



# SOS APP

Projeto de Aplicações Web



Luis Mota	38186
Toni Lopes	36644



# Svelte: Introdução

- Svelte é uma ferramenta para construir aplicações web rápidas;
- Facilita o processo de criação de interfaces de utilizador interativas;
- Converte a app em tempo de compilação, ao em vez de tempo de execução.





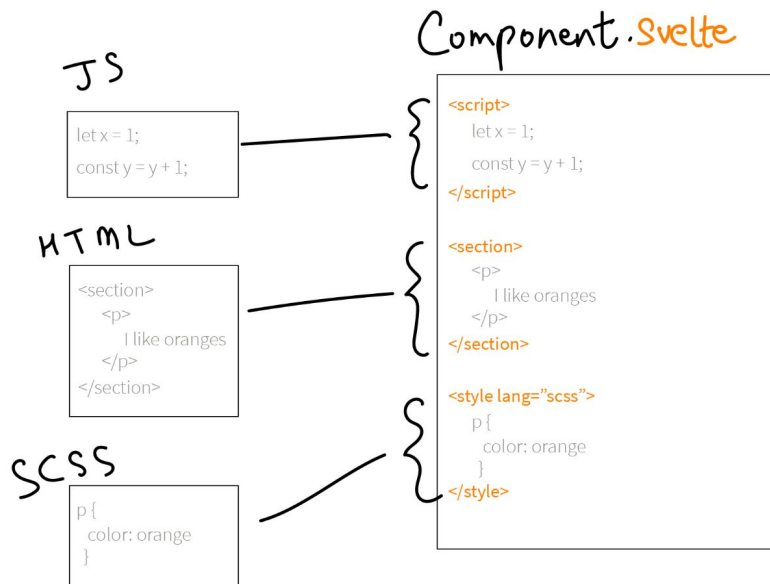
# Svelte: Estrutura de Diretórios

- Existem 4 diretórios principais:
  - Node modules;
  - Public;
  - Scripts;
  - Src;

```
C:.\n├── node_modules\n├── public\n├── scripts\n└── src
```

# Svelte: Estrutura de um Componente

- Scripts;
- Template;
- Style;



# Svelte: Exports

Exportar propriedades do componente



```
1 <script context="module">
2   export const auth = new class {
3     constructor() {
4       this.getUser = "/auth/getUser"
5       this.login = "/auth/login"
6       this.logout = "/auth/logout"
7       this.register = "/auth/register"
8       this.refresh_token = "/auth/refresh_token"
9     }
10  }
11  export const follower = new class {
12    constructor() {
13      this.getAllFollowers = "/follower/"
14      this.assocFollower = "/follower/assoc"
15      this.deassocFollower = "/follower/deassoc"
16    }
17  }
18 </script>
```

