Tauri with SvelteKit Project Structure

Tauri with SvelteKit for reactive, lightweight desktop apps. Fast builds and minimal bundle size.

#tauri #rust #desktop #svelte #sveltekit

PNG PDF Copy Prompt

Project Directory

```
my-tauri-app/
    package.json
    svelte.config.js  SvelteKit config
    vite.config.ts
    tsconfig.json
  > src/  SvelteKit app
      app.html  HTML template
      app.css
  > routes/  File-based rout…
      +layout.svelte
      +page.svelte  Home page
    > settings/
        +page.svelte
  > lib/  $lib alias
    > components/
        Sidebar.svelte
        TitleBar.svelte
    > tauri/  Tauri bindings
        commands.ts
        events.ts
    > stores/
        app.ts  Svelte stores
    static/  Static assets
  > src-tauri/  Rust backend
      Cargo.toml
      tauri.conf.json
      build.rs
    > src/
        main.rs
        lib.rs
      > commands/
          mod.rs
          app.rs
      icons/
      capabilities/
```

Why This Structure?

SvelteKit brings file-based routing and SSR capabilities to Tauri. Svelte's compile-time reactivity means smaller bundles and faster runtime. The `$lib` alias organizes shared code. Perfect for complex UIs that need to stay lightweight.

Key Directories

src/routes/ - File-based routing with +page.svelte files

src/lib/ - Shared components, stores, and Tauri bindings

src/lib/tauri/ - Typed command and event wrappers

src/lib/stores/ - Svelte stores for reactive state

Getting Started

1. `npm create tauri-app@latest -- --template sveltekit`
2. `cd my-tauri-app && npm install`
3. Configure SvelteKit adapter-static in `svelte.config.js`
4. `npm run tauri dev`

When To Use This

- Complex UIs with routing requirements
- Apps prioritizing small bundle size
- Teams familiar with Svelte ecosystem
- Progressive enhancement from web app

Trade-offs

SSR disabled - Must use adapter-static for Tauri

Routing overhead - Simpler apps may not need file routing

Smaller ecosystem - Fewer UI libraries than React/Vue