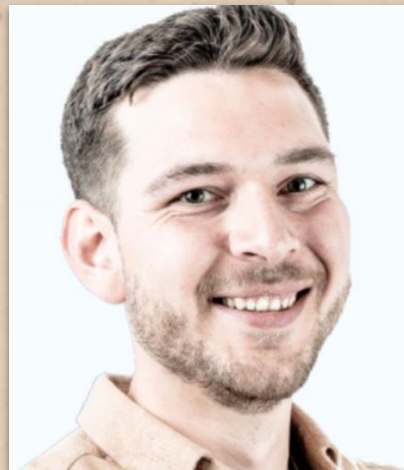# Gather your Party with Svelte

Who Are We?

Dag Erik Løvgren

Frontend developer @ Miles Bergen
@Work: Loves clean, reusable code, and new technologies
@Home: Into RPGs, board games, and woodworking

Alexander Castillo

Fullstack developer @ Miles Bergen
@Work: Loves clean, reusable code, and new technologies & copy pasting
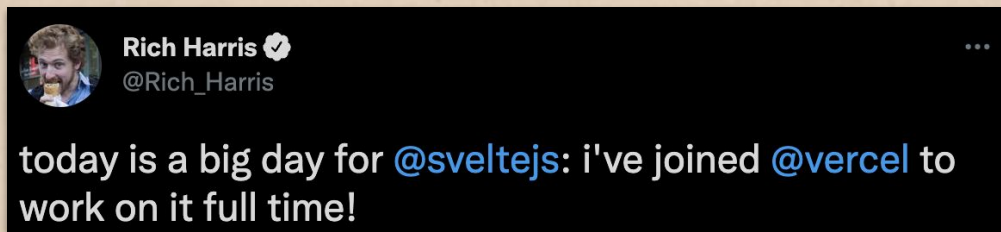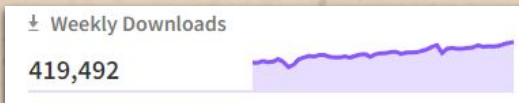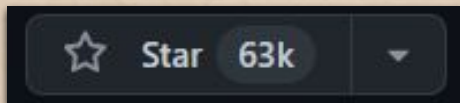@Home: Into gaming, climbing, brewing beer and the great outdoors

Yet Another Frontend Framework?!

- Well, yes, it's an alternative

- Not really a framework, but a *compiler*
- Motivation for creating Svelte
  - Performance
  - Animation-heavy interfaces
  - Write less, do more
  - Embedded devices
- No Virtual DOM
  - But the Virtual DOM is fast, isn't it?
- Reactive
  - *"The best API is no API at all."*

Miles

Svelte in the Real World

State of JS 2021 Satisfaction:

- **Vercel** documented support

- **Netlify** documented support

- **Sanity** has a SvelteKit Template

Rich Harris
@Rich_Harris

today is a big day for @sveltejs: i've joined @vercel to work on it full time!

Miles
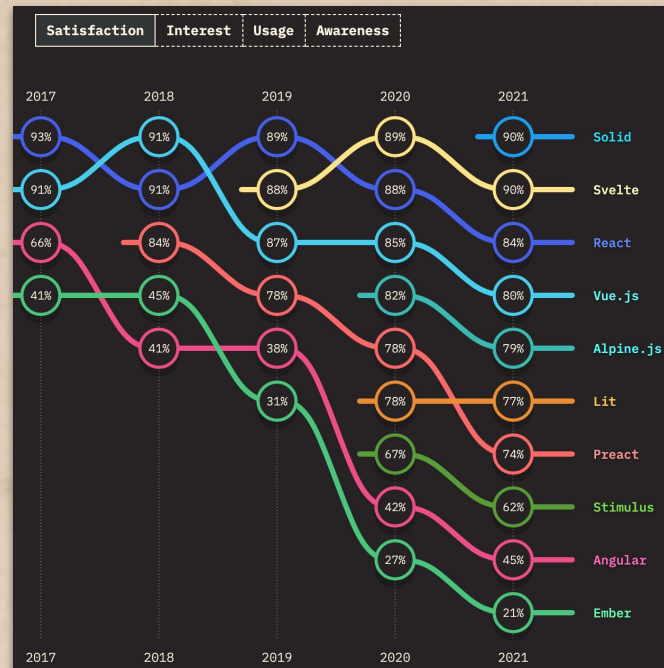
Svelte Core Features

Covered in this workshop:

