Web Advanced

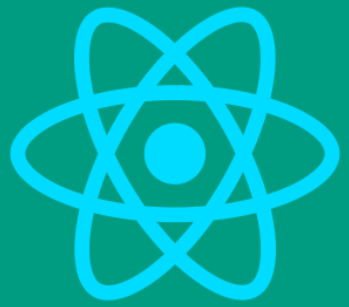
Svelte: Introduction

SAXION HOGESCHOOL

This week, Single Page Application development with Svelte

SAXION HOGESCHOOL

Modern Frontend Frameworks



- Nowadays there are many popular frontend frameworks, such as Angular, React, Vue, Ember, Svelte, …

- All of these frameworks and libraries offer you tools to build highly dynamic and modular frontend applications.

- Features that you often get are:
  - Reactive data binding **(this week)**
  - Component based architectures **(week 3)**
  - State management **(week 4)**

SAXION HOGESCHOOL

Single Page Application (SPA)

- A single page is loaded in the browser and the contents of this page are updated using JavaScript.
- Gives the user a smooth app-like experience.
- AJAX is used for loading data on the page.
- State is stored within the application (and does not need to be passed between pages).
- Can be encapsulated in mobile wrappers (Progressive Web Apps or Hybrid Mobile Applications)

Why Svelte

- Most modern frontend frameworks use similar techniques, so when you understand the concepts of one framework, it is easy to learn another framework!

- So why do you need to learn Svelte?
  - Svelte is easy to learn and not a lot of boilerplate code.
  - The performance is great since it is highly optimized when compiled.

Svelte

Svelte is a component framework

We will build various components that can be put together to form an application

Svelte defines a component as:

A reusable self-contained block of code that encapsulates HTML, CSS and JavaScript that belong together, written into a .svelte file.

So what is a component

YouTube NL

svelte

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 libraries – Svelte brings reactivity to JavaScript itself
learn more →

Svelte is a radical new approach to building user interfaces. Whereas traditional frameworks like React and Vue do the bulk of their work in the *browser*, Svelte shifts that work into a *compile step* that happens when you build your app.

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

Read the introductory blog post to learn more.

```
npx degit sveltejs/template my-svelte-project
cd my-svelte-project

npm install
npm run dev
```

See the quickstart guide for more informati

0:25 / 32:19

Svelte Crash Course

110,186 views • Jun 6, 2019

3.3K 32 SHARE SAVE ...

Traversy Media
1.64M subscribers

SUBSCRIBE

In this video we will scratch the surface with the Svelte and look at how to set it up and create a simple user interface

Tabnine
Ad www.tabnine.com
LEARN MORE

All | From your search | JavaScript | Angul

SvelteKit Crash Course - SSR, API Routes, Stores, Tailwind...
James Q Quick
30K views • 3 months ago
59:15

Get Started with Svelte and SvelteKit
Auth0
5.1K views • 2 months ago
30:05

Svelte Tutorial - Is it better than React?
freeCodeCamp.org
43K views • 2 years ago
54:53

