

Vue3 & luckySheet

一、Vue的介绍

1. 什么是Vue?

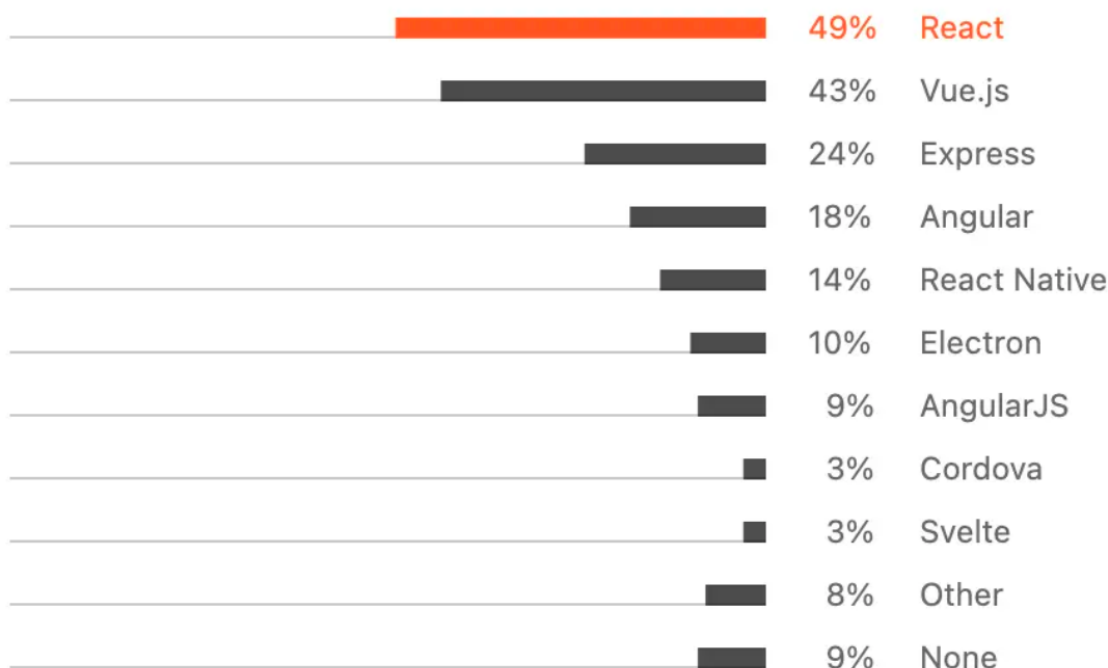
Vue(发音为 /vju:/, 类似 **view**) 是一款用于构建用户界面的 **JavaScript** 框架。它基于标准 HTML、CSS 和 JavaScript 构建, 并提供了一套声明式的、组件化的编程模型, 帮助我们高效地开发用户界面。


2. 为什么选择Vue?

1. 目前行业开发主流框架之一, 比较好用, 数据源自[Stack Overflow调查](#)

JetBrains 每年都会有程序员调查, 在前年的调查中, 最受欢迎的 JavaScript 框架是 React。不过 Vue 也在逐年上升。

What JavaScript frameworks do you regularly use?








Evan You
yyx990803

OverviewRepositories171ProjectsPa

Pinned


 **vuejs/vue**
Vue.js is a progressive, incrementally-adoptable JavaScript framework for building UI on the web.
JavaScript ☆ 183k 28.9k 

 **vitejs/vite**
Next generation frontend tooling. It's fast!
TypeScript ☆ 25.5k 1.6k

@稀土掘金技术社区

Weekly Downloads

3,339,194



Version

3.2.45

License

MIT

Unpacked Size

2.54 MB

Total Files

33

krausest 是一个测试各大框架性能的网站。

基准测试是在 MaBook Air M1 (16 GB RAM, OSX 12.5)、Chrome 104.0.5112.79 (arm64) 上使用 puppeteer 基准测试驱动程序运行的。对比如下：