- Components

- Reactivity

- Blocks

- Bindings and Events

- Lifecycle

- Motion

- Transitions

- Animations

Not covered in this workshop:

- Stores

  - Subscribable global objects

  - https://svelte.dev/tutorial/writable-stores

- Actions

  - Element-level lifecycle functions

  - https://svelte.dev/tutorial/actions

- Context API

  - Inherit state in child components

  - https://svelte.dev/tutorial/context-api

Miles

Components

Reactivity

```ts
/// Usage:
<Button hint="< Click me" small={false}>Click me!</Button>
/// Button.svelte:
<script lang="ts">
 import ClickIcon from './ClickIcon.svelte';

 export let hint = '';
 export let small: boolean;
 export let disabled: boolean;
</script>

<button class="my-button" class:small {disabled} on:click>
 <slot />
 <ClickIcon />
</button>
<span>{hint}</span>

<style>
 .my-button {
   border-radius: 4px;
   padding: 1rem 2rem;
 }
 .small {
   padding: 0.25rem;
 }
</style>
```

```ts
<script lang="ts">
 let count = 0;

 const increment = () => {
   count += 1;
 };

 $: sum = count * 2;
</script>

<button on:click={increment}>Click me to add 2</button>

<div>Clicked {count} {count === 1 ? 'time' : 'times'}</div>

<div>Sum: {sum}</div>

<style>
 * {
   margin-bottom: 1rem;
 }
</style>
```

Click me! < Click me

Click me to add 2

Clicked 0 times

Sum: 0

Miles

Blocks

Bindings and Events

```ts
/// Usage:
<PortraitPreview id="portrait_11.jpg" tags={['Human', 'Bard']} />
/// PortraitPreview.svelte:
<script lang="ts">
 import { apiFetch } from '$lib/utils/api-fetch';

 export let id: string;
 export let tags: string[] = [];
</script>

<div class="container">
 {#if id}
   {#await apiFetch(`/api/portraits/${id}`)}
     Loading...
   {:then portrait}
     <img src={portrait} alt="Portrait" />
   {:catch error}
     Could not fetch portrait image.
   {/await}
 {/if}
</div>

{#each tags as tag (tag)}
 <span>({tag}) </span>
{/each}
```



```ts
/// Usage:
<NameInput
 on:change={({ detail: { value } }) ⇒ console.log(value)}
 on:you-shall-not-enter={() ⇒ alert('No!!!')}
/>
/// NameInput.svelte:
<script lang="ts">
 import { createEventDispatcher } from 'svelte';
 const dispatch = createEventDispatcher();

 let name = 'world';

 const handleChange = () ⇒ {
   dispatch('change', { value: name });
 };
 const handleEnter = (event) ⇒ {
   if (event.key ≡ 'Enter') {
     dispatch('you-shall-not-enter');
   }
 };
</script>

<input bind:value={name} on:input={handleChange}
on:keyup={handleEnter} />

<p>Hello {name}!</p>
```


Miles

Lifecycle

```ts
<script lang="ts">
 import { onDestroy, onMount } from 'svelte';
 import { apiFetch } from '$lib/utils/api-fetch';

 let portrait: string;

 let counter = 0;
 const interval = setInterval(() ⇒ {
   counter += 1;
 }, 1000);

 onMount(async () ⇒ {
   portrait = await apiFetch('/api/portraits/portrait_15.jpg');
 });
 onDestroy(() ⇒ {
   clearInterval(interval);
 });
</script>

{#if portrait}
 <img src={portrait} alt="Portrait" />
{/if}

<p>This component has been mounted for {counter} seconds.</p>
```

THIS COMPONENT HAS BEEN MOUNTED FOR 0 SECONDS.

