which a class assertion cannot.

Evidence: Three token tests: light, dark and destructive, each an exact `rgb()`.

Field Guide 14, T01.

☐ Test React both ways

V03 **KIT TEST**

The wrapper sets props and forwards `onClick`. React 19 without a wrapper sets `tone` and `loading` as properties. If the element is defined after render, React 19 writes attributes instead; strings and `true` survive.

Evidence: Three React tests on React 19.3.0, including `loading=""` before definition and `el.loading === true` after.

React 19 custom element support (S15); Field Guide 13, N01 and N02.

☐ Expect the dev-mode warning in tests only

V04 **SPEC**

Vite's dev server resolves Lit's development build, which logs `Lit is in dev mode`. Production builds use the default production build. The warning in test output is expected; in a production bundle it is a bug.

Evidence: The warning appears in `vitest` output; the string is absent from every static Storybook asset.

Lit development and production builds (S13).

Test the seams where the platform behaves differently, and nothing else twice.

SECTION 05

Shipping: what consumers install.

The package is the product. Keep React out of non-React installs and the element in every bundle.

Suggested owners: Release owner

☐ An `exports` map with four doors

PK01 **BUILD**

`.`, `./react`, `./tokens.css` and `./custom-elements.json`. Nothing else is importable.

Evidence: Field Guide 13's `usage.react19.tsx` and `usage.react18.tsx` type-check against `ds-lit-starter` and `ds-lit-starter/react`.

Kit `package.json`.

☐ `sideEffects` names every file that registers, and a guard proves it

PK02 **BUILD**

A list without `src/ds-button.ts` let the bundle drop `customElements.define`: the static Storybook showed plain text while every dev-mode story test passed. `verify-storybook.mjs` now fails any build that never registers a manifest element.

Evidence: Bad list: `ds-button never registered`, exit 1. Fixed list: `1 element(s) registered`.

Observed in the kit run; bundler `sideEffects` semantics (S14).

☐ React is an optional peer

PK03 **BUILD**

An optional peer, imported only from `./react`. A Vue or CMS consumer installs Lit and `@lit/react`, never React.

Evidence: `dependencies` lists only `lit` and `@lit/react`.

Field Guide 13, W03.

☐ Smoke-test server rendering

PK04 **BUILD**

`@lit-labs/ssr` renders the built element to declarative shadow DOM. Measure it: styles are inlined per instance.

Evidence: `npm run ssr`: 3,087 bytes of HTML for one button, 2,671 of them its inline `<style>`; `delegatesFocus` serialized.

Lit SSR overview (S10); Field Guide 14, T22.

☐ One command in CI

PK05 **PRACTICE**

`npm run check` is the gate: build, analyze, 19 tests, SSR, Storybook build and the registration guard. The same command runs locally, in CI and in an agent's verify step.

Evidence: Exit 0 on the verification run of 24 September 2026.

Field Guide 03, R21.

A green test run in dev mode says nothing about the production bundle.

APPENDIX A

The component and its wrapper.

Excerpts from `src/ds-button.ts` and `src/react/index.ts`. The styles are in Field Guide 14, Appendix B.

`src/ds-button.ts` (excerpt)

```
export class DsButton extends LitElement {
  static formAssociated = true;
  static override shadowRootOptions: ShadowRootInit = {
    ...LitElement.shadowRootOptions, delegatesFocus: true };
  static override properties = {
    tone: { reflect: true }, loading: { type: Boolean, reflect: true } };
  declare tone: "primary" | "secondary" | "ghost" | "destructive";
  declare loading: boolean;
  readonly #internals: ElementInternals; // = this.attachInternals()

  #onClick(event: MouseEvent) {
    if (this.disabled || this.loading || this.#formDisabled) {
      event.preventDefault(); event.stopImmediatePropagation(); return;
    }
    if (this.type === "submit") this.#internals.form?.requestSubmit();
    else if (this.type === "reset") this.#internals.form?.reset();
  }

  override render() {
    return html`<button part="control" type="button"
      ?disabled=${this.disabled || this.#formDisabled} @click=${this.#onClick}
      aria-busy=${this.loading ? "true" : nothing}>${spinner}<slot></slot></button>`;
  }
}
```

Condensed for print; the full file has all five properties with JSDoc, `aria-disabled` while loading and a guarded `customElements.define`.

`src/react/index.ts`

```
import * as React from "react";
import { createComponent } from "@lit/react";
import { DsButton } from "../ds-button.js";

export const Button = createComponent({
  tagName: "ds-button",
  elementClass: DsButton,
  react: React,
  events: {},
  displayName: "Button",
});
```

APPENDIX B

Storybook and Vitest configuration.

The two files that took the most iterations; `.storybook/preview.ts` is three lines, covered by SB02 and SB03. Comments name the problem each setting solves.

`.storybook/main.ts`

```
const config: StorybookConfig = {
  stories: ["../src/**/*.stories.ts"],
  addons: ["@storybook/addon-docs", "@storybook/addon-a11y", "@storybook/addon-vitest"],
  framework: "@storybook/web-components-vite",
  core: { disableTelemetry: true },
  // One pre-bundled copy of Lit (SB05).
  viteFinal: async (config) => ({
    ...config,
    optimizeDeps: {
      ...config.optimizeDeps,
      include: [...(config.optimizeDeps?.include ?? []), "lit", "lit/directive-helpers.js"],
    },
  }),
};
```

`vitest.config.ts`

```
const chromium = () => ({ // a fresh object per project, or Vitest
  enabled: true, headless: true, // rejects the duplicate "chromium" project
  provider: playwright(),
  instances: [{ browser: "chromium" as const }],
});

export default defineConfig({
  test: {
    projects: [
      { test: { name: "unit", include: ["test/**/*.test.ts"], browser: chromium() } },
      { plugins: [storybookTest({ configDir: ".storybook" })],
        test: { name: "stories", browser: chromium() } },
    ],
  },
});
```

APPENDIX C

The verification run, and its versions.

