使用Vue构建CLI应用

将Vue渲染到命令行界面上的相关实践

蔡少辉

VueConf
2022年12月10日

webfansplz

Vue.js & VueUse 团队成员

Vue Challenges, Temir 等项目作者

webfansplz

webfansplz

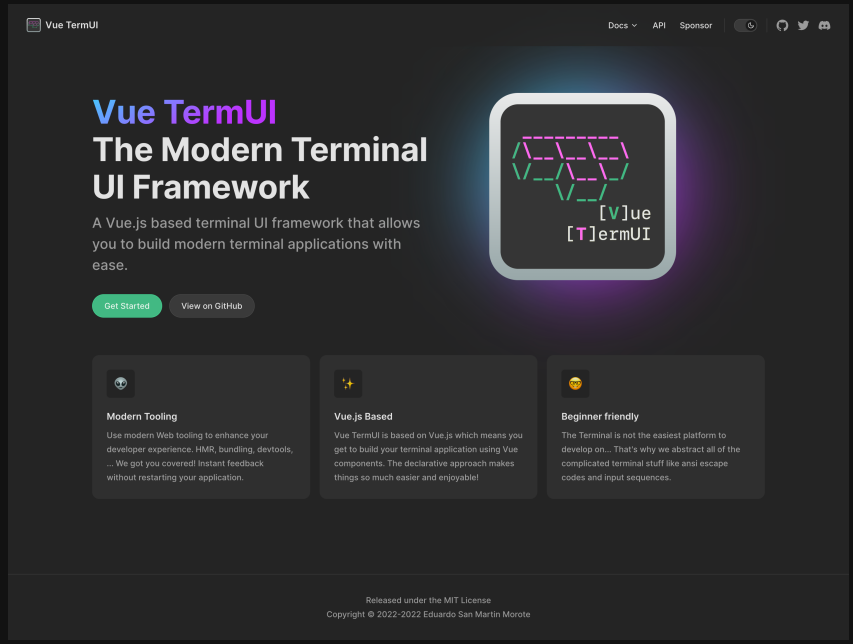
知 webfansplz


使用 Vue 构建终端应用
⌥⌘4
vue-conf-1 •
~/g/v/packages/playground
18%
VueConf 2022
主题名：
演讲者：
开始分享

Vue TermUI

一个基于Vue.js的终端UI框架，帮助你轻松构建终端应用。

https://github.com/vue-terminal/vue-termui



使用Node.js构建终端UI的痛点

- ✖ 上手成本
- ✖ 开发者体验
- ✖ 定制化布局

使用Node.js构建终端UI的痛点

- ✖ 上手成本
- ✖ 开发者体验
- ✖ 定制化布局

I think it's not as convenient as building web apps with a declarative approach. That's all I want: a declarative approach similar to vue that is simple to use

Anthony Fu @antfu7 · Aug 9

Replying to @webfansplz @vuejs and @vite_js

Building CLI and managing the colors/updates was always a hard experience.

使用Node.js构建终端UI的痛点

- ✖ 上手成本
- ✖ 开发者体验
- ✖ 定制化布局

Node.js : 关注过程

Vue.js : 关注结果

I think it's not as convenient as building web apps with a declarative approach. That's all I want: a declarative approach similar to vue that is simple to use

Anthony Fu @antfu7 · Aug 9

Replying to @webfansplz @vuejs and @vite_js

Building CLI and managing the colors/updates was always a hard experience.

使用Node.js构建终端UI的痛点

- ✖ 上手成本
- ✖ 开发者体验
- ✖ 定制化布局

Node.js : 关注过程

Vue.js : 关注结果

I think it's not as convenient as building web apps with a declarative approach. That's all I want: a declarative approach similar to vue that is simple to use

- 字符串拼接
- 局部更新处理
- 字符串长度计算

...

Anthony Fu @antfu7 · Aug 9

Replying to @webfansplz @vuejs and @vite_js

Building CLI and managing the colors/updates was always a hard experience.

