



Ilionx Devdays

“Modern webapps with Svelte”



Peter Kassenaar –
info@kassenaar.com



Zutphen, NL

Peter Kassenaar

- Sr. Frontend developer @Conclusion Confidential
- Specialty: *“Everything JavaScript”*
- JavaScript, ES6, Angular, NodeJS, TypeScript, jQuery, React, Vue, etc.

www.kassenaar.com

info@kassenaar.com

Twitter: [@PeterKassenaar](https://twitter.com/@PeterKassenaar)



CONCLUSION
CONFIDENTIAL

Expertises Cases Thema's Markten Producten Over ons Nieuws Werken bij

YOUR TRUSTED DATA PARTNER

Conclusion Confidential draagt bij aan een veiligere en gezondere maatschappij middels dataregistratie in betrouwbare dossierformingsapplicaties

Onze applicaties → Neem contact op →



Conclusion.nl Contact NL EN

<https://www.conclusion.nl/confidential>

github.com/PeterKassenaar/illionx-svelte

The screenshot shows a GitHub repository page. At the top, there's a navigation bar with links for Pull requests, Issues, Codespaces, Marketplace, and Explore. Below that is a header for the repository "PeterKassenaar / ilionx-svelte" (Public). To the right of the header are buttons for Pin, Unwatch (with 1 follower), and Fork (with 0 forks). A sidebar on the left has tabs for OctoTree, Code (selected), Issues, Pull requests, Actions, Projects, Wiki, Security, Insights, and Settings. The main content area shows a commit from "PeterKassenaar Initial commit" (commit e66d546) made "now". It lists files: .gitignore, LICENSE, and README.md, all with "Initial commit" status and "now" timestamp. Below this is a large "README.md" section containing the text: "illionx-svelte" and "Slides and demo code on the presentation Svelt, Ilionx DevDays, May 2023". To the right of the main content are sections for "About", "Releases", and "Packages", each with their respective details.

PeterKassenaar / **illionx-svelte** Public

Code Issues Pull requests Actions Projects Wiki Security Insights Settings

main 1 branch 0 tags Go to file Add file Code

PeterKassenaar Initial commit e66d546 now 1 commit

.gitignore Initial commit now

LICENSE Initial commit now

README.md Initial commit now

README.md

illionx-svelte

Slides and demo code on the presentation Svelt, Ilionx DevDays, May 2023

About

Slides and demo code on the presentation Svelt, Ilionx DevDays, May 2023

Readme 0 stars 1 watching 0 forks

Releases

No releases published Create a new release

Packages

No packages published Publish your first package

© 2023 GitHub, Inc. Terms Privacy Security Status Docs Contact GitHub Pricing API Training Blog About

Agenda



- **Introduction** – what is Svelte, comparison
- **Svelte tooling** – installation
- The **structure and architecture** of basic Svelte apps.
- Svelte building blocks/concepts

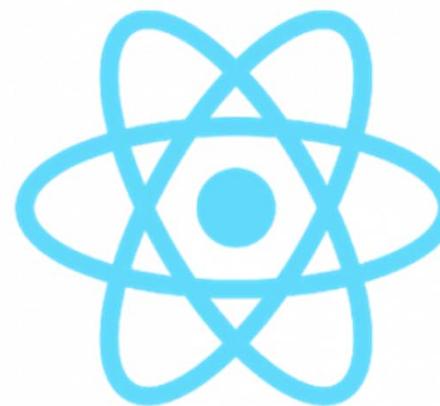
- *Live coding demo* - creating **non-trivial** applications (time permitting)