Motion

```ts
<script lang="ts">
 import { cubicOut } from 'svelte/easing';
 import { tweened } from 'svelte/motion';

 const MIN = 8;
 const MAX = 25;
 const DIFF = MAX - MIN;

 const strength = tweened(0, {
   duration: 800,
   easing: cubicOut,
 });
</script>

<button on:click={() ⇒ strength.set(MIN + Math.random() *
DIFF)}>Randomize Strength</button>

<progress value={($strength - MIN) / DIFF} />
Strength: {Math.round($strength)}
```

RANDOMIZE STRENGTH   STRENGTH: 0

Miles

Transitions

Animations

```ts
<script lang="ts">
 import { fly } from 'svelte/transition';

 const scale = (node, { delay = 0, duration = 2000 }) => ({
   delay,
   duration,
   css: (t, u) => `
     font-size: calc(1em * ${u * 2 + 1});
     opacity: ${t};
   `
 });

 let visible = false;
</script>

<button on:click={() => (visible = !visible)}>Say {visible ?
'goodbye' : 'hello'}</button>

{#if visible}
 <p in:fly={{ x: -200, duration: 2000 }} out:scale>Hello
there</p>
{/if}
```

```ts
<script lang="ts">
 import { flip } from 'svelte/animate';
 import { crossfade } from 'svelte/transition';

 let todos = ['create a hero', 'level up', 'take first boss'];

 // NB: This `crossfade` is a transition, `flip` is the animation
 const [send, receive] = crossfade({
   duration: (d) => Math.sqrt(d * 200),
   fallback: () => ({
     duration: 600,
     css: (t) => `transform: scale(${t}); opacity: ${t}`,
   }),
 });
</script>

{#each todos as todo (todo)}
 <button
   in:receive={{ key: todo }}
   out:send={{ key: todo }}
   animate:flip
   on:click={() => (todos = todos.filter((e) => e !== todo))}
 >
   {todo}
 </button>
{/each}
```
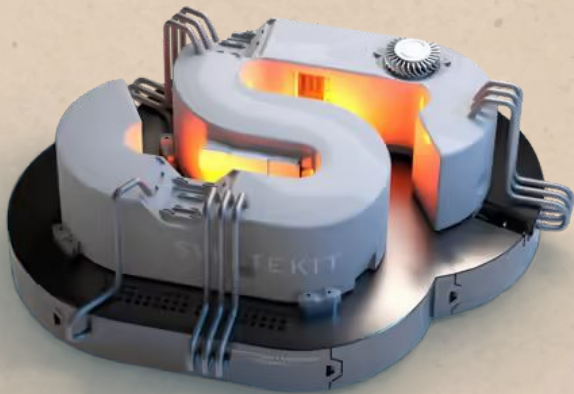
SAY HELLO

CREATE A HERO   LEVEL UP   TAKE FIRST BOSS

Miles

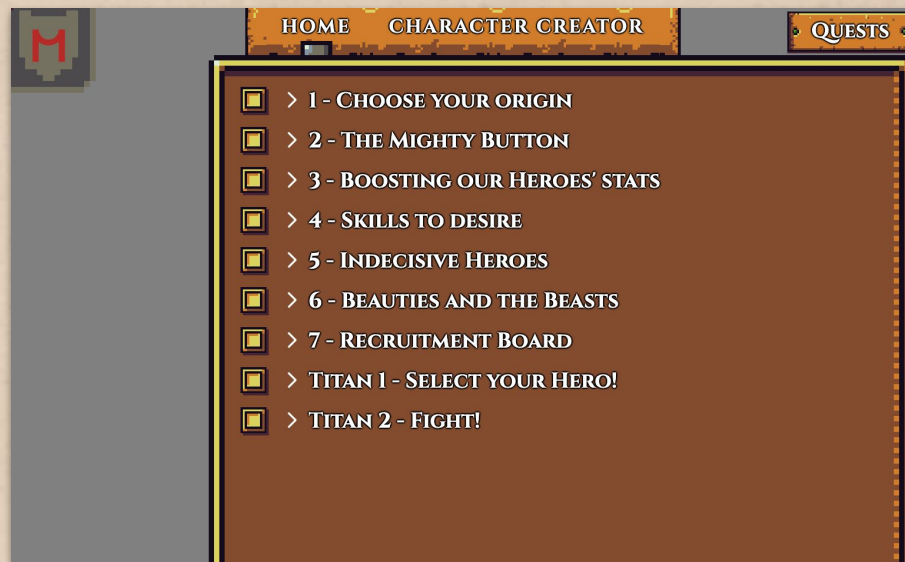# SvelteKit - There's another layer?

