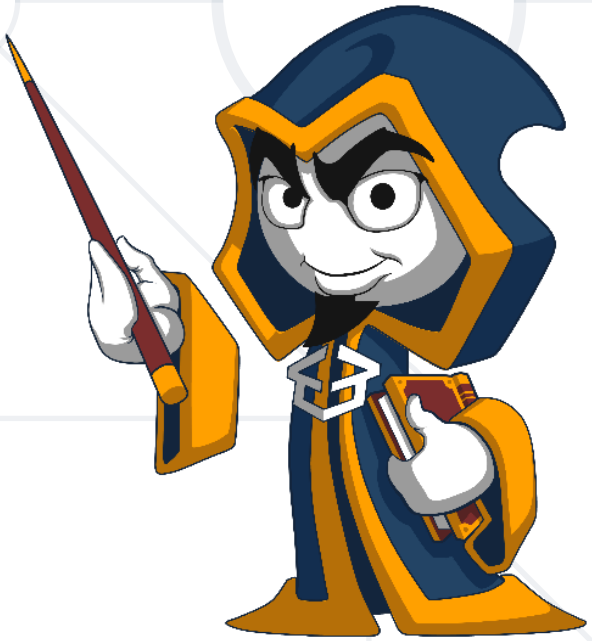
Composition API and Libraries

SoftUni Team

Technical Trainers

Software University

SoftUni



Software University

https://about.softuni.bg

Table of Contents

Have a Question?

Software University

sli.do

#vue-js

Composables

Composition vs Mixins

Composables

- Use Vue's Composition API to encapsulate and reuse stateful logic
- Similar to Vue 2 Mixin's idea
- It is a convention to name composable functions with camelCase names that start with "**use**"
- Always return a plain, non-reactive object containing multiple refs. This allows it to be destructured in components while retaining reactivity
- Composables can be extracted not only for reuse, but also for code organization

Problem – reusable fetch function

- Create a "composable" named **useFetch**
  - Should accept a string
  - Should return **data**, **isLoading** and **hasError** variables

```
<template>
  <div>
    <h1>Pokemon Data</h1>
    <ul v-if="!isLoading">
      <li v-for="pokemon in data.results"
:key="pokemon.id">
        {{ pokemon.name }}
      </li>
    </ul>
  </div>
</template>
```

```
<script setup>
import { useFetch } from "./useFetch";
const { data, isLoading, hasError } = useFetch("https://pokeapi.co/api/v2/pokemon");
</script>
```

Mixins drawbacks

- **Unclear source of properties**
  - Unclear which instance property is injected by which mixin
- **Namespace collisions**
  - Multiple mixins from different authors can potentially register the same property keys
- **Implicit cross-mixin communication**
  - Mixins that need to interact with one another have to rely on shared property keys

Vue Router

Accessing Router and Route

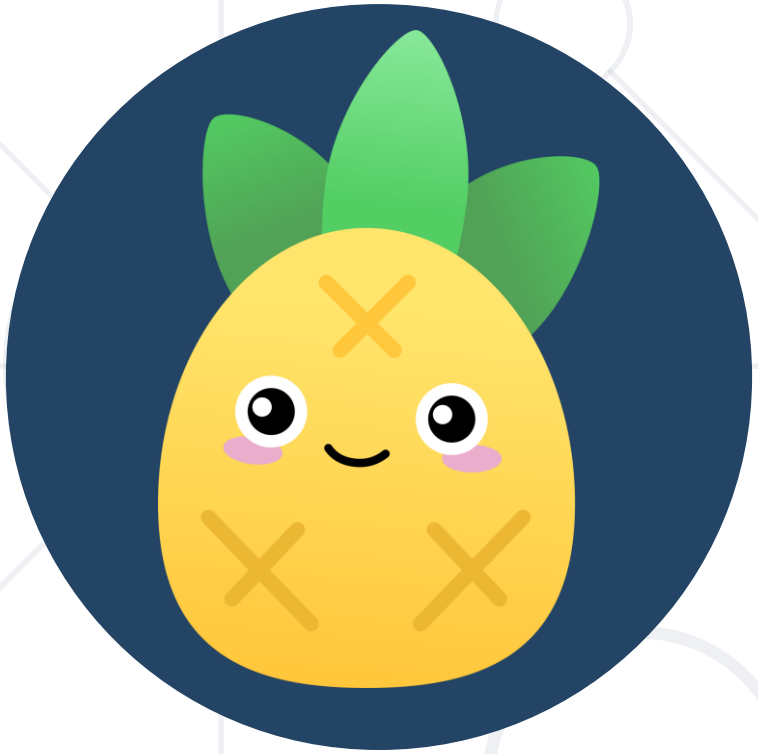
```
import { useRouter, useRoute } from 'vue-router'

const router = useRouter()
const route = useRoute()

function pushWithQuery(query) {
  router.push({
      name: 'search',
      query: {
        ...route.query,
        ...query,
      },
  })
}
```

Watching route

```
<script setup>
import { useRoute } from 'vue-router'
import {  watch } from 'vue'

const route = useRoute()

watch(
  () => route.params.id,
  newId => {
      // Do some logic based on the new id
  }
)
</script>
```

Navigation Guards

```
<script setup>
import { onBeforeRouteLeave, onBeforeRouteUpdate } from 'vue-router'

onBeforeRouteLeave((to, from) => {
  const answer = window.confirm(
    'Do you really want to leave? you have unsaved changes!'
  )
  if (!answer) return false
})

onBeforeRouteUpdate(async (to, from) => {
  if (to.params.id !== from.params.id) {
    // only fetch the user if the id changed
    userData.value = await fetchUser(to.params.id)
  }
})
</script>
```



