# REFERENSI CEPAT SVELTE

Komponen, reaktivitas, store, transisi, SvelteKit

## Komponen

### Komponen Dasar

```
<script>
  let name = "world";
</script>
<h1>Hello {name}!</h1>
<style>
  h1 { color: purple; }
</style>
```

### Struktur Komponen

| | |
|---|---|
| `<script>` | Logika komponen (JS/TS) |
| `Markup` | Template HTML dengan `{expressions}` |
| `<style>` | CSS yang di-scope (otomatis per komponen) |
| `<script context="module">` | Dijalankan sekali per modul, bukan per instance |

## Reaktivitas

### Penugasan Reaktif

```
<script>
  let count = 0;
  function increment() { count += 1; } // triggers re-render
</script>
<button on:click={increment}>{count}</button>
```

### Deklarasi Reaktif

```
<script>
  let width = 10;
  let height = 5;
  $: area = width * height;       // recomputes when deps change
  $: console.log("area is", area); // reactive statement
</script>
```

$: menandai deklarasi dan pernyataan reaktif

### Aturan Reaktivitas

| | |
|---|---|
| `Assignment triggers` | `count += 1` memicu update; `obj.x = 1` juga |
| `Array mutation` | Gunakan `arr = [...arr, item]` (reassign untuk memicu) |
| `$: declaration` | Dihitung ulang otomatis ketika variabel yang dirujuk berubah |
| `$: statement` | Menjalankan efek samping secara reaktif |

## Props

### Deklarasi & Pemberian Props

```
<!-- Child.svelte -->
<script>
  export let name;
  export let greeting = "Hello";  // default value
</script>
<p>{greeting}, {name}!</p>

<!-- Parent.svelte -->
<Child name="Alice" />
```

### Spread Props

```
<script>
  const props = { name: "Alice", greeting: "Hi" };
</script>
<Child {...props} />
```

## Event

### Event DOM

```
<button on:click={handleClick}>Click</button>
<input on:input={(e) => value = e.target.value} />
<form on:submit|preventDefault={handleSubmit}>
```

### Modifier Event

| | |
|---|---|
| `preventDefault` | Memanggil `e.preventDefault()` |
| `stopPropagation` | Menghentikan event bubbling |
| `once` | Handler hanya dipanggil sekali |
| `self` | Hanya jika `event.target` adalah elemen itu sendiri |
| `capture` | Gunakan fase capture |

### Event Komponen

```
<!-- Child.svelte -->
<script>
  import { createEventDispatcher } from "svelte";
  const dispatch = createEventDispatcher();
</script>
<button on:click={() => dispatch("greet", { text: "hi" })}>

<!-- Parent.svelte -->
<Child on:greet={(e) => alert(e.detail.text)} />
```

## Binding

### Two-Way Binding

```
<input bind:value={name} />
<input type="checkbox" bind:checked={agreed} />
<select bind:value={selected}>
  <option value="a">A</option>
</select>
```

### Binding Elemen & Komponen

```
<div bind:this={element}></div>
<canvas bind:clientWidth={w} bind:clientHeight={h}></canvas>
<Child bind:value={childValue} />
```

### Jenis Binding

| | |
|---|---|
| `bind:value` | Nilai input/select/textarea |
| `bind:checked` | Status checkbox |
| `bind:group` | Grup radio/checkbox |
| `bind:this` | Referensi elemen DOM |
| `bind:clientWidth/Height` | Dimensi elemen (hanya baca) |

## Store

### Writable Store

```
// store.js
import { writable } from "svelte/store";
export const count = writable(0);

// Component — auto-subscribe with $
<script>
  import { count } from "./store.js";
</script>
<button on:click={() => $count += 1}>{$count}</button>
```

### Metode Store

```
count.set(10);              // set value
count.update(n => n + 1);   // update from current
const unsub = count.subscribe(v => console.log(v));
```

### Jenis Store

| | |
|---|---|
| `writable(val)` | Store baca-tulis |
| `readable(val, fn)` | Hanya baca, diset oleh fungsi start |
| `derived(stores, fn)` | Dihitung dari store lain |
| `$store` | Sintaks auto-subscribe dalam komponen |

## Transisi

### Transisi Bawaan

```
<script>
  import { fade, slide, fly } from "svelte/transition";
  let visible = true;
</script>
{#if visible}
  <div transition:fade>Fades in/out</div>
  <div in:fly={{ y: 200 }} out:fade>Fly in, fade out</div>
{/if}
```

### Opsi Transisi

| | |
|---|---|
| `fade` | Opacity 0 ke 1 |
| `fly` | Animasi offset x/y + opacity |
| `slide` | Masuk/keluar dengan geser (tinggi) |
| `scale` | Skala dan fade |
| `draw` | Animasi stroke SVG path |
| `duration` | Durasi transisi dalam ms |
| `delay` | Tunda sebelum dimulai |

## Slot

### Slot Default & Bernama

```
<!-- Card.svelte -->
<div class="card">
  <slot name="header">Default header</slot>
  <slot>Default content</slot>
</div>

<!-- Usage -->
<Card>
  <h2 slot="header">Title</h2>
  <p>Body content goes here</p>
</Card>
```

### Slot Props

```
<!-- List.svelte -->
{#each items as item}
  <slot {item} index={item.id} />
{/each}

<!-- Usage -->
<List {items} let:item let:index>
  <p>{index}: {item.name}</p>
</List>
```

## Context

### Set & Get Context

```
<!-- Parent.svelte -->
<script>
  import { setContext } from "svelte";
  setContext("theme", { color: "dark" });
</script>

<!-- Descendant.svelte -->
<script>
  import { getContext } from "svelte";
  const theme = getContext("theme"); // { color: "dark" }
</script>
```

### Context vs Store

| | |
|---|---|
| `Context` | Di-scope ke pohon komponen, tidak reaktif secara default |
| `Stores` | Global, reaktif, bisa diimpor dari mana saja |
| `Context + Store` | Teruskan store via context untuk reaktivitas yang di-scope |

## Dasar SvelteKit

### Routing Berbasis File

```
src/routes/
  +page.svelte          <!-- / -->
  about/+page.svelte    <!-- /about -->
  blog/[slug]/+page.svelte  <!-- /blog/:slug -->
```

### Fungsi Load

```
// +page.js (runs on client & server)
export async function load({ params, fetch }) {
  const res = await fetch(`/api/posts/${params.slug}`);
  return { post: await res.json() };
}
```

### File Penting

| | |
|---|---|
| `+page.svelte` | Komponen halaman |
| `+page.js / +page.ts` | Fungsi load client/universal |
| `+page.server.js` | Load khusus server / form actions |
| `+layout.svelte` | Pembungkus layout bersama |
| `+error.svelte` | Halaman error |
| `+server.js` | API endpoint (GET, POST, ...) |