The Frontend Landscape

An overview of the frontend landscape

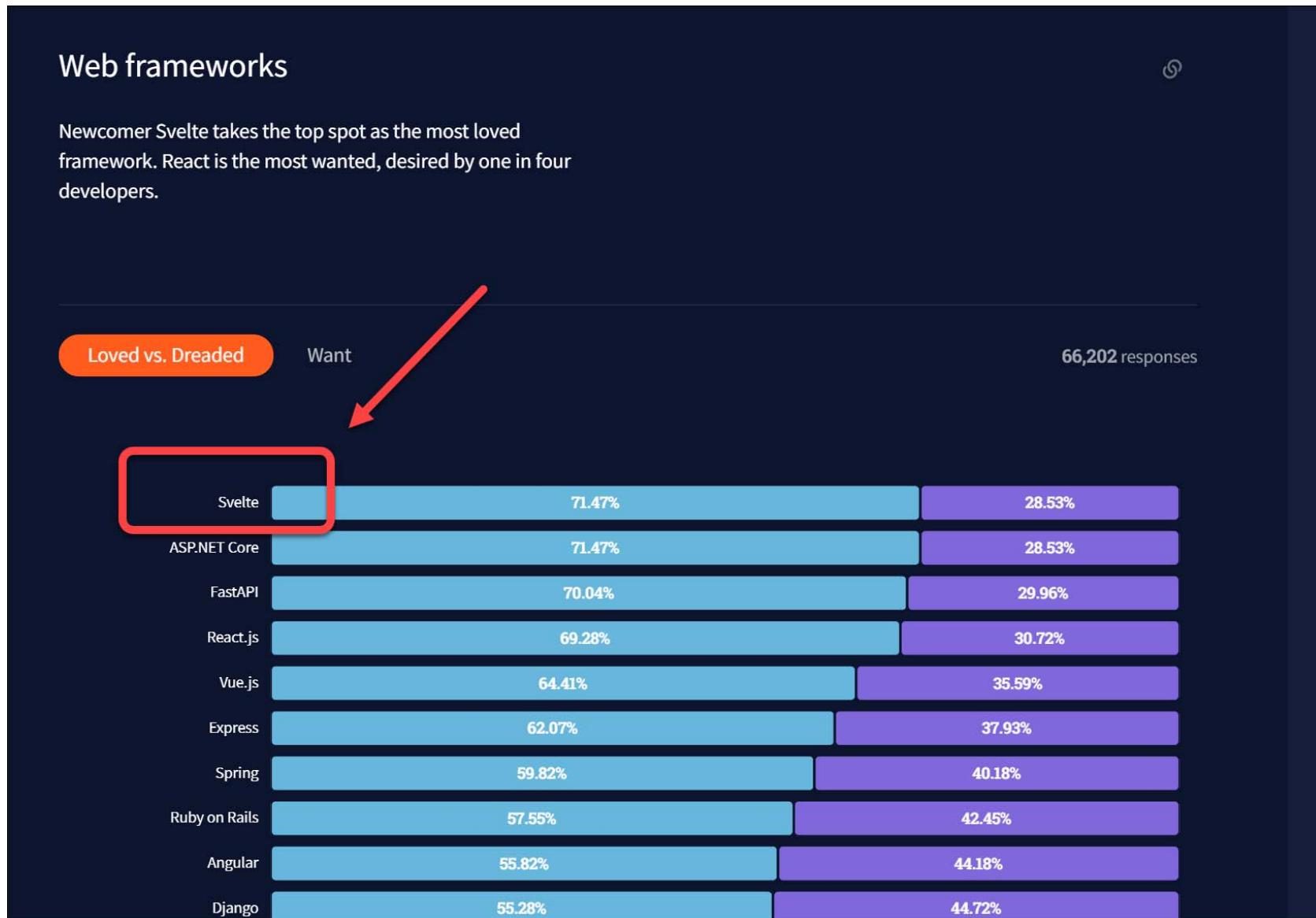
Front-end Frameworks – the big three



React



Svelte - “Most loved framework”



<https://insights.stackoverflow.com/survey/2021#section-most-loved-dreaded-and-wanted-web-frameworks>

Why Svelte, What is it?



svelte.dev

Svelte – Created by Rich Harris

The screenshot shows the official Svelte website. At the top, there's a navigation bar with links to Tutorial, Docs, Examples, REPL, Blog, and FAQ. To the right of the navigation is a "SvelteKit" button with a dropdown menu icon. The main header features a large, stylized "SVELTE" logo where the "S" is white and part of a turntable, with the word "SVELTE" written vertically on the turntable's base. Below the logo, the text "CYBERNETICALLY ENHANCED WEB APPS" is displayed. To the right of the logo is a 3D rendering of a turntable with a winding track leading away from it, with small 3D models of a car and a truck on the track. The website has three main callout boxes: an orange one on the left titled "Write less code" with the subtext "Build boilerplate-free components using languages you already know – HTML, CSS and JavaScript" and a "learn more" button; a blue one in the middle titled "No virtual DOM" with the subtext "Svelte compiles your code to tiny, framework-less vanilla JS – your app starts fast and stays fast" and a "learn more" button; and a dark grey one on the right titled "Truly reactive" with the subtext "No more complex state management". At the bottom, there's a red banner announcing "Announcing SvelteHack Our first hackathon with c..." and a video player showing a smiling Rich Harris wearing headphones, with the text "devtools FM #15" above him.

SVELTE
CYBERNETICALLY ENHANCED
WEB APPS

Write less code

Build boilerplate-free components using languages you already know – HTML, CSS and JavaScript

learn more →

No virtual DOM

Svelte compiles your code to tiny, framework-less vanilla JS – your app starts fast and stays fast

learn more →

Truly reactive

No more complex state management

devtools FM #15

Rich Harris

Svelte, Rollup

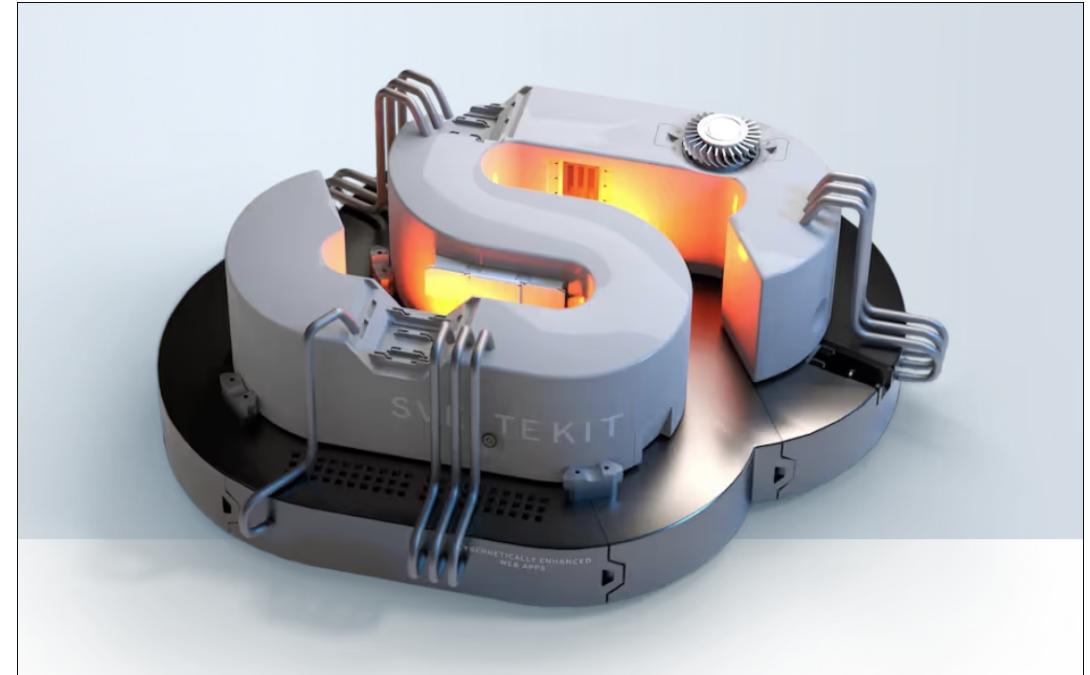
Announcing SvelteHack Our first hackathon with c...

Svelte? Actually 2 main parts



Svelte

The framework, boilerplate, syntax, compiler, runtime, etc.