State Management

Pinia stores

Store usage

```
<script setup>
import { useCountStore } from "./stores/counter";

const store = useCountStore();

console.log(store.doubledCount);
console.log(store.doubledCountPlusOne);
</script>

<template>
  <h2>{{ store.count }}</h2>
  <button @click="store.increment">change</button>
</template>
```

Defining a Setup Store

- Similar to setup()
- Pass in a function that defines reactive properties and methods
- Returns an object with the properties and methods we want to expose

```
export const useCounterStore = defineStore('counter', () => {
  const count = ref(0)
  const name = ref('Eduardo’)

  const doubleCount = computed(() => count.value * 2)
  function increment() {
    count.value++
  }

  return { count, name, doubleCount, increment }
})
```

Suspense and Top-level await

Pinia stores

Top-level await

- Let's say we want to fetch some pokemons

```
// pokemonAPI.js

export const fetchThePokemon = async () => {
  try {
    const response = await
fetch("https://pokeapi.co/api/v2/pokemon");
    if (!response.ok) {
      throw new Error("Network response was not
ok");
    }
    const data = await response.json();
    return data.results;
  } catch (error) {
    console.error("Error fetching data:", error);
    return [];
  }
};
```

```
// FetchingComponent.vue
<template>
  <div>
    <h1>Pokemon Data</h1>
    <ul>
      <li v-for="pokemon in pokemons"
:key="pokemon.id">
        {{ pokemon.name }}
      </li>
    </ul>
  </div>
</template>

<script setup>
import { fetchThePokemon } from "./pokemonAPI";
const pokemons = await fetchThePokemon();
</script>
```

Suspense

```
<template>
  <FetchingComponent />
</template>

<script setup>
import FetchingComponent from
"./components/FetchingComponent.vue";
</script>
```

```
[Vue warn]: Component <Anonymous>: setup function
returned a promise, but no <Suspense> boundary was
found in the parent component tree. A component with
async setup() must be nested in a <Suspense> in
order to be rendered.
  at <FetchingComponent>
  at <App>
```

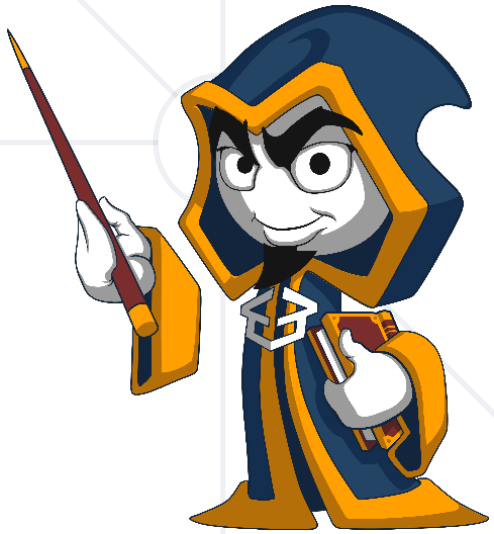
```
<template>
  <Suspense>
    <FetchingComponent />

    <template #fallback> Loading... </template>
  </Suspense>
</template>
```

Practice

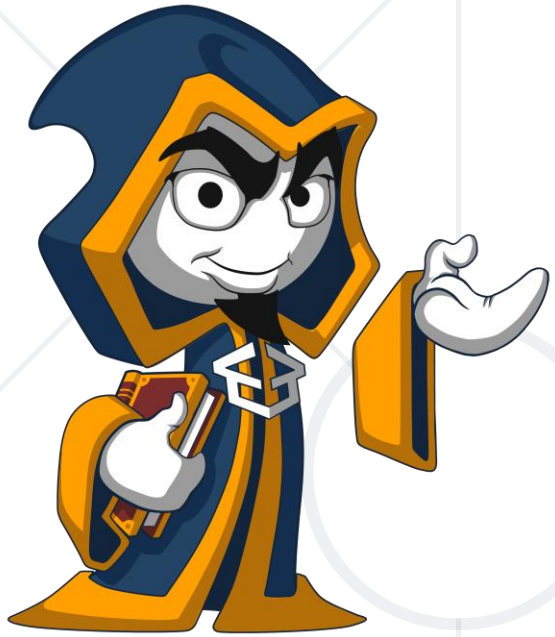
Live Exercise in Class (Lab)

Summary

- Composables are alternative to Mixins in Vue 2
- Composable function help us reuse logic
- Vue Router and Pinia both integrate seamlessly with Composition API
- Suspense helps us use top level await or load async components



Questions?



SoftUni

Software University | SoftUni Creative | SoftUni Digital | SoftUni Foundation | SoftUni Kids | Finance Academy

SoftUni Diamond Partners

Software University

SUPER HOSTING .BG

Coca-Cola HBC Bulgaria

Flutter™ International

INDEAVR
Serving the high achievers

AMBITIONED

DRAFT KINGS

SOFTWARE GROUP

BOSCH

Postbank
Решения за твоето утре

PHAR VISION

SmartIT

DXC TECHNOLOGY

createX

License

- This course (slides, examples, demos, exercises, homework, documents, videos and other assets) is **copyrighted content**
- Unauthorized copy, reproduction or use is illegal
- © SoftUni – https://about.softuni.bg/
- © Software University – https://softuni.bg

Trainings @ Software University (SoftUni)

- Software University – High-Quality Education, Profession and Job for Software Developers
  - softuni.bg, about.softuni.bg
- Software University Foundation
  - softuni.foundation
- Software University @ Facebook
  - facebook.com/SoftwareUniversity
- Software University Forums
  - forum.softuni.bg