Name	vue-	react-
Duration for...	v3.2.37	hooks-v18.2.0

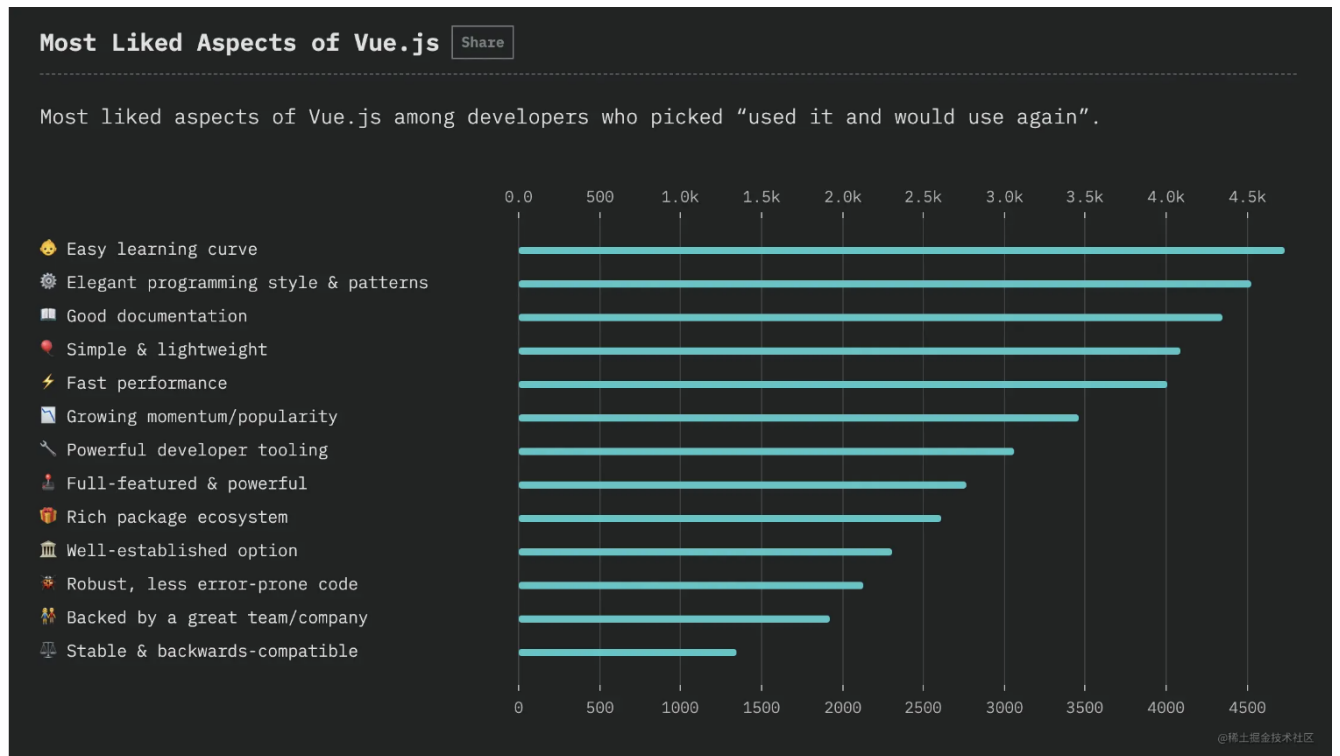
Implementation notes		
create rows creating 1,000 rows (5 warmup runs).	43.2 \pm 0.3 (1.00)	45.9 \pm 0.4 (1.06)
replace all rows updating all 1,000 rows (5 warmup runs).	43.5 \pm 0.2 (1.00)	47.3 \pm 0.3 (1.09)
partial update updating every 10th row for 1,000 rows (3 warmup runs). 16x CPU slowdown.	107.8 \pm 1.8 (1.00)	109.8 \pm 1.8 (1.02)
select row highlighting a selected row. (5 warmup runs). 16x CPU slowdown.	19.8 \pm 0.8 (1.00)	27.3 \pm 1.1 (1.38)
swap rows swap 2 rows for table with 1,000 rows. (5 warmup runs). 4x CPU slowdown.	28.1 \pm 0.6 (1.00)	155.1 \pm 0.8 (5.52)
remove row removing one row. (5 warmup runs).	12.3 \pm 1.1 (1.04)	11.9 \pm 0.4 (1.00)
create many rows creating 10,000 rows. (5	471.4 \pm 1.5	621.5 \pm 1.6

warmup runs with 1k rows).	(1.00)	(1.32)
append rows to large table appending 1,000 to a table of 10,000 rows. 2x CPU slowdown.	93.9 \pm 0.5 (1.00)	105.5 \pm 0.6 (1.12)
clear rows clearing a table with 1,000 rows. 8x CPU slowdown. (5 warmup runs).	29.6 \pm 0.9 (1.00)	53.6 \pm 0.7 (1.81)
<u>geometric mean</u> of all factors in the table	1.00	1.42
compare: Green means significantly faster, red significantly slower	<u>com-</u> <u>pare</u>	<u>com-</u> <u>pare</u> <small>@稀土掘金技术社区</small>

