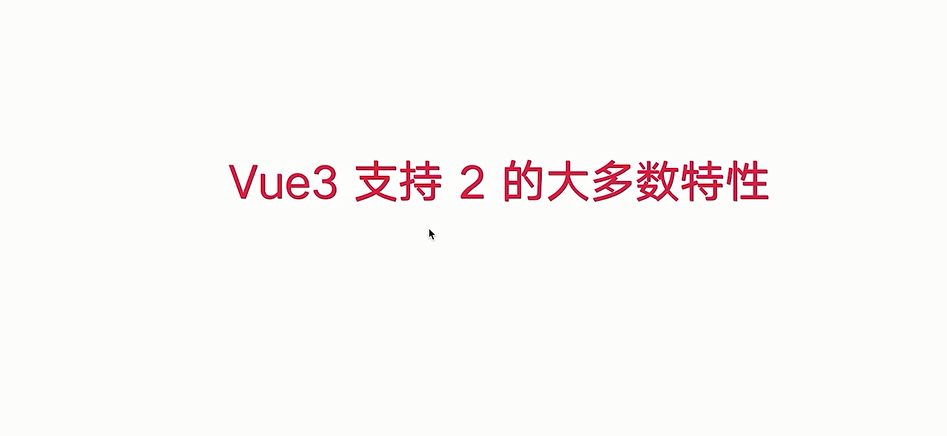


Pull requeset === PR

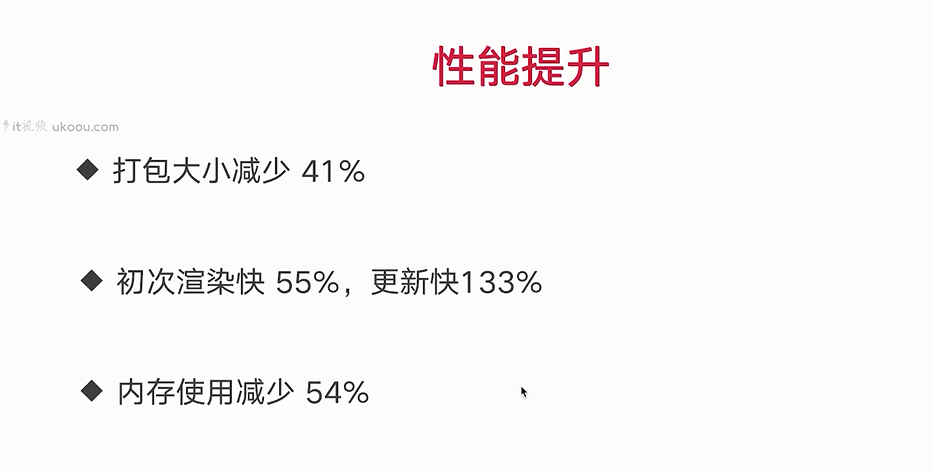


Vue3 保持vue2的大部分特性，这个遵循着渐进式的准则

Vue3官方网站：

<https://v3.vuejs.org>

文档目前还没有完善，目前只有英文版，中文版还没出来；



整个Vue3.0被重新编写，拆分成不同功能的module，根据官方网站的说法，

以上：1,2,3

这得益于重写虚拟DOM的实现 和 Tree-shaking support

------------------------------------------------------

3.0对比一2.0的区别：

<https://www.jianshu.com/p/21953b0b6ea5>

1.Performance //性能

2.Tree-shaking support //通过分析，可以使你代码里没有使用的代码全部删除，简单粗暴点就是代码优化工具，想了解自行百度

<https://zhuanlan.zhihu.com/p/143049468>

当前在2.x中，所有全局API都在单个Vue对象上公开：

**import** Vue from 'vue'

Vue.nextTick(() => {})

**const** obj **=** Vue.observable({})

在3.x中，**只能**将它们作为命名导入进行访问：

**import** Vue, { nextTick, observable } from 'vue'

Vue.nextTick *// undefined*

nextTick(() => {})

**const** obj **=** observable({})

通过ES模块的静态分析友好设计，现代化的打包机与缩减程序相结合，现在可以消除未在捆绑包中任何地方使用的ES模块导出。我们可以重组Vue的全局和内部API，以利用此优势，以便用户只为他们实际使用的功能付费。

