Vue 3 at Keap

Making the world a better place by keeping up
with modern frontend frameworks /s

New to Vue 3

- First-class Typescript support

- Performance

- Composition API

- Other new features

  - Teleport

  - Fragments (multiple root elements)

Why Typescript?

- More reliable/higher confidence in code

- Typing acts as documentation

- Better tooling/intellisense

- Easier to maintain with high amount of developers

Sure, but why Typescript in Vue?

Better type safety for props

```
props: {
  broadcast: {
    type: Object,
    required: true,
    // default: () ⇒ ({}),
  },
}
```

```
props: {
  broadcast: {
    type: Object,
    required: true,
    validator: (value) ⇒ {
      ['id', 'scheduleDate'].forEach((key) ⇒ {
        broadcast.hasOwnProperty(key)
      })
    },
    // But what about the type of the values in broadcast?
  },
}
```

```
import { PropType } from 'vue'

type Broadcast = {
  id: string | number;
  scheduleDate: string;
}

props: {
  broadcast: {
    type: Object as PropType<Broadcast>,
    required: true,
  }
}
```

```
// Using <script setup> syntactic sugar
type Broadcast = {
  id: string | number;
  scheduleDate: string;
}

const { broadcast } = defineProps<{
  broadcast: Broadcast
}>()
```

Better Vue Tooling

- Vetur
- Volar

Performance

- Smaller bundle size due to treeshakeability

- Mounts quicker

- Updates quicker

- Uses less memory

- Spreadsheet with details

New features

- Teleport
  - Similar to `` `portal-vue` ``, can move elements/components through the DOM
- Fragments
  - Multiple root elements in component
- Composition API
- Many others

Fragments example

```
<template>
  <div>
    <AppHeader />
    <AppMain />
    <AppFooter />
  </div>
</template>
```

```
<template>
  <AppHeader />
  <AppMain v-bind="$attrs" />
  <AppFooter />
</template>
```

NOTE: You need to define which top level element inherits attributes

Composition API

Migration

Link