`npm run check` on a clean copy of the kit, 24 September 2026. Test names are Vitest's verbose output.

```
npx vitest run --reporter=verbose (names only)
```

```
✓ unit    ds-button > renders a native button with the slotted label as its name
✓ unit    ds-button > reads semantic tokens through the shadow boundary
✓ unit    ds-button > follows the theme without re-rendering
✓ unit    ds-button > maps tone to semantic tokens, not raw values
✓ unit    ds-button > submits the surrounding light-DOM form when type is submit
✓ unit    ds-button > does not submit while loading, and says so to assistive tech
✓ unit    ds-button > is disabled by a disabled fieldset
✓ unit    ds-button > moves focus to the inner control (delegatesFocus)
✓ unit    a submit button inside a shadow root does not submit the outer form
✓ unit    React > the @lit/react wrapper sets props and forwards clicks
✓ unit    React > React 19 sets custom element properties without a wrapper
✓ unit    React > React 19 falls back to attributes when the element is defined after render
✓ stories Primary, Secondary, Ghost, Destructive, Loading, Theme Matrix, Submits Its Form
```

```
Test Files  2 passed (2)
Tests       19 passed (19)
ssr: 3087 bytes of HTML for one button, 2671 of them its inline <style>
storybook: 1 element(s) registered in the static build (ds-button)
```

The seven story tests are folded into one line here; Vitest prints one line each.

Package	Version	Role
<code>lit</code>	3.3.3	Component base class and templates
<code>@lit/react</code>	1.0.8	React wrapper
<code>@lit-labs/ssr</code>	4.1.0	Server-rendering smoke test (Labs)
<code>storybook</code> and <code>@storybook/*</code>	10.6.0	web-components-vite, addon-docs, addon-a11y, addon-vitest
<code>@custom-elements-manifest/analyzer</code>	0.11.0	custom-elements.json
<code>vitest</code> , <code>@vitest/browser-playwright</code>	4.1.11	Browser tests (pinned below 5, SB04)
<code>playwright</code>	1.63.0	Headless Chromium 153
<code>vite</code>	8.3.0	Dev server and Storybook builder
<code>typescript</code>	7.0.2	Build and declarations
<code>react</code> , <code>react-dom</code>	19.3.0	React tests; optional peer for consumers

KEEP WITH THE REPOSITORY

Leave a starter adoption record.

When you fork the starter into your system, record what you changed and what still proves it works.

Forked from / date / versions

Package name and exports

Token source (Field Guide 03 build?)

Components added since

Browsers in the test matrix

npm run check in CI (link)

Storybook static build and guard (link)

Wrapper route per Field Guide 13 profile

Dependencies pinned below latest, and why

Owner / next upgrade review

Pinned versions need a reason and a date.

Vitest is held at 4 because the Storybook Vitest addon does not accept 5 yet. Write that down, and re-test when Storybook ships a release that does.

SOURCES / MAINTENANCE

Keep the guide current.

Sources checked 24 September 2026. Versions are what `npm install` resolved from registry.npmjs.org that day; no lockfile ships, so later installs may resolve newer patches.

S01 / Lit, reactive properties

static properties, declare fields, class fields.

<https://lit.dev/docs/components/properties/>

S02 / Lit, working with shadow DOM

shadowRootOptions and delegatesFocus.

<https://lit.dev/docs/components/shadow-dom/>

S03 / Storybook, web components with Vite

Framework package and configuration.

<https://storybook.js.org/docs/get-started/frameworks/web-components-vite>

S04 / Storybook, Vitest addon

Stories as browser tests.

<https://storybook.js.org/docs/writing-tests/integrations/vitest-addon>

S05 / Storybook, accessibility testing

a11y test: error fails stories on axe violations.

<https://storybook.js.org/docs/writing-tests/accessibility-testing>

S06 / Storybook issue 33038

setCustomElementsManifest regression in 10.0.7.

<https://github.com/storybookjs/storybook/issues/33038>

S07 / Custom Elements Manifest analyzer

CLI, Lit support, JSDoc tags, plugins.

<https://custom-elements-manifest.open-wc.org/analyzer/getting-started/>

S08 / Custom Elements Manifest schema

What a manifest describes.

<https://github.com/webcomponents/custom-elements-manifest>

S09 / Vitest, browser mode

Real browsers through providers.

<https://vitest.dev/guide/browser/>

S10 / Lit, server-side rendering overview

@lit-labs/ssr status and output.

<https://lit.dev/docs/ssr/overview/>

S11 / web.dev, More capable form controls

Form-associated custom elements.

<https://web.dev/articles/more-capable-form-controls>

S12 / MDN, ElementInternals

Form association API.

<https://developer.mozilla.org/en-US/docs/Web/API/ElementInternals>

S13 / Lit, development and production builds

Dev mode is opt-in; production is default.

<https://lit.dev/docs/tools/development/>

S14 / webpack, tree shaking and sideEffects

Keeping files with side effects.

<https://webpack.js.org/guides/tree-shaking/>

S15 / React, React 19 release post

Custom element properties and events.

<https://react.dev/blog/2024/12/05/react-19>

S16 / Shoelace discussion 1057

Shadow-root buttons do not submit forms.

<https://github.com/shoelace-style/shoelace/discussions/1057>

Maintenance: re-run the verification at every Storybook, Vitest and Lit minor release; lift the Vitest 4 pin when the Storybook addon accepts 5. Update the PDF, HTML, Markdown and JSON together.