### SVELTE 5

What's new, and why did it change?

## WHY??

# REACTIVITY ABSTRACTION - FINE GRAINED REACTIVITY?

```
$:
<!--->
```

#### **CAN YOU SPOT THE BUG?**

```
1 <script>
2  let count = 1;
3  $: double = count * 2;
4  console.log(double);
5 </script>
6
7 <button on:click={() => {
8  count++;
9  console.log(count, double);
10 }}>count: {count}, double: {double}</button><!--->
```

#### Solution

#### **CAN YOU SPOT THE BUG?**

```
1 <script>
2  let count = 1;
3  $: double = count * 2;
4  console.log(double);
5 </script>
6
7 <button on:click={() => {
8  count++;
9  console.log(count, double);
10 }}>count: {count}, double: {double}</button><!--->
```

#### Solution

#### **CAN YOU SPOT THE BUG?**

```
1 <script>
2  let count = 1;
3  $: double = count * 2;
4  console.log(double);
5 </script>
6
7 <button on:click={() => {
8  count++;
9  console.log(count, double);
10 }}>count: {count}, double: {double}</button><!--->
```

#### Solution

#### **RUNES**

A letter or mark used as a mystical or magic symbol.

Runes have a "\$" prefix and look like functions:

```
<script>
   let message = $state('hello');
</script>
<!--->
```

Now you can also use svelte runes in ".svelte.js" or ".svelte.ts", it's not only limited to ".svelte" files

#### **RUNES STATE - CLASSES**

```
1 class Todo {
    done = $state(false);
3 text = $state();
5 constructor(text) {
      this.text = text;
9 reset() {
10 this.text = '';
this.done = false;
12 }
13 }
14 <!--->
```

#### SIDE EFFECT & DERIVED STATE

In Svelte 5, there is a separation between side effects and derived state

Prior we used the "\$:" to express side effects and derived states

#### **RUNES DERIVED**

```
1 <script>
2  let count = $state(0);
3  let doubled = $derived(count * 2);
4 </script>
5
6 <button onclick={() => count++}>
7  {doubled}
8 </button>
9
10 {count} doubled is {doubled}
11 <!--->
```

Back to problem

#### **RUNES EFFECTS**

```
1 <script>
2 let size = $state(50);
    let color = $state('#ff3e00');
 4
 5
     let canvas;
6
     $effect(() => {
8
       const context = canvas.getContext('2d');
       context.clearRect(0, 0, canvas.width, canvas.height);
10
11
       // this will re-run whenever 'color' or 'size' change
12
       context.fillStyle = color;
       context.fillRect(0, 0, size, size);
13
14
     });
```

avoid overusing it!!

#### **SNIPPETS REPLACE "SLOTS"**

```
1 <script>
2   let { children } = $props();
3 </script>
4   {@render children?.()}
6 <!--->
```

# SNIPPETS CREATE REUSABLE CHUNKS OF MARKUP

```
1 <script>
2  let { message = 'it's great to see you!' } = $props();
3 </script>
4
5 {#snippet hello(name)}
6  hello {name}! {message}!
7 {/snippet}
8
9 {@render hello('alice')}
10 {@render hello('bob')}
11 <!--->
```

#### **EVENTS**

```
0
Pump.svelte
<script>
  import { createEventDispatcher } from 'svelte';
  const dispatch = createEventDispatcher();
 let { inflate, deflate } = $props();
 let power = $state(5);
</script>
<button onclick={() => dispatch('inflate', power)inflate(power)}>
  inflate
</button>
<button onclick={() => dispatch('deflate', power)deflate(power)}>
  deflate
</button>
<button onclick={() => power--}>-</button>
Pump power: {power}
<button onclick={() => power++}>+</button>
```

#### **COMPATIBILITY**

It is possible to use Svelte 3 components in Svelte 5 (but probably not in Svelte 6)

```
"devDependencies": {
   "@sveltejs/vite-plugin-svelte": "^3.0.0",
   "svelte": "^4",
   // ...
}<!--->
  "devDependencies": {
    "@sveltejs/vite-plugin-svelte": "^4.0.0",
   "svelte": "^5",
```

#### **MIGRATION**

From Svelte 4 to Svelte 5, there is a migration script

```
1 npx sv migrate
2 <!--->
```