Count: 3

```
import chalk from 'chalk'
import { createLogUpdate } from 'log-update'
const log = createLogUpdate(process.stdout)
const createCounterRenderer = (max: number) => {
  let timer: NodeJS.Timer
  let count = 0
  function stop() {
    clearInterval(timer)
    log.clear()
  }
  function update() {
    count++
    if (count >= max) stop()
    log('Count: ' + chalk.yellow(count))
  }
  return {
    start() {
      timer = setInterval(update, 150)
    },
  }
}
const renderder = createCounterRenderer(100)
renderder.start()
```

Count: 3

```
import chalk from 'chalk'
import { createLogUpdate } from 'log-update'
const log = createLogUpdate(process.stdout)
const createCounterRenderer = (max: number) => {
  let timer: NodeJS.Timer
  let count = 0
  function stop() {
    clearInterval(timer)
    log.clear()
  }
  function update() {
    count++
    if (count >= max) stop()
    log('Count: ' + chalk.yellow(count))
  }
  return {
    start() {
      timer = setInterval(update, 150)
    },
  }
}
const renderder = createCounterRenderer(100)
renderder.start()
```

Count: 3

```
import chalk from 'chalk'
import { createLogUpdate } from 'log-update'
const log = createLogUpdate(process.stdout)
const createCounterRenderer = (max: number) => {
  let timer: NodeJS.Timer
  let count = 0
  function stop() {
    clearInterval(timer)
    log.clear()
  }
  function update() {
    count++
    if (count >= max) stop()
    log('Count: ' + chalk.yellow(count))
  }
  return {
    start() {
      timer = setInterval(update, 150)
    },
  }
}
const renderder = createCounterRenderer(100)
renderder.start()
```

Count: 3

```
import chalk from 'chalk'
import { createLogUpdate } from 'log-update'
const log = createLogUpdate(process.stdout)
const createCounterRenderer = (max: number) => {
  let timer: NodeJS.Timer
  let count = 0
  function stop() {
    clearInterval(timer)
    log.clear()
  }
  function update() {
    count++
    if (count >= max) stop()
    log('Count: ' + chalk.yellow(count))
  }
  return {
    start() {
      timer = setInterval(update, 150)
    },
  }
}
const renderder = createCounterRenderer(100)
renderder.start()
```

Count: 3

```
import chalk from 'chalk'
import { createLogUpdate } from 'log-update'
const log = createLogUpdate(process.stdout)
const createCounterRenderer = (max: number) => {
  let timer: NodeJS.Timer
  let count = 0
  function stop() {
    clearInterval(timer)
    log.clear()
  }
  function update() {
    count++
    if (count >= max) stop()
    log('Count: ' + chalk.yellow(count))
  }
  return {
    start() {
      timer = setInterval(update, 150)
    },
  }
}
const renderder = createCounterRenderer(100)
renderder.start()
```

Count: 3

```ts
import chalk from 'chalk'
import { createLogUpdate } from 'log-update'
const log = createLogUpdate(process.stdout)
const createCounterRenderer = (max: number) => {
  let timer: NodeJS.Timer
  let count = 0
  function stop() {
    clearInterval(timer)
    log.clear()
  }
  function update() {
    count++
    if (count >= max) stop()
    log('Count: ' + chalk.yellow(count))
  }
  return {
    start() {
      timer = setInterval(update, 150)
    },
  }
}
const renderder = createCounterRenderer(100)
renderder.start()
```

```vue
<script setup lang="ts">
import { ref } from '@vue/runtime-core'
const count = ref(0)
const timer = setInterval(()=>{
  count.value++
  if(count.value >=100) clearInterval(timer)
},150)
</script>

<template>
  <Text>
    Count: <Text color="yellow">{{count}}</Text>
  </Text>
</template>
```