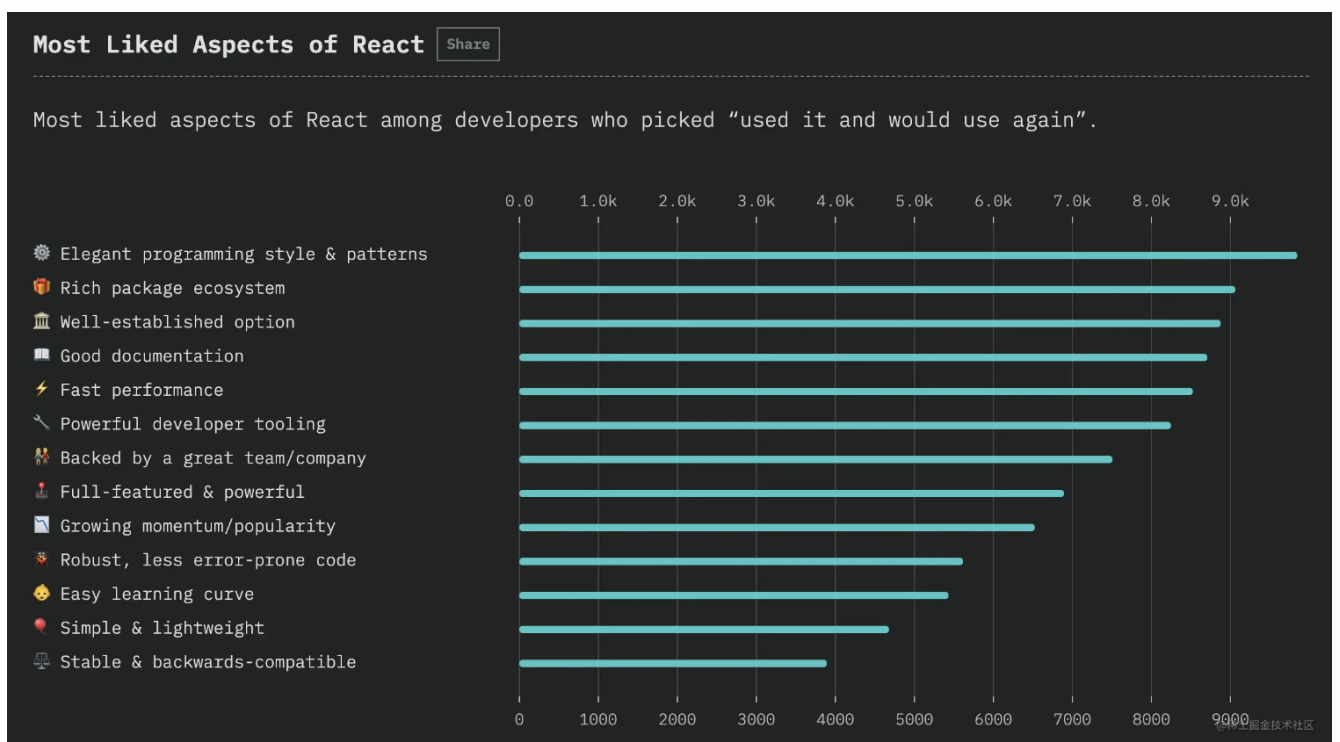
结论：可以看到 Vue3.2 会比 React 18.2 的性能更强

2. Vue易于学习，优雅的编程风格和图案，好的学习文档（相比React）[来源](#)

Vue



React



3. Vue生态系统发展好，比如利用Vue开发的UI组件element-plus、ant-design-vue,状态管理vuex、pinia,路由管理vue-router，React没有用于路由或状态管理的官方软件包。

3. Vue2和Vue3的区别?

- Vue3的响应式使用proxy对数据进行代理
- 使用TS开发, 使得类型推导更加友好
- 代码编译使用静态提升和动态节点标记, 提升渲染效率
- 源码体积优化, 移除了过滤器filter, inline-template, 支持代码的tree-shaking, 减少打包体积
- 架构上采用monorepo方式, 拆包开发, 易于维护, 局部可打包更新
- Vue3 性能提升了 1.3~2 倍, ssr快了2~3倍