3.Composition API //特色语法

4.Fragment, Teleport, Suspense //“碎片”，Teleport即Protal传送门，“悬念”

5.Better TypeScript support //TypeScript支持度

6.Custom Renderer API //自定义rende

背景：vue2.0复杂组件的代码变的越来越难以维护，缺少一种比较干净的，在多个组件之间提取和复用逻辑的一种机制，现在的重用机制都有一些弊端，所以VUE3,0推出了Composition API

优势：

1. 把相关的feature组合在一起，不要分散在不同的地方
2. 它可以非常方便的重用，比minxins有更高的灵活度

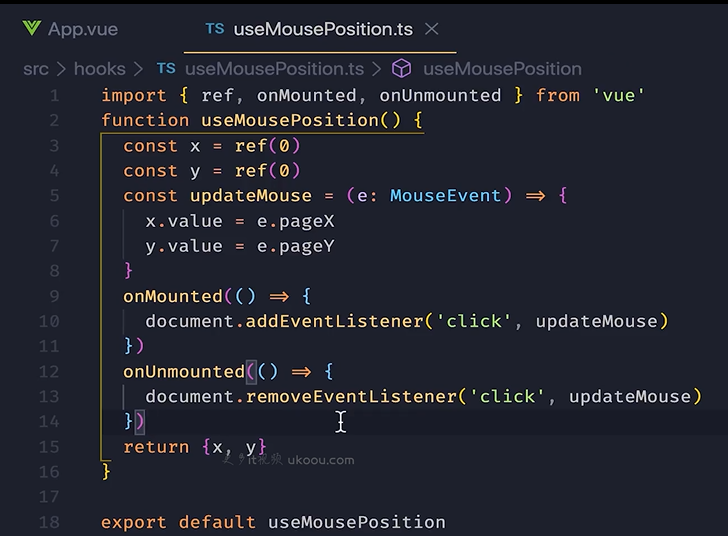
相关的代码都可以写在setup里



优化抽离----自定义hooks：

在hooks文件下创建一个useMousePosition.ts





导入：



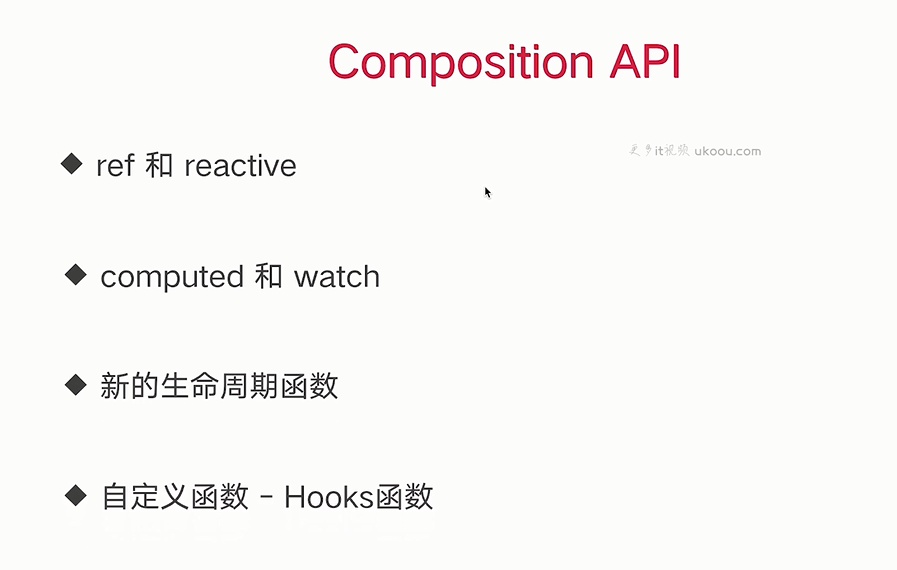
调用：

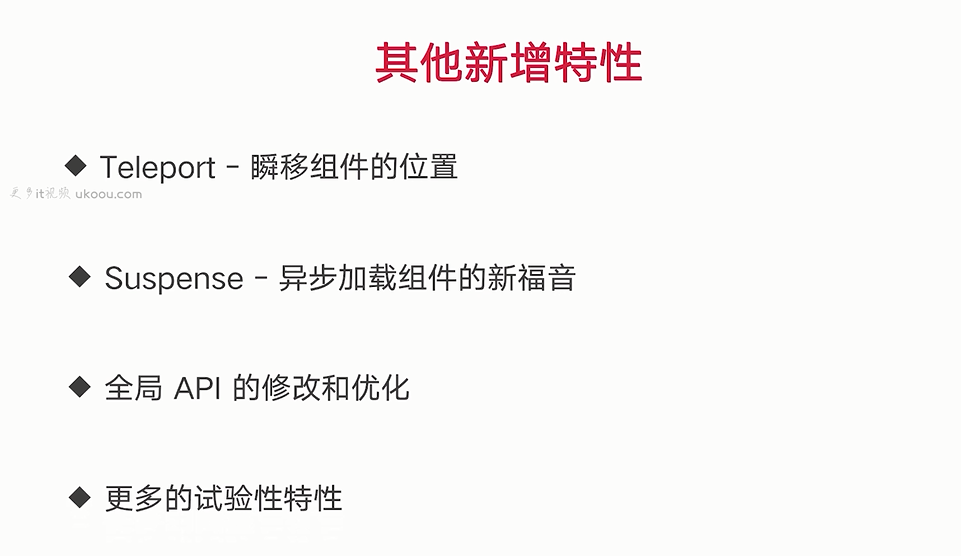


Vue3.0新推出的API