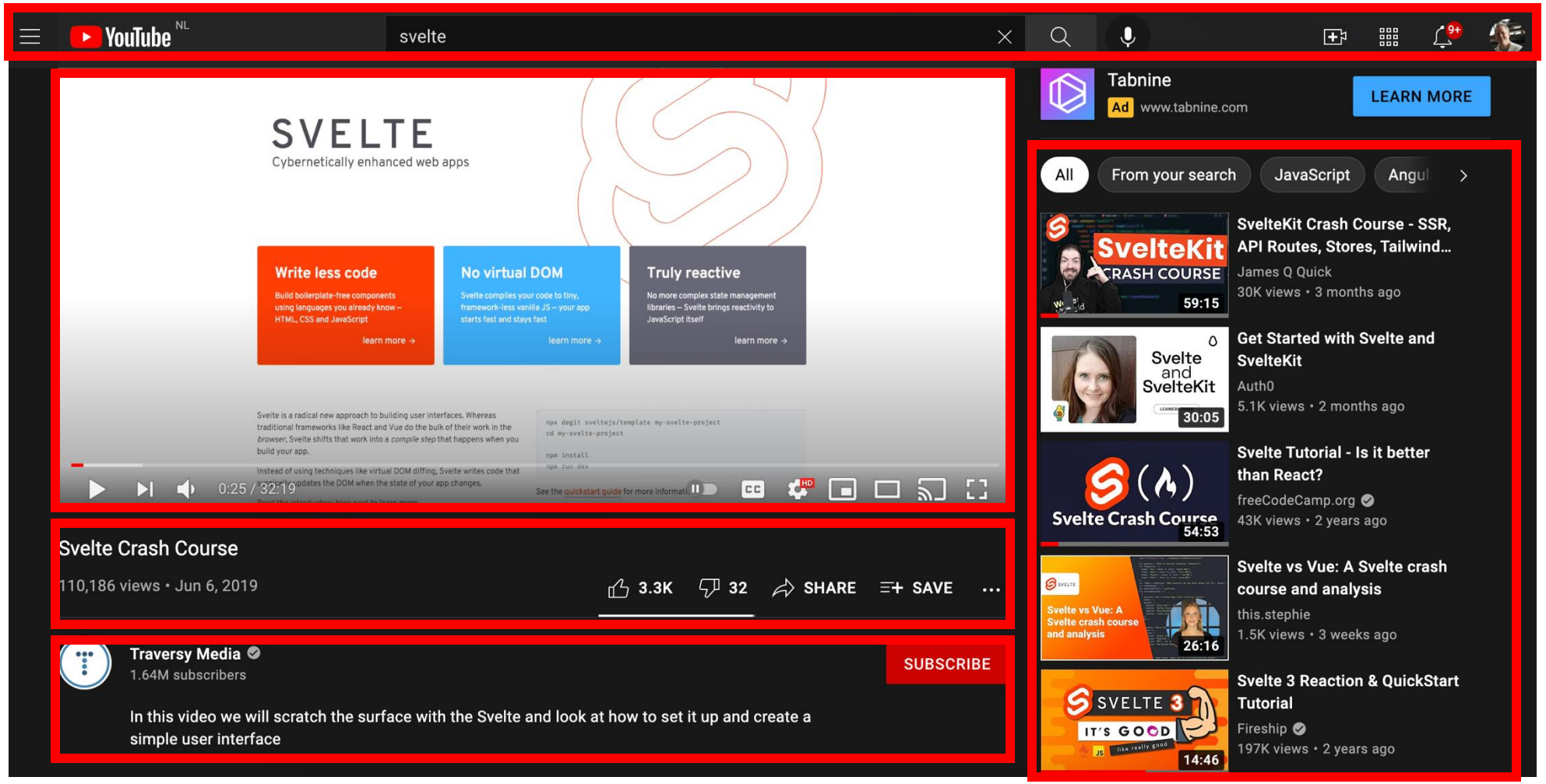
Svelte vs Vue: A Svelte crash course and analysis
this.stephie
1.5K views • 3 weeks ago
26:16