Svelte KIT

Application framework,
comparable with *Next*, *NestJS*
(React) and *Nuxt* (Vue)

Brief history of Svelte



2016 – Svelte 1 – JavaScript

2018 – Svelte 2 – also JavaScript, better reactivity

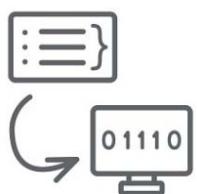
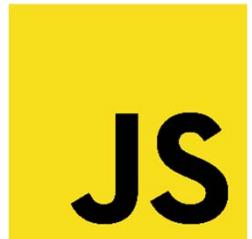
2019 – Svelte 3 – written in TypeScript

2021 – SvelteKit – Application Framework on top of Svelte

What is Svelte



- JavaScript/TypeScript tool for building UI
`<Components />`
- Svelte is a **Compiler**.
 - No framework is send to the browser
 - .svelte code is compiled to JavaScript
 - Directly executed by the browser
 - No Shadow DOM or Virtual DOM
- **High** performance, **small** footprint
- So, Svelte **runs on the server**



COMPILER

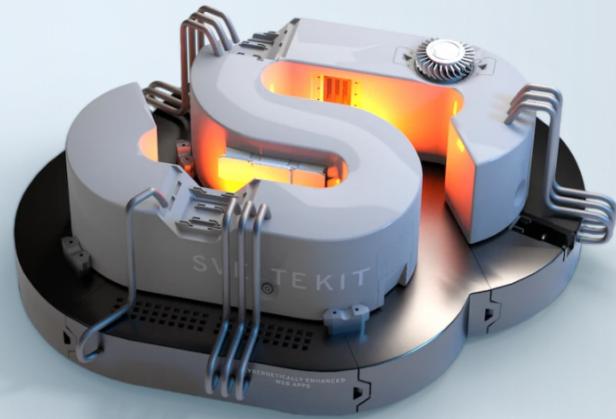


*“Instead of using techniques like virtual DOM diffing, Svelte writes code that **surgically updates the DOM** when the state of your app changes.”*

<https://svelte.io/>

SVELTE KIT

THE FASTEST WAY TO
BUILD SVELTE APPS



Powered by Svelte

SvelteKit is an application framework powered by Svelte — build bigger apps with a smaller footprint

[learn Svelte](#)

Best of both worlds

All the SEO and progressive enhancement of a server-rendered app, with the slick navigation of an SPA

[read the docs](#)

Build fast

Hit the ground running with advanced routing, server-side rendering, code-splitting, offline support and more

[read the docs](#)

kit.svelte.dev

What is SvelteKit



- Built on top of Svelte
- For creating complete Svelte applications
- SEO, SSR, Routing, Code splitting built-in
 - (like [Next.js](#) for React, [Nuxt](#) for Vue)

*“THE way to build apps or
websites with Svelte”*



Tooling and your first app

What do you need to create Svelte applications?

NodeJS 16.14+ (check your version!)



The screenshot shows the official Node.js website. At the top, there's a dark header bar with the Node.js logo and a navigation menu with links for HOME, ABOUT, DOWNLOADS, DOCS, GET INVOLVED, SECURITY, CERTIFICATION, and NEWS. Below the header, there are two circular icons: one with a crescent moon and another with a gear and an 'A'. The main content area features a large text block stating "Node.js® is an open-source, cross-platform JavaScript runtime environment." Below this, there's a section titled "Download for Windows (x64)" with two prominent green buttons: "18.16.0 LTS" (labeled "Recommended For Most Users") and "20.1.0 Current" (labeled "Latest Features"). Underneath these buttons are links for "Other Downloads", "Changelog", and "API Docs" for both versions. A note below the download sections says, "For information about supported releases, see the [release schedule](#)." At the bottom of the page, there's a footer with copyright information and a disclaimer about trademarks.

Node.js® is an open-source, cross-platform JavaScript runtime environment.

Download for Windows (x64)

18.16.0 LTS
Recommended For Most Users

20.1.0 Current
Latest Features

[Other Downloads](#) | [Changelog](#) | [API Docs](#) [Other Downloads](#) | [Changelog](#) | [API Docs](#)

For information about supported releases, see the [release schedule](#).

Copyright OpenJS Foundation and Node.js contributors. All rights reserved. The OpenJS Foundation has registered trademarks and uses trademarks. For a list of trademarks of the OpenJS Foundation, please see our [Trademark Policy](#) and [Trademark List](#). Trademarks and logos not indicated on the list of OpenJS Foundation trademarks are trademarks™ or registered® trademarks of their respective holders. Use of them does not imply any affiliation with or endorsement by them.

Again, two ways of building an app



1. `npm init vite` – select svelte option

- build a simple, frontend only site/app
- `npm run build` will generate HTML, CSS, JS-files in `/dist` directory
- Mostly just like Angular, React, Vue

2. `npm create svelte@latest my-svelte-app`

- Build a Svelte KIT app
- Including *development server, routing, SSR* and more

<https://svelte.dev/docs#getting-started>



1. Simple Svelte app

A screenshot of a terminal window titled 'npm'. The command 'npm init vite' is run, followed by 'npx: installed 1 in 2.03s'. A red box highlights the output of the command. Below, a question 'Project name:' is followed by a placeholder '... simple-svelte-app'. A question mark indicates to 'Select a framework: » - Use arrow-keys. Return to submit.' A red arrow points from the bottom left towards the 'Svelte' option in the list of frameworks.

```
PS C:\svelte> npm init vite
npx: installed 1 in 2.03s
Project name: ... simple-svelte-app
? Select a framework: » - Use arrow-keys. Return to submit.
  Vanilla
  Vue
  React
  Preact
  Lit
>  Svelte
  Others
```

Modern option: pick TypeScript



```
PS C:\svelte> npm init vite
npx: installed 1 in 2.03s
⚡ Project name: ... simple-svelte-app
⚡ Select a framework: » Svelte
? Select a variant: » - Use arrow-keys. Return to submit.
>  TypeScript
    JavaScript
    SvelteKit ↵
```

A large red arrow points from the bottom left towards the 'TypeScript' option in the terminal window, highlighting it as the selected choice.

cd into directory and run app



```
Scaffolding project in C:\svelte\simple-svelte-app...
```

```
Done. Now run:
```

```
cd simple-svelte-app  
npm install  
npm run dev
```

```
PS C:\svelte> |
```

npm install && npm run dev



The screenshot shows a Visual Studio Code (VS Code) interface with the following details:

