Svelte:
O Não-Framework

Por João Pedro Vanderlei

Svelte

É um compilador de Javascript para desenvolvimento de aplicações de front-end. Permite a criação de componentes, estilização e utilização de Js dentro de um código extremamente similar a HTML puro. Seu criador desenvolvia no New York Times, e para que pudesse levar informação a todos, necessitava de sites mais leves e performáticos.

O Svelte foca muito no Developer Experience (DX), e uma das maneiras de melhorar nossa experiência é permitir adicionar no mesmo arquivo HTML, CSS e Javascript, facilitando o entendimento e a performance dos desenvolvedores. Sua sintaxe adiciona poucos caracteres especiais, diferenciando do React, que necessita chamar funções. Estes fatores contribuem para um código de menor complexidade.

3 princípios do Svelte

- Menos código, mesmo resultado
- Mais reativo a interação do usuário, de maneira mais simples.
- Sem Virtual Dom, mais rápido

Mesmo resultado, menos código

Quanto mais linhas de código seu sistema tiver, mais espaço haverá para bugs e más otimizações se esconderem. Os códigos em Svelte costumam utilizar 40% menos linhas para implementar o mesmo mecanismo.

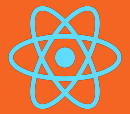
O Svelte traz a atualização de estado embutida no objeto, se desvencilhando da necessidade de ter uma função para cada estado. Exemplo:

```
let contador = 0

function soma( ) {
        contador += 1;
}
```

```
const [contador,  setContador] = useState(0)

function some( ) {
        setContador(contador + 1)
}
```

O mesmo acontece com as declarações reativas, que podem facilmente substituir o UseEffect, porém com sintaxe simplificada.

```
let contador = 0
$ : dobro = contador * 2
```



```
const [contador,  setContador] = useState(0)
const [dobro, setDobro] = useState(0)

useEffect(( ) => {
        setDobro(contador * 2)
 }, [contador])
```

Melhor desempenho sem Virtual DOM

O que é Virtual DOM

DOM: Document Object Model é uma API que nos permite acessar e modificar tags HTML dentro de sua hierarquia.

Virtual DOM: Um conceito implementado por algumas bibliotecas que representa o DOM e realiza a reconciliação com o DOM.

Reconciliação: Processo de comparação no qual o Virtual DOM é utilizado para repassar as mudanças para o DOM.

Diffing: É o processo de comparação utilizado na Reconciliação, que define quais elementos devem ser atualizados.

Diffing: O Problema do Virtual DOM

A utilização do Virtual DOM é performática em comparação a utilizar o DOM, pois as alterações no DOM são demoradas e mal otimizadas, Porém, a reconciliação ainda leva 3 passos.

Primeiro atualizar o componente no Virtual DOM, depois comparar as hierarquias e a partir disso atualizar o DOM.

A solução do Svelte

Quando o código do Svelte é compilado, ele já informa quais componentes devem ser renderizados quando ocorrem mudanças. Basicamente ele realiza apenas a 3º etapa do processo de reconciliação, o que confere muito mais desempenho.

```
if (changed.name) {
        text.data = name;
 }
```

Sintaxe

: /

- # Usado para iniciar estruturas de programação.. Exemplos: if, each, await
- : Usado para chamar demais partes de uma estrutura de programação. Exemplos: else, else if, then.
- / Usado para fechar as estruturas de programação.

#If :else if /if

O Svelte permite a interpolação de estruturas condicionais em meio ao código. Exemplo:

```
let lampada = “off”

{#if (lampada === “off”)}
        <img src:”imgLampadaApagada”>
{:else if(lampada === “on”)}
        <img src:”imgLampadaAcessa”>
{:else}
        <img src:”imgLampadaQuebrada”>
{/if}
```

#each

É o laço de repetição do Svelte. Exemplo:

```
let baralho = []// API Deck de Cartas

{#each baralho as carta }
        <div>
                <p>{carta.value, carta.suit}</p>
                <img src:{carta.svg}>
        </div>
 {/each}
```

$ Variáveis dinâmicas

```
<script>
	let cartaVirada // busca carta do deck da API
	$: zap = {
		value: proximaCarta(cartaVirada),
		suit: “clubs”
	}
</script>
```

Style

O Svelte permite a estilização ser realizada no mesmo arquivo que o código do componente, sendo sempre este seu escopo,com exceção de quando especificado pela chave “:global()” . Exemplo:

```
<style>
        div {
                color: red;
                background: white;
        }
        :global(div.spades),
        :global(div.clubs) {
                colour: black
        }
</style>
```

Props

O Svelte não exige que os componentes sejam exportados para serem utilizados, portanto as keywords “Export” e “Import” são reservadas para os props.

```
<script>
	import Carta from “./Carta.svelte“
</script>

<Carta value={x} suit={y} />
```

```
<script>
	export let value;
	export let suit;
</script>

<div>
	//Layout da Carta
</div>
```

Prática

Para testar o código, acesse: https://svelte.dev/repl/

```
<script>
	let contador = 1;
	$: dobro = contador * 2;

	function soma() {
		return contador += 1;
	}
</script>
<h1>Contador! {contador}</h1>
<h1>dobro! {dobro}</h1>
<button on:click={soma}>
	Soma e Dobra
</button>

<style>
	h1{
		color: red;
	}
	button{
		background-color: black;
		color: white;
	}
</style>
```

Referências:

https://svelte.dev/

https://svelte.dev/docs

https://svelte.dev/blog

Dúvidas?

Obrigado pela atenção!