- Application framework for Svelte
- Similar to Next.js for React
- Stable release!
- Uses Vite

- Features:
  - Server-side Rendering (SSR)
  - Filesystem routing
  - SPA-like navigation
  - Code-splitting
  - Offline support
  - Hot Module Replacement (HMR)

Miles

Workshop Description

Glorious Quests! - Quest log

Alternative:
workshop-tasks.md

> 1 - CHOOSE YOUR ORIGIN
> 2 - THE MIGHTY BUTTON
> 3 - BOOSTING OUR HEROES' STATS
> 4 - SKILLS TO DESIRE
> 5 - INDECISIVE HEROES
> 6 - BEAUTIES AND THE BEASTS
> 7 - RECRUITMENT BOARD
> TITAN 1 - SELECT YOUR HERO!
> TITAN 2 - FIGHT!

HOME   CHARACTER CREATOR   QUESTS

Glorious Quests! - Quests

Quest files - unscoped

```
/**
 * Quest 1 - Choose your origin
 */
```
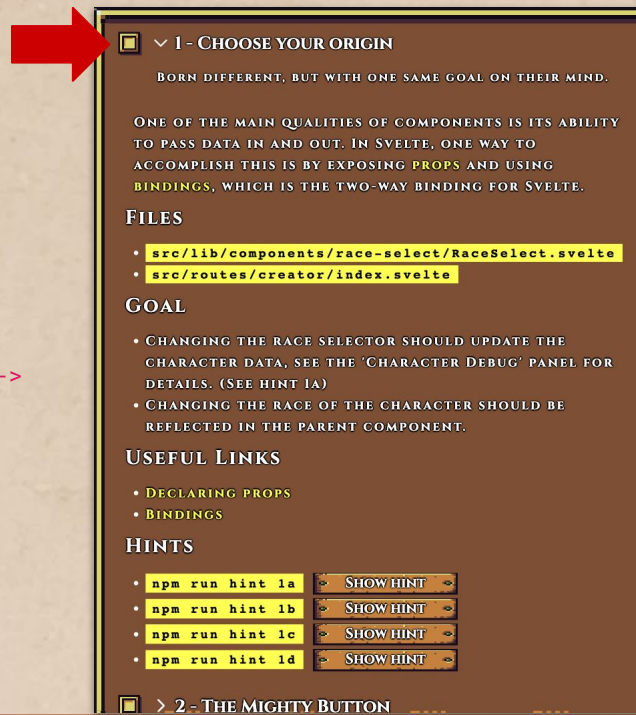
Quest areas - scoped

```
<!-- Quest 7 - Recruitment Board: Enter HTML markup here START -->

<!-- Quest 7 - Recruitment Board: Enter HTML markup here END -->
```

Quest indicators

1 - Choose your origin

Born different, but with one same goal on their mind.

One of the main qualities of components is its ability to pass data in and out. In Svelte, one way to accomplish this is by exposing PROPS and using BINDINGS, which is the two-way binding for Svelte.

Files

- `src/lib/components/race-select/RaceSelect.svelte`
- `src/routes/creator/index.svelte`

Goal

- Changing the race selector should update the character data, see the 'Character Debug' panel for details. (See hint 1a)
- Changing the race of the character should be reflected in the parent component.

Useful Links

- Declaring props
- Bindings

Hints

`npm run hint 1a`	Show hint
`npm run hint 1b`	Show hint
`npm run hint 1c`	Show hint
`npm run hint 1d`	Show hint

2 - The Mighty Button

Glorious Quests! - Hints

npm run hint 1a

```
  _____
/ Hi, I will be your guide today! \
\ You can call me Mr.Cowsy.        /
  --------------------------------
        \   ^__^
         \  (oo)_____
            (__)\       )\/\
                ||----w |
                ||     ||
```

Miles

Glorious Quests! - Solutions

Not THE solution, but A solution

Try first, but do take a peak if you feel stuck, and have no more hints left.

Miles

Let's get to Work!

Good luck, and feel free to ask questions!

tinyurl.com/miles-gypws

Start here 👉 README.md

Miles

Conclusion

Questions?

Thanks for Us!

dag.erik.lovgren@miles.no

alexander.castillo@miles.no