- Project Explorer:** On the left, a tree view shows the project structure: `simple-svelte-app` (containing `C:\svelte\simple-svelte-app`, `.vscode`, `node_modules` (highlighted with a red box), `public`, `src` (containing `assets`, `lib`, `app.css`, `App.svelte` (selected and highlighted with a blue bar), `main.ts`, `vite-env.d.ts`, `.gitignore`, `index.html`, `package.json`, `package-lock.json`, `README.md`, `svelte.config.js`, `tsconfig.json`, `tsconfig.node.json`, and `vite.config.ts`), `External Libraries`, and `Scratches and Consoles`).
- Code Editor:** The main area displays the `App.svelte` file content:

```
<script lang="ts">
  import ...
</script>

<main>
  <div>
    <a href="https://vitejs.dev" target="_blank" rel="noreferrer">
      <img src={viteLogo} class="logo" alt="Vite Logo" />
    </a>
    <a href="https://svelte.dev" target="_blank" rel="noreferrer">
      <img src={svelteLogo} class="logo svelte" alt="Svelte Logo" />
    </a>
  </div>
  <h1>Vite + Svelte</h1>

  <div class="card">
    <Counter />
  </div>
</script>
```
- Terminal:** At the bottom, the terminal shows the output of the npm commands:

```
npm WARN notsup SKIPPING OPTIONAL DEPENDENCY: Unsupported platform for @esbuild/win32-arm64@0.17.18: wanted {"os":"win32","arch":"arm64"} (current: {"os":"win32","arch":"x64"})
npm WARN optional SKIPPING OPTIONAL DEPENDENCY: @esbuild/win32-ia32@0.17.18 (node_modules\esbuild\node_modules\@esbuild\win32-ia32):
npm WARN notsup SKIPPING OPTIONAL DEPENDENCY: Unsupported platform for @esbuild/win32-ia32@0.17.18: wanted {"os":"win32","arch":"ia32"} (current: {"os":"win32","arch":"x64"})

added 84 packages from 91 contributors and audited 107 packages in 7.7s

9 packages are looking for funding
  run `npm fund` for details

found 0 vulnerabilities
```
- Bottom Status Bar:** Shows "Install dependencies: From package.json // Run 'npm install' // Don't ask again (moments ago)" and "Indexing dependencies".

Open localhost <http://127.0.0.1:5173/>

A screenshot of a web browser window. The title bar says "Vite + Svelte + TS". The address bar shows "127.0.0.1:5173". The page content features the Vite logo (a blue and yellow lightning bolt) and the Svelte logo (a red 'S' in a circle). Below the logos, the text "Vite + Svelte" is displayed in a large, bold, dark font. Underneath this, there is a button with the text "count is 0". At the bottom of the page, there is a link to "SvelteKit" and a call to action: "Click on the Vite and Svelte logos to learn more".

Vite + Svelte

count is 0

Check out [SvelteKit](#), the official Svelte app framework powered by Vite!

Click on the Vite and Svelte logos to learn more

2. Using Svelte KIT



The screenshot shows a terminal window with the following output:

```
PS C:\svelte> npm create svelte@latest svelte-kit-app
npm: installed 7 in 3.957s
```

create-svelte version 4.1.1

Welcome to SvelteKit!

- ↳ Which Svelte app template?
- SvelteKit demo app (A demo app showcasing some of the features of SvelteKit – play a word guessing game that works without JavaScript!)
- Skeleton project
- Library project

A red arrow points from the bottom of the left terminal window towards the "SvelteKit demo app" option in the right terminal window.

The right terminal window shows the continuation of the command and the configuration options:

```
PS C:\svelte> npm create svelte@latest svelte-kit-app
npm: installed 7 in 3.957s
```

create-svelte version 4.1.1

Welcome to SvelteKit!

- ↳ Which Svelte app template?
- SvelteKit demo app

↳ Add type checking with TypeScript?

- Yes, using JavaScript with JSDoc comments
- Yes, using TypeScript syntax
- No

cd into directory and npm install



```
Install community-maintained integrations:  
https://github.com/svelte-add/svelte-add
```

Next steps:

- 1: cd svelte-kit-app
- 2: npm install (or pnpm install, etc)
- 3: git init && git add -A && git commit -m "Initial commit" (optional)
- 4: npm run dev -- --open

To close the dev server, hit Ctrl-C

Stuck? Visit us at <https://svelte.dev/chat>

PS C:\svelte> |

```
npm run dev -- --open
```

A screenshot of a web browser displaying a SvelteKit application. The page features a large, 3D red 'WELCOME' text centered on the screen. Below it, the text 'to your new SvelteKit app' is displayed in a smaller, dark font. A note at the bottom suggests editing 'src/routes/+page.svelte'. In the center, there is a numerical input field with '0' in red, flanked by minus and plus signs. At the bottom, a link to 'kit.svelte.dev' is provided.

HOME ABOUT SVERDLE

WELCOME

to your new
SvelteKit app

try editing `src/routes/+page.svelte`

- 0 +

visit kit.svelte.dev to learn SvelteKit

More complex application – study the files



The screenshot shows a code editor interface with the following details:

- Project Structure:** A red box highlights the left sidebar showing the project structure: `svelte-kit-app` (library root), `src`, `routes`, `Header.svelte`, `lib`, `static`, and various configuration files like `package.json`, `tsconfig.json`, and `vite.config.js`.
- Current File:** The file `Header.svelte` is open in the main editor area.
- Code Content:** The code defines a `<script>` block with imports for `$app/stores`, `$lib/images/svelte-logo.svg`, and `$lib/images/github.svg`. It then defines a `<header>` component with a link to `https://kit.svelte.dev` and an SVG icon. Below that is a `<nav>` component containing an ``.
- Terminal:** The terminal shows the command `> svelte-kit-app@0.0.1 dev` and `> vite dev --open`, followed by Vite startup logs and a message about forced re-optimization of dependencies.
- Notifications:** A small notification icon in the bottom right corner indicates there are notifications.



...looking at pure **Svelte framework**

App structure, file structure, directives

Overall File structure



The screenshot shows the VS Code interface with the following details:

- Project Explorer:** Shows the project structure with files like `simple-svelte-app`, `.vscode`, `node_modules`, `public`, `src` (containing `assets`, `lib`, and `App.svelte`), `app.css`, `main.ts`, `Vite-env.d.ts`, `.gitignore`, `index.html`, `package.json`, `package-lock.json`, `README.md`, `svelte.config.js`, `tsconfig.json`, `tsconfig.node.json`, and `vite.config.ts`.
- Code Editor:** The `App.svelte` file is open, highlighted with a red rounded rectangle. It contains Svelte code:

```
<script lang="ts">
  import svelteLogo from './assets/svelte.svg'
  import viteLogo from '/vite.svg'
  import Counter from './lib/Counter.svelte'
</script>

<main>
  <a href="https://vitejs.dev" target="_blank" rel="noreferrer">
    <img src={viteLogo} class="logo" alt="Vite Logo" />
  </a>
  <a href="https://svelte.dev" target="_blank" rel="noreferrer">
    <img src={svelteLogo} class="logo svelte" alt="Svelte Logo" />
  </a>
</div>
<h1>Vite + Svelte</h1>

<div class="card">
  <Counter />
</div>

<p>
  Check out <a href="https://github.com/sveltejs/kit#readme" target="_blank" rel="noreferrer">
</p>
```
- Terminal:** Shows the local host URL: `Local: http://localhost:5173/`.
- Bottom Status Bar:** Shows the file is 11:9 LF, UTF-8, 2 spaces*, and 0/N/A.