Count: 3

```ts
import chalk from 'chalk'
import { createLogUpdate } from 'log-update'
const log = createLogUpdate(process.stdout)
const createCounterRenderer = (max: number) => {
  let timer: NodeJS.Timer
  let count = 0
  function stop() {
    clearInterval(timer)
    log.clear()
  }
  function update() {
    count++
    if (count >= max) stop()
    log('Count: ' + chalk.yellow(count))
  }
  return {
    start() {
      timer = setInterval(update, 150)
    },
  }
}
const renderder = createCounterRenderer(100)
renderder.start()
```

```vue
<script setup lang="ts">
import { ref } from '@vue/runtime-core'
const count = ref(0)
const timer = setInterval(()=>{
  count.value++
  if(count.value >=100) clearInterval(timer)
},150)
</script>

<template>
  <Text>
    Count: <Text color="yellow">{{count}}</Text>
  </Text>
</template>
```

Count: 2

```
import chalk from 'chalk'
import { createLogUpdate } from 'log-update'
const log = createLogUpdate(process.stdout)
const createCounterRenderer = (max: number) => {
  let timer: NodeJS.Timer
  let count = 0
  function stop() {
    clearInterval(timer)
    log.clear()
  }
  function update() {
    count++
    if (count >= max) stop()
    log('Count: ' + chalk.yellow(count))
  }
  return {
    start() {
      timer = setInterval(update, 150)
    },
  }
}
const renderder = createCounterRenderer(100)
renderder.start()
```

→

```
<script setup lang="ts">
import { ref } from '@vue/runtime-core'
const count = ref(0)
const timer = setInterval(()=>{
  count.value++
  if(count.value >=100) clearInterval(timer)
},150)
</script>

<template>
  <Text>
    Count: <Text color="yellow">{{count}}</Text>
  </Text>
</template>
```

Count: 2

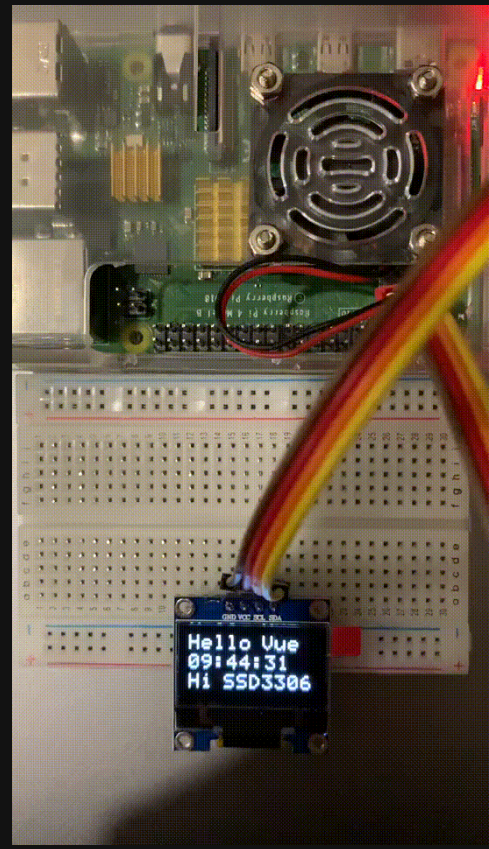
```ts
import chalk from 'chalk'
import { createLogUpdate } from 'log-update'
const log = createLogUpdate(process.stdout)
const createCounterRenderer = (max: number) => {
  let timer: NodeJS.Timer
  let count = 0
  function stop() {
    clearInterval(timer)
    log.clear()
  }
  function update() {
    count++
    if (count >= max) stop()
    log('Count: ' + chalk.yellow(count))
  }
  return {
    start() {
      timer = setInterval(update, 150)
    },
  }
}
const renderder = createCounterRenderer(100)
renderder.start()
```

→

```vue
<script setup lang="ts">
import { ref } from '@vue/runtime-core'
const count = ref(0)
const timer = setInterval(()=>{
  count.value++
  if(count.value >=100) clearInterval(timer)
},150)
</script>

<template>
  <Text>
    Count: <Text color="yellow">{{count}}</Text>
  </Text>
</template>
```

Vue能渲染到命令行界面上吗?

