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.

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

`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

RUNES

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

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

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

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

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

BINDINGS

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

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)

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

DATA AND FORMS

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

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