src/App.svelte



Components

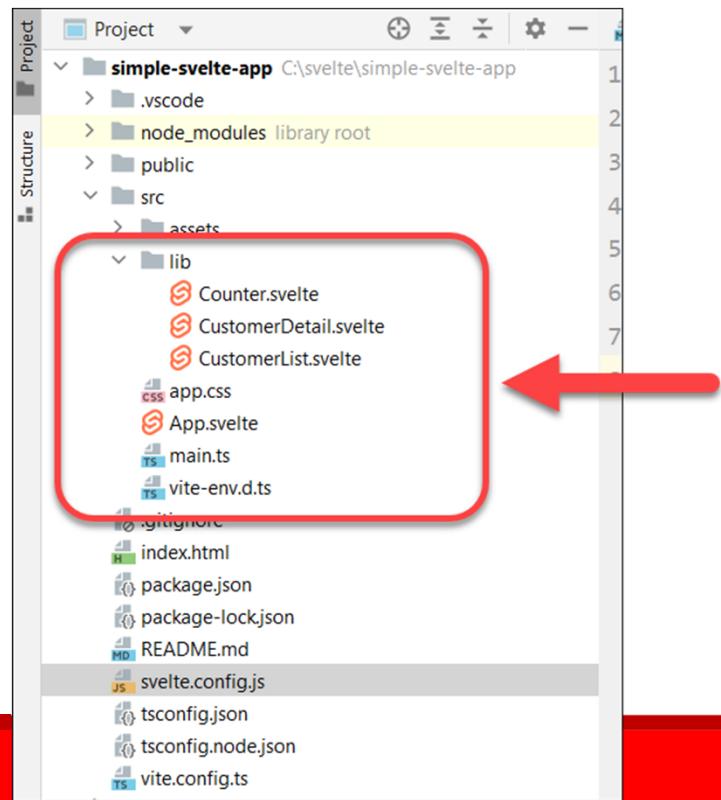
and architecture

What does your app look like, how do you work on it?

Svelte Components



- Components have the `.svelte` extension
- Are compiled and bundled by `vite` by default
 - (you can configure `Rollup` or `WebPack` if you want to)



Anatomy of a component



```
<script>
    // ALL JavaScript and/or TypeScript here.
    // This acts as the 'controller' for the component.
    // Language attributes are optional.
</script>

<style lang="scss">
    /* all CSS or SASS, LESS, SCSS here*/
</style>

<!--ALL other HTML here-->
<h2>Counter Component</h2>
<button>Increment</button>
...
...
```

Creating variables in your components



```
<script>
  // Local variable
  let name='Peter';

  // exported variable, i.e. a 'prop' for this component
  export let count = 0;

</script>
```

```
<script>
  // using the variable and importing other components
  import Counter from "./components/Counter.svelte";
  export let name = 'Peter';
</script>

<main>
  <h1>Hello {name}!</h1>
  <Counter />
</main>
```

Componenten
importeren

Creating functions to update state



- Just add **functions** in your script to update variables
 - or use syntax like `const=()=> { ... }`

```
<script>
    // variables...
    export let count = 0;

    // increment counter
    const increment = () =>{
        count++;
    }

    // decrement counter
    const decrement = () =>{
        count--;
    }

    // reset counter
    const reset = () =>{
        count = 0;
    }
</script>
```

Reactivity



- Change state by **listening to events** in the DOM with the `on:<eventName>` syntax
 - `on:click={functionName}`
- Component will **rerender** to show updated state

```
<h1>Counter Component</h1>
<h2>{count}</h2>
<button on:click={increment}>Increment</button>
<button on:click={decrement}>Decrement</button>
<button on:click={reset}>Reset</button>
```

Result



Increment

decrement

reset

count is 3

Check out [SvelteKit](#), the official Svelte app framework powered by Vite!

Click on the Vite and Svelte logos to learn more

Debugging – use Chrome DevTools as normal



The screenshot shows a browser window for "Vite + Svelte + TS" at "localhost:5173". The page displays "Hello Peter!" and "Vite + Svelte" with three buttons: "Increment", "decrement", and "reset". Below the buttons, the text "count is 0" is shown. A red arrow points from the "Increment" button on the page to the line of code "count +=1" in the DevTools Sources tab. The DevTools interface includes the following elements:

- Sources Tab:** Shows the file "Counter.svelte" with the following code:

```
<script lang="ts">
let count: number = 0
const increment = () => {
    count +=1
}
const decrement = () =>{
    count -=1
}
const reset =() =>{
    count = 0
}
</script>
<button on:click={increment}>
    Increment
</button>
<button on:click={decrement}>decrement</button>
<button on:click={reset}>reset</button>
<h2>count is {count}</h2>
```
- Breakpoints:** A breakpoint is set on the line "count +=1".
- Watch:** The variable "count" is being watched.
- Breakpoints List:** Shows a list of breakpoints, including one for "Counter.svelte" at line 4.
- Scope:** Shows the current scope variables.
- Local:** Shows the return value and this context.
- Call Stack:** Shows the call stack for the current function.
- Breakpoint Details:** Shows details for the breakpoint at "Counter.svelte:4".

Passing arguments to a function



Use the **inline function syntax** (like in React)

```
// Update counter with an arbitrary value
const update = val => {
    count += val;
}
```

```
<button on:click={()=>update(5)}>Increment by 5</button>
```

Counter Component

20

Increment Increment by 5 Decrement Reset

Visit the [Svelte tutorial](#) to learn how to build Svelte apps.

Reading values from an input field



- Use the bind: directive
- bind can be used in multiple ways and on multiple elements

```
<!--Reading values from a text field-->
<input type="number" placeholder="value..." 
       bind:value={val}
       on:keyup={e => e.key==='Enter' && update(val)} >
```

Counter Component

130

Increment Decrement Reset

Increment by 5 65 ▾

A red arrow points to the 'Decrement' button in the component's user interface, which is a light gray rectangular button with black text.