Svelte 3 Reaction & QuickStart Tutorial
Fireship
197K views • 2 years ago
14:46

SAXION HOGESCHOOL

An entire page

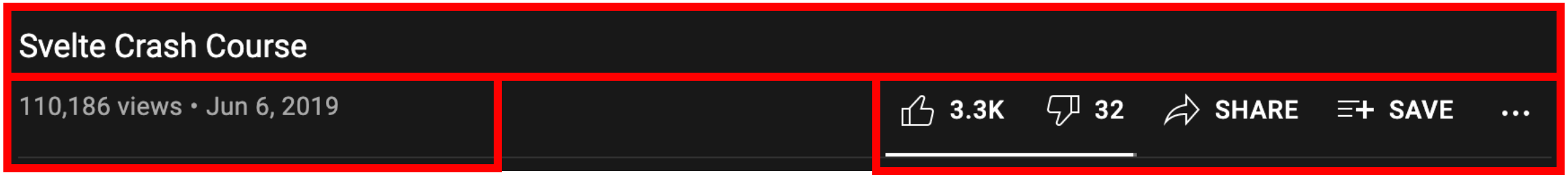
So what is a component



Blocks of content on a page

So what is a component

A title



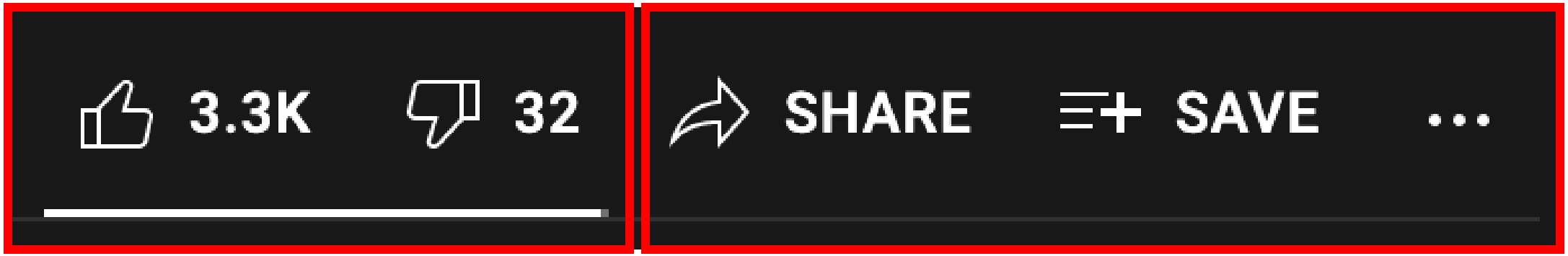
Grouped data

Button group

SAXION HOGESCHOOL

So what is a component

Like & Dislike



Actions

SAXION HOGESCHOOL

So what is a component

Button

Button

More options

Svelte

Component examples

- Login form
- Alert messages
- Links
- Popups
- Pagination

Basically anything on the page can be a component

SAXION HOGESCHOOL

Svelte

A component is a .svelte file

A component consists of 3 elements

- A `<script>` tag which contains the functionality
- Markup that defines the HTML structure as well as some event handling
- An optional `<style>` tag that defines the layout

SAXION HOGESCHOOL

Svelte

The <script> section

- Defines the behavior.
- Is as close to Javascript as possible
- You can use Javascript libraries

Svelte

HTML Structure

- Can be any HTML
- Does not need to have a single root element
- Can contain “template logic”
- Can contain event handling
- Can consist of other components

Svelte

The <style> section

- Contains plain CSS
- Any style placed there is “component scoped”

```
<style>
    h1 { color: red; }
</style>
```

- Will only affect h1 elements within the component

SAXION HOGESCHOOL

Svelte

Putting it all together

Lets make a simple (counter) component

Start off with the basics:

```
<script>
  let counter = $state(0);
</script>

Counted: {counter}
<button>+</button>
<button>-</button>
<button>0</button>
```

- A variable to keep track of the count (note the `$state()`)
- Markup that shows the count
- Some buttons to change the counter

SAXION HOGESCHOOL

Reactive Data Binding

Reactive Data Binding

- One of the most important concepts in modern frontend frameworks is reactive data binding.
- Reactive data binding establishes a link between the variables (often called the model) and the UI (the html)
- When the data changes, the framework or library automatically updates the UI elements.

```
<script>
  let counter = $state(0);
</script>

Counted: {counter}
<button>+</button>
<button>-</button>
<button>0</button>
```

The counter variable is an example of a variable that is reactive. When the value changes, the UI is updated automatically.

SAXION HOGESCHOOL

Reactive Data Binding

Important to know in Svelte!!!

