Svelte Compiled UI framework

CONCEPTS

Svelte is a compiler: .svelte files hold script, markup and scoped style. Reactivity uses runes ($-prefixed), which are language keywords and need no import.

SvelteKit routing, load, SSR, adapters
Vite dev server and build
Svelte components, runes

This sheet follows Svelte 5.57 and SvelteKit 3. On SvelteKit 2, config is in svelte.config.js, $lib and $app/stores still exist, and matchers live in src/params/; npx sv migrate sveltekit-3 upgrades.

PROJECT (SV CLI)

`npx sv create <dir>`
Scaffold a new project

`npx sv add`
Pick add-ons interactively

`npx sv add tailwindcss`
Add an official add-on

`npm i -D svelte-check`
Install the checker once

`npx sv check`
Type and a11y check the project

DEV AND BUILD

`npm run dev`
Start the Vite dev server

`npm run build`
Production build (runs the adapter)

`npm run preview`
Serve the production build locally

COMPONENTS

`<script lang="ts">`
Enable TypeScript in a component

`<style>`
Scoped CSS for this component

`<Child {prop} />`
Pass a variable named prop

`{@render children?.()}`
Render nested content

COMPONENTS (cont.)

`{#snippet name(arg)}…{/snippet}`
Reusable markup chunk

`{@render name(arg)}`
Render a snippet

RUNES

`let n = $state(0)`
Reactive state

`let d = $derived(expr)`
Value computed from state

`let d = $derived.by(() => {…})`
Derived with statements

`$effect(() => {…})`
Run after DOM updates, on changes

`$effect(() => { return () => {…} })`
Cleanup before re-run or destroy

`let { a, b = 1 } = $props()`
Props with a default

`let { v = $bindable() } = $props()`
Allow bind: from the parent

Prefer $derived over setting state inside $effect. Reach for $effect only for side effects like DOM, timers or subscriptions.

TEMPLATE SYNTAX

`{expr}`
Output a value

`{#if c}…{:else if d}…{:else}…{/if}`
Conditional block

`{#each items as item, i (item.id)}`
Loop with index and key

`onclick={fn}`
Event attribute, no on: prefix

`onclick={() => n++}`
Inline handler

BINDINGS

`bind:value={v}`
Two-way input value

`bind:checked={b}`
Checkbox state

SVELTEKIT ROUTES

`src/routes/+page.svelte`
Page for this URL

`src/routes/+layout.svelte`
Wrapper; must render children

`+page.js / +page.server.js`
load function (universal / server)

`+server.js`
API endpoint (GET, POST, …)

`src/routes/blog/[slug]`
Dynamic segment

`src/lib/`
Shared code, server/ is server-only

`import x from '#lib/x.js'`
Import from lib (Kit 3; $lib in Kit 2)

DATA AND FORMS

`export const load = ({ params, fetch }) => {…}`
Load data for the page

`let { data } = $props()`
Receive load data in the page

`export const actions = { default }`
Form actions in +page.server.js

`use:enhance`
Progressive form submission

`redirect(303, '/path')`
Redirect from load or action

`goto('/path')`
Navigate from code ($app/navigation)

`import { page } from '$app/state'`
Reactive page url, params, data

RECIPES

New app, ready to check

```
npx sv create --template minimal
--types ts my-app
cd my-app
npm i -D svelte-check
npm run dev -- --open
```

Counter component

```
<script>
  let count = $state(0);
  let double = $derived(count *
2);
</script>
<button onclick={() => count++}
>{count} / {double}</button>
```