If necessary – cast your value!



```
// update value with a random number.  
// Parse it explicitly to an integer (there ARE other ways)  
const update = val =>{  
    count= parseInt(count) +  parseInt(val);  
}
```



Bindings

As a general rule, data flow in Svelte is *top down* – a parent component can set props on a child component, and a component can set attributes on an element, but not the other way around.

Sometimes it's useful to break that rule. Take the case of the `<input>` element in this component – we *could* add an `on:input` event handler that sets the value of `name` to `event.target.value`, but it's a bit... boilerplatey. It gets even worse with other form elements, as we'll see.

Instead, we can use the `bind:value` directive:

```
<input bind:value={name}>
```

This means that not only will changes to the value of `name` update the input value, but changes to the input value will update `name`.

App.svelte

```
1 <script>
2   let name = 'world';
3 </script>
4
5 <input value={name}>
6
7 <h1>Hello {name}!</h1>
```

Result

world

Hello world!

<https://svelte.io/tutorial/text-inputs>

Template syntax



Lots of directives / template syntax available

Tags

Attributes and props

Text expressions

Comments

{#if ...}

→

{#each ...}

{#await ...}

{#key ...}

{@html ...}

{@debug ...}

Element directives

on:*eventname*

bind:*property*

bind:*group*

https://svelte.io/docs#Template_syntax

Iterating over a data set {#each}



```
<script>
  // Define array of data
  let customers = [
    {id: 1, name: 'KPN', city: 'Amsterdam'},
    {id: 2, name: 'Belastingdienst', city: 'Apeldoorn'},
    {id: 3, name: 'Ilionx', city: 'Maastricht'},
    {id: 4, name: 'Rabobank', city: 'Utrecht'}
  ]
</script>
```

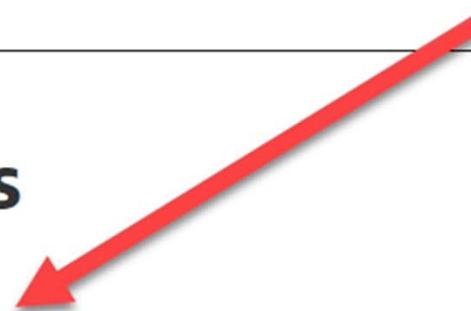
```
<h2>List of customers</h2>
<ul>
  <!-- Loop over data in array--&gt;
  {#each customers as customer}
    &lt;li&gt;{customer.id} - { customer.name }&lt;/li&gt;
  {/each}
&lt;/ul&gt;</pre>
```

Result



List of customers

- 1 - KPN
- 2 - Belastingdienst
- 3 - Ilionx
- 4 - Rabobank



Visit the [Svelte tutorial](#) to learn how to build Svelte apps

Conditional Logic – {#if}



```
<script>
    // boolean value, used in the UI
    let showCustomers = true;
</script>
<!--html-->
<h2>List of customers</h2>
{#if showCustomers}
    <ul>
        ...
    </ul>
{/if}

<button on:click={()=> showCustomers = !showCustomers}>
    Toggle customer list
</button>
```

Result



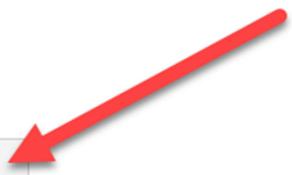
List of customers

- 1 - KPN
- 2 - Belastingdienst
- 3 - Ilionx
- 4 - Rabobank

Toggle customer list

List of customers

Toggle customer list





More Svelte Concepts

See documentation for extensive descriptions

<https://svelte.dev/docs>

Reactive assignments



- To trigger re-render, just (re-)assign local variable

2. Assignments are 'reactive'

To change component state and trigger a re-render, just assign to a locally declared variable.

Update expressions (`count += 1`) and property assignments (`obj.x = y`) have the same effect.

```
<script>
  let count = 0;

  function handleClick () {
    // calling this function will trigger an
    // update if the markup references `count`
    count = count + 1;
  }
</script>
```

Because Svelte's reactivity is based on assignments, using array methods like `.push()` and `.splice()` won't automatically trigger updates. A subsequent assignment is required to trigger the update. This and more details can also be found in the [tutorial](#).



Reactive statements with \$

3. \$: marks a statement as reactive

Any top-level statement (i.e. not inside a block or a function) can be made reactive by prefixing it with the `$:` [JS label syntax](#). Reactive statements run after other script code and before the component markup is rendered, whenever the values that they depend on have changed.

```
<script>
  export let title;
  export let person;

  // this will update `document.title` whenever
  // the `title` prop changes
  $: document.title = title;

  $: {
    console.log(`multiple statements can be combined`);
    console.log(`the current title is ${title}`);
  }

  // this will update `name` when 'person' changes
  $: ({ name } = person);

  // don't do this. it will run before the previous line
  let name2 = name;
</script>
```

(Compare: Angular does this now also in Angular V16, with `signal()` functions)

Built-in state management



- Store
- Observable (`import { writable } from 'svelte/store'`), `$subscribe`

4. Prefix stores with `$` to access their values

A *store* is an object that allows reactive access to a value via a simple *store contract*. The `svelte/store` module contains minimal store implementations which fulfil this contract.

Any time you have a reference to a store, you can access its value inside a component by prefixing it with the `$` character. This causes Svelte to declare the prefixed variable, subscribe to the store at component initialization and unsubscribe when appropriate.

Assignments to `$`-prefixed variables require that the variable be a writable store, and will result in a call to the store's `.set` method.

Note that the store must be declared at the top level of the component – not inside an `if` block or a function, for example.

Extensive Template Syntax



<style>

- TEMPLATE SYNTAX
- Tags
- Attributes and props
- Text expressions
- Comments
- {#if ...}
- {#each ...}
- {#await ...}
- {#key ...}
- Element directives
 - on:eventname
 - bind:property
 - bind:group
 - bind:this
 - class:name
 - style:property
 - use:action
 - transition:fn
 - in:fn/out:fn
 - animate:fn

{#if ...}

```
{#if expression}...{/if}
```

```
{#if expression}...{:else if expression}...{/if}
```

```
{#if expression}...{:else}...{/if}
```

Content that is conditionally rendered can be wrapped in an if block.

```
{#if answer === 42}
  <p>what was the question?</p>
{/if}
```

Additional conditions can be added with `:else if expression`, optionally ending in an `:else` clause.

```
{#if porridge.temperature > 100}
  <p>too hot!</p>
{:else if 80 > porridge.temperature}
  <p>too cold!</p>
{:else}
  <p>just right!</p>
{/if}
```

(Blocks don't have to wrap elements, they can also wrap text within elements!)