**Nota:** para exportar um componente não são necessárias estas conveniências

# Svelte: Imports

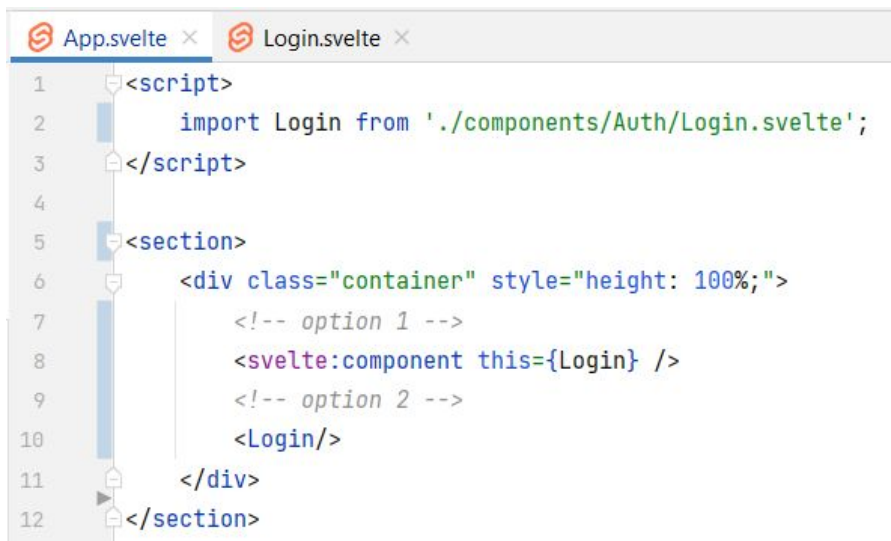


The screenshot shows a code editor with two tabs: 'Follower.svelte' and 'Routes.svelte'. The 'Follower.svelte' tab is active, displaying the following code:

```
1 <script>
2   // usual import
3   import axios from "axios";
4   // import methods/variables from svelte/js file
5   import { onMount } from "svelte";
6   import { isAuthenticated, updateStore } from "../../store/store";
7   import { auth } from "../../Routes.svelte";
8   // import component
9   import AssocFollower from "./AssocFollower.svelte";
10  import ListFollowers from "./ListFollowers.svelte";
```

# Svelte: Estrutura de Componentes

Utilizar um componente



```
1 <script>
2   import Login from './components/Auth/Login.svelte';
3 </script>
4
5 <section>
6   <div class="container" style="height: 100%;">
7     <!-- option 1 -->
8     <svelte:component this={Login} />
9     <!-- option 2 -->
10    <Login/>
11  </div>
12 </section>
```

# Svelte: Comunicação entre Componentes

Componente 'pai' à escuta de um certo evento do componente chamado

```
App.svelte x Login.svelte x
1 <script>
2   import Login from './components/Auth/Login.svelte';
3 </script>
4
5 <section>
6   <div class="container" style="height: 100%;">
7     <Login on:login_success={event} => console.log(event.detail.tab)}>
8   </div>
9 </section>
```

```
App.svelte x Login.svelte x
1 <script>
2   import { createEventDispatcher } from "svelte";
3
4   const dispatch = createEventDispatcher();
5
6   dispatch("login_success", { tab: "Home" });
7
```

Despachar um evento para o componente 'pai'



# Svelte: Obter Valor de Elemento HTML

Variável tem sempre o mesmo valor  
que o input

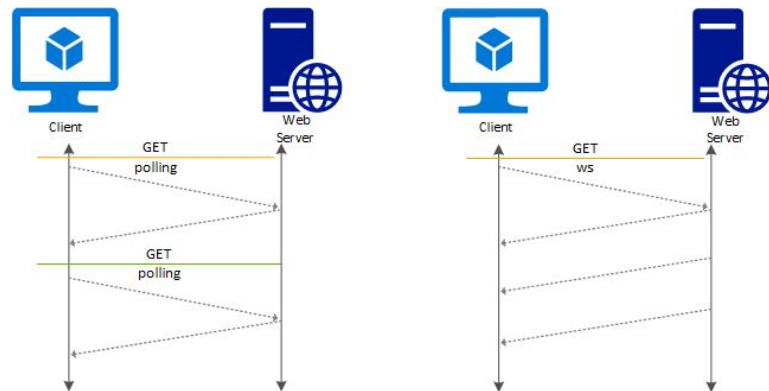
```
Login.svelte x
1  <script>
2      let username = "";
3  </script>
4
5  <div id="card">
6      <input id="username" name="username" bind:value={username} />
7  </div>
```

## Svelte: Condições/Iteradores em Template HTML

```
68 <div class="container text-center">
69   {#each followers as follower}
70     {#if follower.id !== -1}
71       <tr>
72         <td>{follower.id}</td>
73         <td>{follower.username}</td>
74         <td>
75           <button type="button" id={follower.id} on:click={() => submit(follower.id)}></button>
76         </td>
77       </tr>
78     {/if}
79   {/each}
80 </div>
```

# Websocket

- WebSocket é um protocolo de comunicação que fornece canais de comunicação full-duplex em uma única conexão TCP;
- Neste projeto o canal é aberto no Endpoint de Login e é fechado quando a storage é limpa.



## Link do Repositório

- [SOS APP](#)



# GitHub