Composition组合

Composition API新推出一系列API的合集





正在实现还没有稳定的新特性，比如使用：

Script 添加一个setup属性，就可以直接使用composition API等等



使用vue2.0如果集成过typesript，就会发现非常多的痛点

因为vue2.0在推出的时候，就没有把ts作为一个考量的范围

Vue3.0才用TS重写vue源代码，提供非常完备的类型定义，

所以在使用VUE3.0进行项目开发的时候，我们可以很方便的使用类型推论，

等一系列TS的红利

同时也可以在VScode也可以安装TS插件，来享受代码提示，语法检查等功能

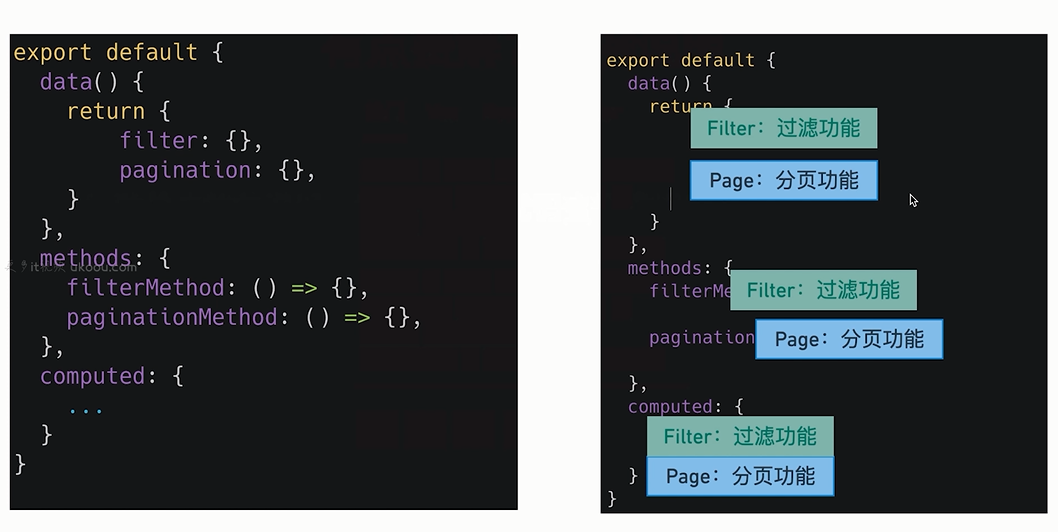
以上涵盖了Vue3.0的大体的新特性，具体详细，可以从官方了解

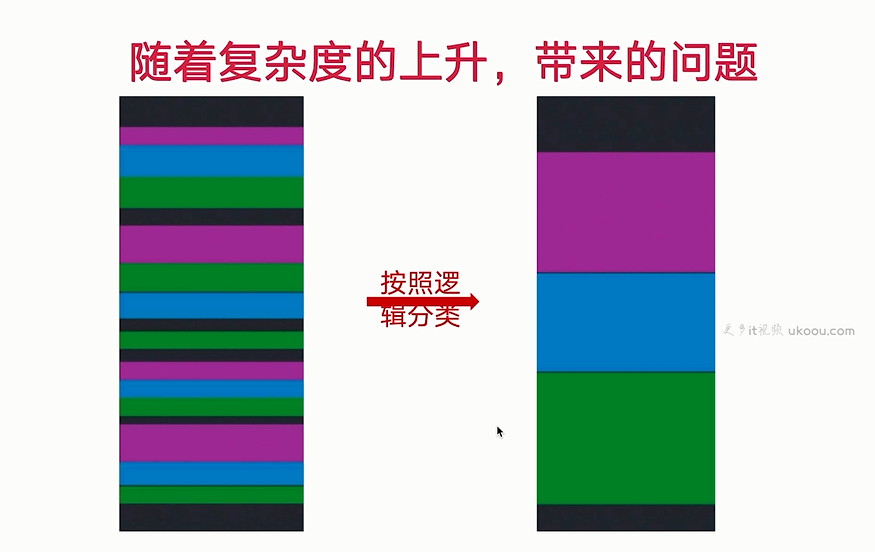
Vue3官方网站：

<https://v3.vuejs.org>

--------------------------------------------------------------------------------Vue3 Composition API诞生的动机 和 原因