<https://svelte.dev/docs#template-syntax-comments>

Lifecycle events



```
  <script>
    <svelte:fragment>
```

RUN TIME

- svelte
- onMount
- beforeUpdate
- afterUpdate
- onDestroy
- tick
- setContext
- getContext
- hasContext
- getAllContexts
- createEventDispatcher
- svelte/store
- writable
- readable
- derived

svelte

The `svelte` package exposes **lifecycle functions** and the **context API**.

onMount

```
onMount(callback: () => void)
```

```
onMount(callback: () => () => void)
```

The `onMount` function schedules a callback to run as soon as the component has been mounted to the DOM. It must be called during the component's initialisation (but doesn't need to live *inside* the component; it can be called from an external module).

`onMount` does not run inside a **server-side component**.

```
<script>
```

<https://svelte.dev/docs#run-time-svelte>

So, **PRO's** of using Svelte



- Small bundle sizes
- Fast(er) performance
- (very) good documentation and tutorials
- Built-in State Management
- Integration of TypeScript, Sass, PostCSS
- SvelteKit as first-class citizen

CON's, downsides of using Svelte?



- Small(er) Open Source Svelte-ecosystem
 - UI libraries, Testing support, Stack Overflow, etc.
- More difficult to find experienced programmers for your company
- Early stages – a lot of `build your own`
- You are on the vanguard / you might not like that
- Forwarding events to child components not fluent
- https://www.reddit.com/r/sveltejs/comments/oxrv1r/svelte_pros_and_cons/

More info on Svelte on the interwebzzz



Discover the latest web development insights on our new [MDN Blog](#)

mdn web docs References Guides Plus Blog Theme Log in Get MDN Plus English (US)

Guides > Tools and testing > Understanding client-side JavaScript frameworks > Getting started with Svelte

▶ Introduction to client-side frameworks
▶ React
▶ Ember
▶ Vue
▼ Svelte
Getting started with Svelte

Starting our Svelte to-do list app
Dynamic behavior in Svelte: working with variables and props
Componentizing our Svelte app
Advanced Svelte: Reactivity, lifecycle, accessibility
Working with Svelte stores
TypeScript support in Svelte
Deployment and next steps
▶ Angular
▶ Git and GitHub
▶ Cross browser testing

Getting started with Svelte

[Previous](#) [Overview: Client-side JavaScript frameworks](#) [Next](#)

In this article we'll provide a quick introduction to the [Svelte framework](#). We will see how Svelte works and what sets it apart from the rest of the frameworks and tools we've seen so far. Then we will learn how to set up our development environment, create a sample app, understand the structure of the project, and see how to run it locally and build it for production.

| | |
|-----------------------|--|
| Prerequisites: | At minimum, it is recommended that you are familiar with the core HTML , CSS , and JavaScript languages, and have knowledge of the terminal/command line . |
| Objective: | To setup a local Svelte development environment, create and build a starter app, and understand the basics of how it works. |

Svelte: A new approach to building rich user interfaces

https://developer.mozilla.org/en-US/docs/Learn/Tools_and_testing/Client-side_JavaScript_frameworks/Svelte_getting_started

In this article

- Svelte: A new approach to building rich user interfaces
- Use cases
- How does Svelte work?
- First steps with Svelte
- Having a look at our first Svelte component
- Making a couple of changes
- Inspecting main.js: the entry point of our app
- A look under the hood
- Following this tutorial
- The code so far
- Summary



daily.dev/blog APPS DOCS DAILY.DEV Get daily.dev for free

In this article

Why Svelte?
What's the benefit?
What's the downside?

Resources to get you started
Svelte ecosystem 101 🤝
Tutorials 🎓
Use Cases and Demos 💡
Useful code repositories 🏛
Developer Tools 🚧
Stay updated about Svelte news 📢
All-in-one coding news reader ↗

Resources to get you started

Svelte ecosystem 101 🤝

- [Svelte's Homepage](#)
- [The official Svelte community](#)
- [Svelte GitHub repository](#)
- [Svelte Discord server](#)
- [Svelte sub-reddit](#)

Tutorials 🎓

We don't want to overwhelm you with tutorials. There are tons of tutorials out there and we picked only the ones that are both **practical, comprehensive and user-friendly**:

- [The official tutorial](#) by Svelte developers.
- [Building My First Svelte App: Thoughts and Impressions](#) by [Chris on Code](#)

Use Cases and Demos 💡

Like anything, inspiration is important when starting with something new. Check out some cool

<https://daily.dev/blog/building-with-svelte-all-you-need-to-know-before-you-start>

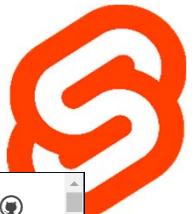
Chrome extension, Svelte DevTools



A screenshot of the Chrome Web Store page for the "Svelte Devtools" extension. The page shows the extension's icon, name, rating (53 reviews), developer information (Developer Tools), and user count (10,000+ users). A prominent "Add to Chrome" button is visible. Below the main header, there are tabs for Overview, Privacy practices, Reviews, Support, and Related. The Overview tab is selected, displaying a search bar and a detailed view of the extension's code structure. On the right side, there are sections for Props and State, showing specific properties and state variables. The overall interface is clean and modern, typical of the Chrome Web Store.

<https://chrome.google.com/webstore/detail/svelte-devtools/ckolcbmkpjpmangdbmnkpjigpkddpogn>

Don't forget: Svelte documentation!



The screenshot shows the official Svelte documentation website. On the left, there's a sidebar with navigation links like 'BEFORE WE BEGIN', 'GETTING STARTED', 'COMPONENT FORMAT', and 'TEMPLATE SYNTAX'. The main content area has a large heading 'Documentation' with a red rectangular box and an arrow pointing to it. Below this, there are two sections: 'BEFORE WE BEGIN' and 'GETTING STARTED'. The 'BEFORE WE BEGIN' section contains text about API reference documentation and links to 'interactive tutorial' and 'examples'. It also encourages asking for help in the 'Discord chatroom' and mentions 'v2 docs'. The 'GETTING STARTED' section provides information on trying Svelte in an online environment using 'REPL' or 'StackBlitz', and on creating local projects with 'SvelteKit'. At the bottom, there's a footer bar with the text 'Svelte Summit The Svelte Conference for everyone →' and a close button ('X').