二、Vue项目使用

1. 代码获取

2. 安装依赖

- 代码拉到本地后, 在项目中打开终端窗口, 输入yarn回车安装项目所需依赖, 安装好后会多出一个node_modules的文件
- 启动项目, 在终端窗口中输入yarn dev启动项目, 启动后浏览器中输入http://localhost:8080/即可打开项目

3. 项目结构



- > icons **svg图标**
- > layout
- > router **路由**
- > store **数据存储**
- > styles **全局样式**
- > utils **开发工具**
- > views **页面组件**

App.vue

JS main.js

JS settings.js

4. 请求数据&页面点击交互（传参）的demo

```
<template>
  <div class="demo">
    <h3>this is a demo</h3>
    <p></p>
    <div>
      <button @click="add">add</button>
      <input type="number" v-model="state.num" />
      <button @click="reduce">reduce</button>
    </div>
    <p class="text-color">
      input number double:
      <span class="num">{{ double }}</span>
    </p>
  </div>
</template>
<script setup name="Demo">
import { reactive,computed } from 'vue'
const state = reactive({ num: 0 })
const double = computed(() => state.num * 2)
function add() {
  state.num++
}
```

```

}
function reduce() {
  state.num--
}
</script>
<style lang="scss" scoped>
.text-color {
  .num {
    color: red;
  }
}
</style>

```

三、Luckysheet应用

1.Luckysheet介绍

Luckysheet，一款纯前端类似excel的在线表格，功能强大、配置简单、完全开源。[官网](#)、[在线Demo](#)

2.单元格配置

Luckysheet的数据是一个二维数组，二维数组的个数代表Luckysheet的行数，二维数组里的每一项是单元格配置项，包括了显示值、批注、公式、单元合并等配置字段，以下是实例：

```

const sheetData = [
  [
    {
      "m": "其他相关信息-1", // 显示值
      "v": "其他相关信息-1", // 原始值
      "bl": 1, // 字体粗细 0 常规、1加粗
      "ff": 1, // 字体 1 Arial、2 Tahoma、3 Verdana、4 微软雅黑、5 宋体 (Song)
      "fc": 'red', // 字体颜色
      "ct": { // 单元格值格式 - 当前设置为纯文本
        "fa": "@", // Format格式的定义串
        "t": "s" // 格式类型
      },
      "ht": 0, // 水平对齐 0 居中、1 左、2右
      "ps": { // 批注
        "width": 120, // 批注框宽度
        "height": 48, // 批注框高度
        "value": "支持代码和中文名称", // 批注内容
      },
      "f": "=sheet2!C2+sheet2!C3", // 单元格是一个引用sheet求和公式
      "f": "=SUM(A3+B3)", // 单元格是一个求和公式
      "mc": { // 合并单元格必备属性
        "r": 0, // 主单元格的行号
        "c": 64, // 主单元格的列号
        "rs": 1, // 合并单元格占的行数
        "cs": 3 // 合并单元格占的列数
      }
    }
  ]
]

```



```

    },
  },
  {
    "mc": {
      "r": 0, // 主单元格的行号
      "c": 64 // 主单元格的列号
    }
  },
  {
    "mc": {
      "r": 0, // 主单元格的行号
      "c": 64 // 主单元格的列号
    }
  },
]
]

```

配置注意事项：

1. 返回的单元格值格式ct的值须为{"fa": "@", "t": "s"},避免回显出现科学计数值等
2. 当公式引用其他sheet的值时，若引入的值是数字加字母的组合（1234567890m）,需要用双引号包裹而不是单引号
3. 合并单元格时，合并单元格占的列数和占位用的单元格配置项数量匹配，合并单元格的下一单元格的列号避免写错

3.数据加载

```

async function initSheet(sheetData) {
  if (!sheetData?.length) return
  luckysheet.destroy()
  const colLen = sheetData[0].length
  const row = [2, sheetData.length - 1]
  await nextTick()
  > sheet1 = genSheetData({ ...
  })
  options = {
    ...genSheetOptions({
      container: 'zzh',
      enableAddRow: false, //允许添加行
  > ... hook: { ...
    }
  }),
  data: [{ ...sheet1, celldata: luckysheet.transToCellData(sheetData) }]
  }
  luckysheet.create(options)
  return true
}

```