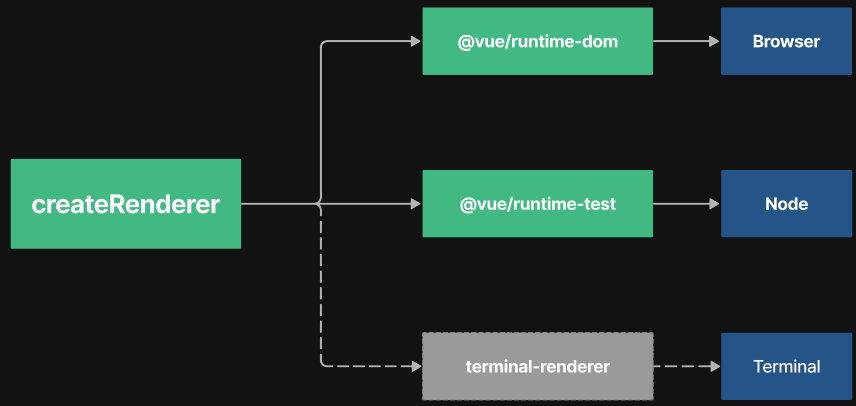

Hello Vue
09:44:31
Hi SSD3306

自定义渲染器

```
import { createRenderer } from '@vue/runtime-core'

const { render, createApp } = createRenderer({
  patchProp,
  insert,
  remove,
  createElement
  // ...
})

export { render, createApp }
```



如何在命令行界面上进行布局?

Yoga

基于Flexbox的跨平台布局引擎

- 布局灵活
- 简单易用
- 功能强大

Yoga

基于Flexbox的跨平台布局引擎

- 布局灵活
- 简单易用
- 功能强大



...

核心实现

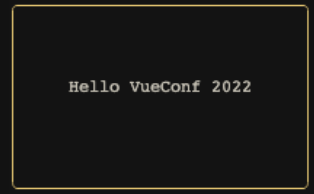
```ts
// renderer.ts
import {
  createRenderer
} from '@vue/runtime-core'
import Yoga from 'yoga-layout-prebuilt'


export const renderer = createRenderer({
  createElement() {
    Yoga.Node.create()
  },
  insert(childNode, node) {
    node.yogaNode?.insertChild(
      childNode.yogaNode,
      node.yogaNode?.getChildCount(),
    )
  },
  patchProp(el,key,_, { margin }) {
    if (key === 'margin') {
      el.yogaNode.setMargin(margin)
    }
  },
  // ...
})
```

```ts
// main.ts
import {
  renderer
} from './renderer.ts'
import App from './App.vue'
renderer.createApp(App)

// App.vue
<script setup lang="ts">
import { TuiBox, TuiText } from 'vue-termui'
</script>

<template>
  <TuiBox
    :width="30"
    :height="10"
    border-style="round"
    border-color="yellow"
    justify-content="center"
    align-items="center"
  >
    <TuiText>Hello VueConf 2022</TuiText>
  </TuiBox>
</template>
```



核心实现

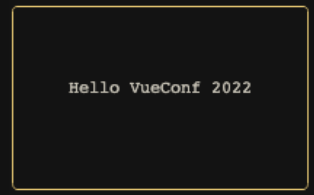
```ts
// renderer.ts
import {
  createRenderer
} from '@vue/runtime-core'
import Yoga from 'yoga-layout-prebuilt'


export const renderer = createRenderer({
  createElement() {
    Yoga.Node.create()
  },
  insert(childNode, node) {
    node.yogaNode?.insertChild(
      childNode.yogaNode,
      node.yogaNode?.getChildCount(),
    )
  },
  patchProp(el,key,_, { margin }) {
    if (key === 'margin') {
      el.yogaNode.setMargin(margin)
    }
  },
  // ...
})
```

```ts
// main.ts
import {
  renderer
} from './renderer.ts'
import App from './App.vue'
renderer.createApp(App)

// App.vue
<script setup lang="ts">
import { TuiBox, TuiText } from 'vue-termui'
</script>

<template>
  <TuiBox
    :width="30"
    :height="10"
    border-style="round"
    border-color="yellow"
    justify-content="center"
    align-items="center"
  >
    <TuiText>Hello VueConf 2022</TuiText>
  </TuiBox>
</template>
```