<https://svelte.io/docs>



Let's create an app

Creating a non-trivial app





1. Simple Svelte app

npm init vite – select svelte option

```
PS C:\svelte> npm init vite
npx: installed 1 in 2.03s
J Project name: ... simple-svelte-app
? Select a framework: » - Use arrow-keys. Return to submit.
  Vanilla
  Vue
  React
  Preact
  Lit
>  Svelte
  Others
```

2. Using Svelte KIT



```
npm create @svelte/latest svelte-kit-app
```

A screenshot of a terminal window titled 'npm'. The command 'npm create svelte@latest svelte-kit-app' is entered, followed by 'npx: installed 7 in 3.957s'. Below this, the 'create-svelte' version 4.1.1 is shown. A section titled 'Welcome to SvelteKit!' asks 'Which Svelte app template?'. It lists three options: 'SvelteKit demo app (A demo app showcasing some of the features of SvelteKit - play a word guessing game that works without JavaScript!)', 'Skeleton project', and 'Library project'. The first option is highlighted with a green dot. A red arrow points from the bottom left towards the 'SvelteKit demo app' line, and a red rectangle highlights the terminal output above the question.

```
PS C:\svelte> npm create svelte@latest svelte-kit-app
npx: installed 7 in 3.957s

create-svelte version 4.1.1

  Welcome to SvelteKit!
  └── Which Svelte app template?
      • SvelteKit demo app (A demo app showcasing some of the features of SvelteKit - play a word guessing game that works without JavaScript!)
          o Skeleton project
          o Library project
```



3. Non-trivial app

- Show UI
 - Collect form data / user information
- Talk to backend/API
- Process results
- Handle user interaction

Svelte Country picker



localhost:5173

Svelte Country Picker!

bel Search Clear

Found countries

| |
|---|
| 
Belarus
Minsk |
| 
Belgium
Brussels |
| 
Belize
Belmopan |

Details for Belgium

| |
|--|
|  |
| Belgium (native: België) |
| Capital: Brussels |
| Population: 11555997 |
| Region: Europe / Western Europe |
| Top-level internet domain: .be |

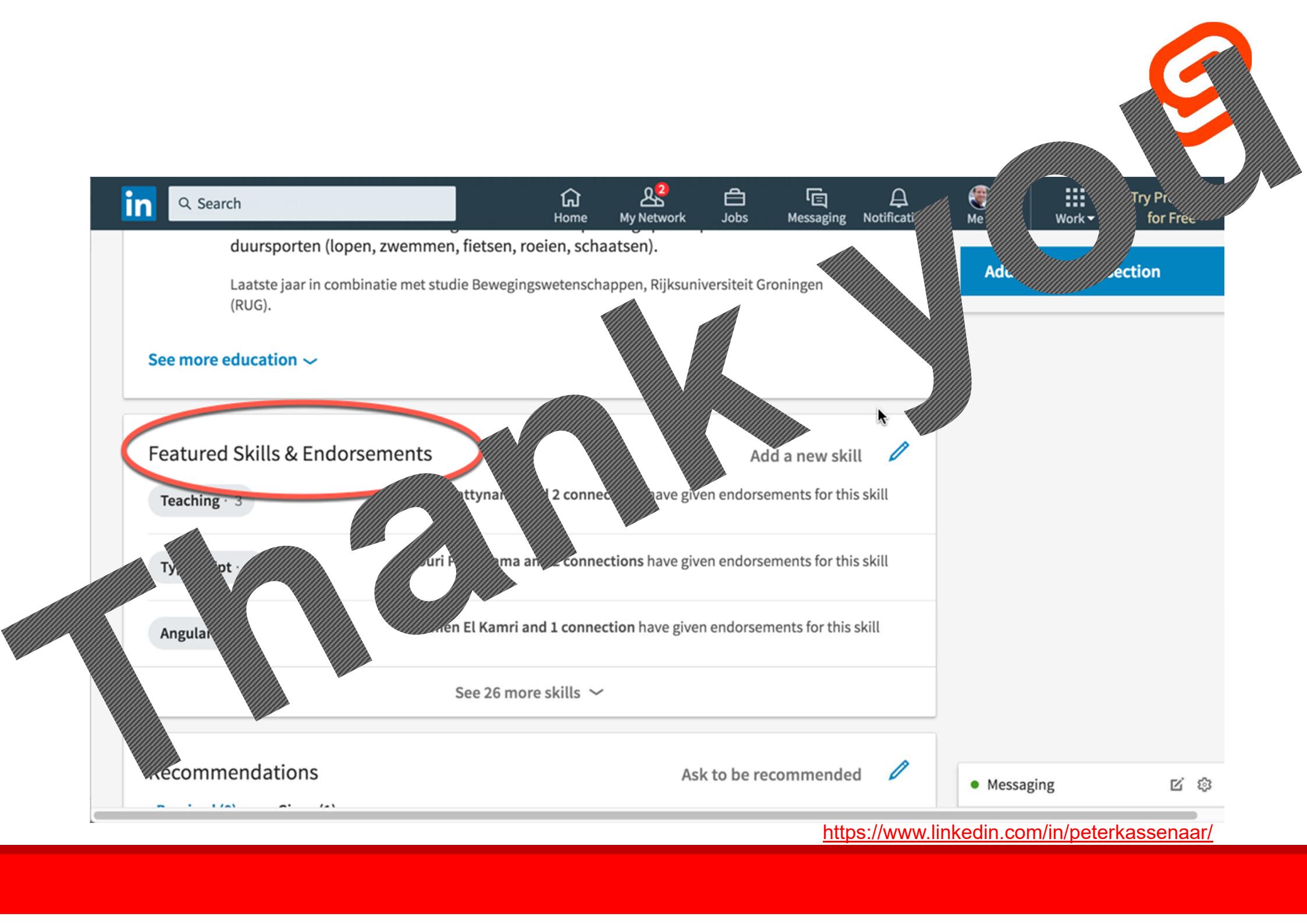
A red arrow points from the search input field to the first country result, Belarus. Another red arrow points from the first country result, Belarus, to the detailed information panel for Belgium.

Summary



- **Introduction** – what is Svelte, comparison
- **Svelte tooling** – installation
- The **structure and architecture** of basic Svelte apps.
- Svelte building blocks/concepts

- *Live coding demo* - creating **non-trivial** applications (time permitting)



<https://www.linkedin.com/in/peterkassenaar/>