#### **PERFORMANCE**

Efficient code with signals

#### **PERFORMANCE**

#### Efficient code with signals

#### Duration in milliseconds ± 95% confidence interval (Slowdown = Duration / Fastest)

Name Duration for	vanillajs- lite	vanillajs-3	vanillajs	solid- v1.8.15	svelte- v5.0.5	svelte- classic- v5.0.5	vue-v3.5.3	angular-cf- nozone- v18.0.1	angular-cf- v18.0.1	angular-cf- signals- v18.0.1	vue-jsx- v3.4.29	react- classes- v18.2.0	react- compiler- hooks- v19.0.0- rc- 4c58fce7- 20240904
Implementation notes	772	772	772										
Implementation link	code	code	code	code	code	code	code	code	code	code	code	code	code
create rows creating 1,000 rows. (5 warmup runs).	35.3 ± 0.3 (1.00)	36.6 ± 0.3 (1.04)	36.2 ± 0.3 (1.03)	38.2 ± 0.3 (1.08)	38.8 ± 0.2 (1.10)	39.7 ± 0.3 (1.12)	45.3 ± 0.3 (1.28)	47.4 ± 0.3 (1.34)	48.1 ± 0.2 (1.36)	48.0 ± 0.3 (1.36)	45.0 ± 0.2 (1.27)	48.5 ± 0.3 (1.37)	47.3 ± 0.4 (1.34)
replace all rows updating all 1,000 rows. (5 warmup runs).	39.2 ± 0.3 (1.00)	39.9 ± 0.3 (1.02)	39.7 ± 0.4 (1.01)	43.8 ± 0.2 (1.12)	43.7 ± 0.3 (1.11)	45.8 ± 0.3 (1.17)	50.5 ± 0.4 (1.29)	55.2 ± 0.2 (1.41)	58.7 ± 0.5 (1.50)	59.3 ± 0.3 (1.51)	50.9 ± 0.6 (1.30)	54.6 ± 0.3 (1.39)	54.3 ± 0.2 (1.39)
partial update updating every 10th row for 1,000 row. (3 warmup runs). 4 x CPU slow- down.	16.4 ± 0.4 (1.00)	17.0 ± 0.6 (1.04)	17.2 ± 0.3 (1.05)	18.0 ± 0.4 (1.10)	18.3 ± 0.3 (1.12)	19.4 ± 0.4 (1.18)	21.0 ± 0.4 (1.28)	19.3 ±0.3 (1.18)	20.6 ± 0.3 (1.26)	19.9 ± 0.5 (1.21)	29.2 ± 0.5 (1.78)	23.9 ± 0.6 (1.46)	23.0 ± 0.4 (1.40)
select row highlighting a selected row. (5 warmup runs). 4 x CPU slowdown.	3.1 ±0.1 (1.03)	3.1 ±0.2 (1.03)	3.3 ±0.2 (1.10)	3.5 ±0.2 (1.17)	4.4 ±0.2 (1.47)	4.2 ± 0.2 (1.40)	5.0 ± 0.2 (1.67)	5.6 ± 0.3 (1.87)	6.3 ±0.2 (2.10)	6.5 ± 0.2 (2.17)	13.3 ± 0.3 (4.43)	7.8 ±0.2 (2.60)	7.7 ±0.2 (2.57)
swap rows swap 2 rows for table with 1,000 rows. (5 warmup runs). 4 x CPU slowdown.	19.9 ± 0.3 (1.01)	20.5 ± 0.4 (1.04)	19.7 ± 0.3 (1.00)	21.4 ± 0.3 (1.09)	21.5 ± 0.4 (1.09)	22.8 ± 0.3 (1.16)	22.7 ± 0.5 (1.15)	22.5 ± 0.3 (1.14)	23.4 ± 0.4 (1.19)	23.7 ± 0.3 (1.20)	31.4 ± 0.4 (1.59)	176.2 ± 2.1 (8.94)	176.9 ± 1.2 (8.98)
removing one row. (5 warmup runs). 2 x CPU slowdown.	16.0 ± 0.2 (1.06)	16.0 ± 0.2 (1.06)	16.3 ± 0.4 (1.08)	16.5 ± 0.3 (1.09)	16.7 ± 0.5 (1.11)	17.1 ± 0.2 (1.13)	19.7 ±0.2 (1.30)	17.6 ± 0.1 (1.17)	17.8 ± 0.2 (1.18)	19.6 ± 0.3 (1.30)	21.8 ± 0.2 (1.44)	18.8 ±0.2 (1.25)	18.8 ± 0.2 (1.25)
create many rows creating 10,000 rows. (5 warmup runs).	363.4 ± 1.0 (1.00)	378.9 ± 1.8 (1.04)	374.2 ± 0.8 (1.03)	395.8 ± 2.6 (1.09)	398.2 ± 1.0 (1.10)	411.4 ± 1.3 (1.13)	443.7 ± 1.5 (1.22)	492.3 ± 1.0 (1.35)	497.0 ± 1.1 (1.37)	499.7 ± 1.1 (1.38)	448.8 ±2.2 (1.24)	615.0 ± 3.7 (1.69)	611.0 ±2.7 (1.68)
append rows to large table appending 1,000 to a ta- ble of 1,000 rows. (5 warmup runs).	41.4 ± 0.5 (1.00)	42.1 ± 0.4 (1.02)	43.2 ± 0.2 (1.04)	45.0 ± 0.3 (1.09)	45.0 ± 0.2 (1.09)	47.7 ± 0.3 (1.15)	51.9 ± 0.3 (1.25)	55.3 ± 0.3 (1.34)	55.8 ± 0.2 (1.35)	55.9 ± 0.4 (1.35)	56.8 ± 0.4 (1.37)	54.6 ± 0.3 (1.32)	54.4 ± 0.3 (1.31)
clear rows clearing a table with 1,000 rows. (5 warmup runs). 4 x CPU slow- down.	13.5 ± 0.2 (1.10)	13.8 ± 0.1 (1.12)	13.5 ± 0.3 (1.10)	15.1 ±0.2 (1.23)	14.8 ± 0.3 (1.20)	14.1 ± 0.3 (1.15)	19.5 ± 0.6 (1.59)	24.6 ± 0.3 (2.00)	30.7 ± 0.5 (2.50)	30.8 ± 0.3 (2.50)	17.7 ± 0.3 (1.44)	19.5 ± 0.5 (1.59)	28.3 ± 0.5 (2.30)
weighted geometric mean of all factors in the ta- ble	1.02	1.04	1.05	1.11	1.13	1.16	1.31	1.37	1.45	1.46	1.48	1.55	1.60
compare: Green means significantly faster, red significantly slower	compare	compare	compare	compare	compare	compare	compare	compare	compare	compare	compare	compare	compare