和讨论Vue目前遇到的困境和问题





Vue2.0同一个功能模块，功能逻辑分散到文件不同地方

Vue3.0按照不同功能的代码，按照逻辑分类进行分割放到一起，那么代码的可读性就会的更加清晰，代码的可维护性就会变的更强；

虽然Vue2.0给了我们解决方案：mixins



虽然vue2.0的mixins解决了我们按特征归类问题，但是mixins也有对应的一些问题



这就是Vue3.0 Composition API出现的原因；它的推出，就是为了解决上述问题

Vue2.0还有一个长期困扰着开发者：



Vue2.0的

Vue依赖this对象的上下文，往外暴露的属性和方法

在Vue组件中的this比普通的javascript object多了很多黑魔法（属性和方法），

换而言之，就是Vue2.0 API在推出的时候，就没有考虑过TS集成和类型推论的问题

所以Vue2.0在集成TS的时候，会有很多的困难和险阻

然而，在Vue3.0中，源码是用TS重写过的，提供非常完备的类型定义，

所以在使用VUE3.0进行项目开发的时候，我们可以很方便的使用类型推论，

等一系列TS的红利

让TS帮助我们加速和便捷的开发出更少BUG的产品

Vue3.0开发环境搭建：

Vue-cli

<http://cli.vuejs.org/zh/>



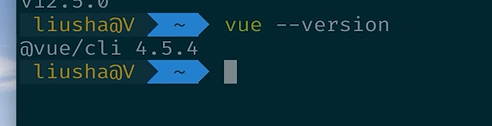
node版本要求：> v10.0.0