- Svelte's reactivity is handled by state
- State is declared by the state rune
- `$state(initialValue);`
- Without the use of `$state()` reactivity is not garuanteed
- There are other runes to help with state management `$derived()`, `$effect()`

SAXION HOGESCHOOL

Svelte

Next: interaction

The buttons have to do something.
Lets make some event handlers.

```
<script>
  let counter = $state(0);


  const increment = () => counter++;
  const decrement = () => counter--;
  const reset = () => counter = 0;
</script>
```

Nice, but the functions don’t trigger themselves

SAXION HOGESCHOOL

Svelte

Interaction 2

We need to tell the component when to call the functions

```
Counted: {counter}
<button onclick={increment}>+</button>
<button onclick={decrement}>-</button>
<button onclick={reset}>0</button>
```

Events in Svelte are the same as in HTML and Javascript.

SAXION HOGESCHOOL

Directives

Directives

- Directives are special syntax or attributes applied to HTML elements that enable dynamic behavior, such as conditional rendering, event handling, or looping through data.

- Javascript code needs to be placed between `{ }`

- The `onevent` directive can be used to listen for DOM events and trigger a custom function.
  - For example: `<button onclick={increment}>`

- Variables can be directly bound to interactive elements (two-way binding) by using the `bind:property` directive
  - For example: `<input bind:value={name}>`

SAXION HOGESCHOOL

Directives

Template logic

- In addition, there is also template logic that can be used for conditional rendering or rendering multiple elements:

- The `{#if} / {:else if} / {:else}` are used for conditional rendering.

- The `{#each}` is used for looping over data and rendering each element.

Svelte

Display a list

Looping data can be done using `#each`

```
<ul>
  {#each list as item}
    <li>{item}</li>
  {/each}
</ul>
```

Svelte can loop anything that has a `length` property

SAXION HOGESCHOOL

Svelte

Event handler

```
<script>
  let list = $state([]);

  const addItem = () => list.push(`Item ${list.length + 1}`);
</script>

<button onclick={addItem}>Add</button>
```

Because of `$state` we can use `list.push()` to add elements to the list.

In this case we add a new string “Item” followed By the number of elements in the list.

SAXION HOGESCHOOL

Svelte

Conditionals

If there are no items, we may want to let the user know.

Lets make the list conditional

```
{#if list.length === 0}
  <p>There are no items on the list!</p>
{:else}
  <ul>
    {#each list as item}
      <li>{item}</li>
    {/each}
  </ul>
{/if}
```

SAXION HOGESCHOOL

Building a sample application

Shopping list application

- The user should be able to enter the name of a product and add it to the shopping list.
- The application should display the shopping list or mention that the shopping list is empty.
- You should be able to remove products from the shopping list.

Svelte

Recap

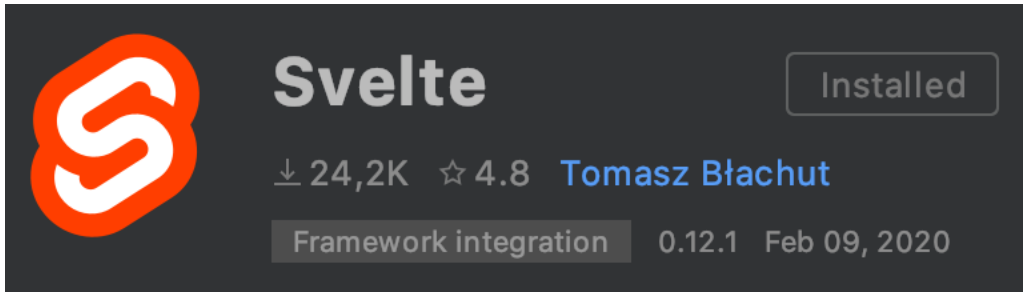
- We will be using Svelte
- Component based
- One file one component
- Component contains script, markup and style
- Can use regular Javascript (libraries)
- Compose simple components into complex structures

SAXION HOGESCHOOL

Svelte

Development environment

- You can use any text editor
- IntelliJ and VS Code are popular
- Both have plugins to make your life easier



Svelte

Resources

If you have a problem, chances are someone else had it too!

Check:

- The website
- StackOverflow
- The Svelte Discord server
- Youtube
- If all else fails…

SAXION HOGESCHOOL

Svelte

Google

Is your friend!

(or ask your teacher)

SAXION HOGESCHOOL