vue-cli版本：> 4.5.0

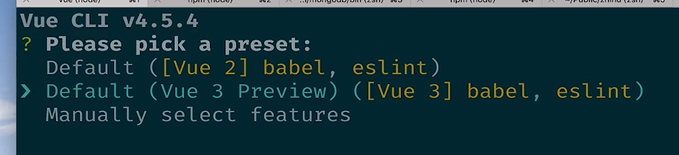
推荐使用：



本地查看vue-cli版本：

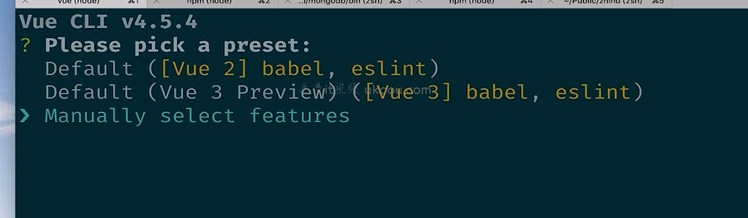


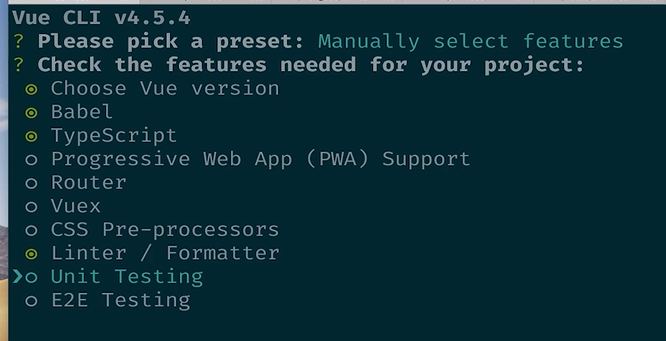
在这个版本才会有支持vue3.0的选项

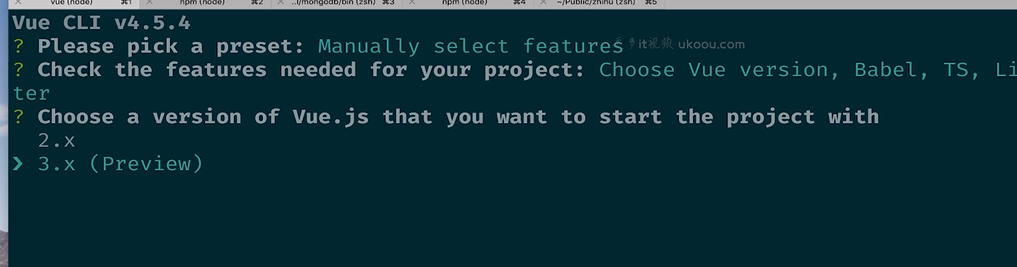


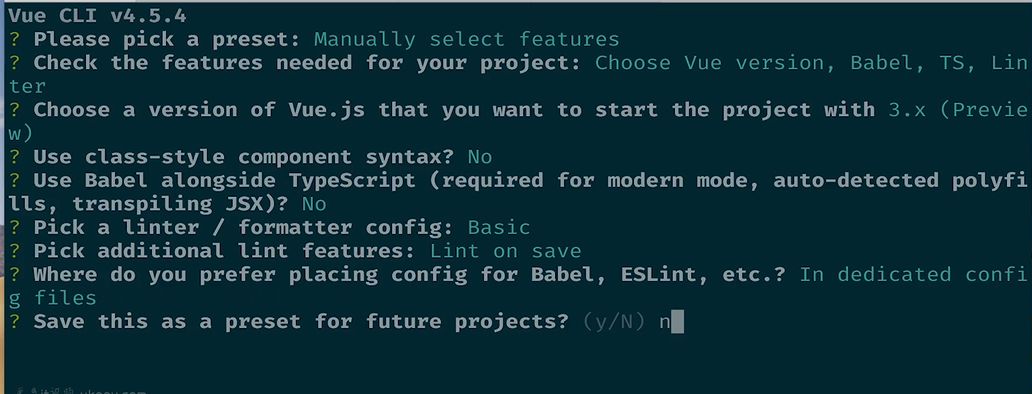
如果没有这个选项，则说明你的vue-cli是旧的版本，需要更新

Ts支持需要选择第三个手动添加特性：



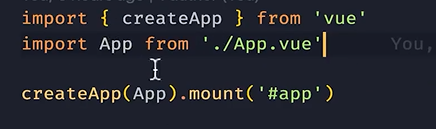






引入个组件App组件，然后通过createApp创建一个Vue app的实例，

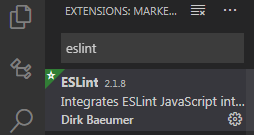
然后再挂载到#app dom上



Vscode插件：

插件1：

Eslint2.1.8



如果不生效，则在项目.vscode配置：





插件2：

Vetur

Vue官方推荐的一个插件

<https://marketplace.visualstudio.com/items?itemName=octref.vetur>