核心实现

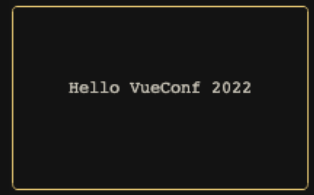
```ts
// renderer.ts
import {
  createRenderer
} from '@vue/runtime-core'
import Yoga from 'yoga-layout-prebuilt'


export const renderer = createRenderer({
  createElement() {
    Yoga.Node.create()
  },
  insert(childNode, node) {
    node.yogaNode?.insertChild(
      childNode.yogaNode,
      node.yogaNode?.getChildCount(),
    )
  },
  patchProp(el,key,_, { margin }) {
    if (key === 'margin') {
      el.yogaNode.setMargin(margin)
    }
  },
  // ...
})
```

```ts
// main.ts
import {
  renderer
} from './renderer.ts'
import App from './App.vue'
renderer.createApp(App)

// App.vue
<script setup lang="ts">
import { TuiBox, TuiText } from 'vue-termui'
</script>

<template>
  <TuiBox
    :width="30"
    :height="10"
    border-style="round"
    border-color="yellow"
    justify-content="center"
    align-items="center"
  >
    <TuiText>Hello VueConf 2022</TuiText>
  </TuiBox>
</template>
```



核心实现

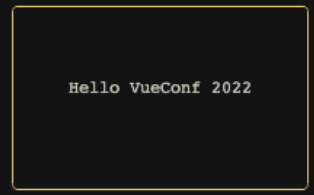
```
// renderer.ts
import {
  createRenderer
} from '@vue/runtime-core'
import Yoga from 'yoga-layout-prebuilt'


export const renderer = createRenderer({
  createElement() {
    Yoga.Node.create()
  },
  insert(childNode, node) {
    node.yogaNode?.insertChild(
      childNode.yogaNode,
      node.yogaNode?.getChildCount(),
    )
  },
  patchProp(el,key,_, { margin }) {
    if (key === 'margin') {
      el.yogaNode.setMargin(margin)
    }
  },
  // ...
})
```

```
// main.ts
import {
  renderer
} from './renderer.ts'
import App from './App.vue'
renderer.createApp(App)

// App.vue
<script setup lang="ts">
import { TuiBox, TuiText } from 'vue-termui'
</script>

<template>
  <TuiBox
    :width="30"
    :height="10"
    border-style="round"
    border-color="yellow"
    justify-content="center"
    align-items="center"
  >
    <TuiText>Hello VueConf 2022</TuiText>
  </TuiBox>
</template>
```



核心实现

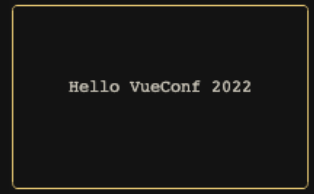
```ts
// renderer.ts
import {
  createRenderer
} from '@vue/runtime-core'
import Yoga from 'yoga-layout-prebuilt'


export const renderer = createRenderer({
  createElement() {
    Yoga.Node.create()
  },
  insert(childNode, node) {
    node.yogaNode?.insertChild(
      childNode.yogaNode,
      node.yogaNode?.getChildCount(),
    )
  },
  patchProp(el,key,_, { margin }) {
    if (key === 'margin') {
      el.yogaNode.setMargin(margin)
    }
  },
  // ...
})
```

```ts
// main.ts
import {
  renderer
} from './renderer.ts'
import App from './App.vue'
renderer.createApp(App)
```

```vue
// App.vue
<script setup lang="ts">
import { TuiBox, TuiText } from 'vue-termui'
</script>

<template>
  <TuiBox
    :width="30"
    :height="10"
    border-style="round"
    border-color="yellow"
    justify-content="center"
    align-items="center"
  >
    <TuiText>Hello VueConf 2022</TuiText>
  </TuiBox>
</template>
```



核心实现

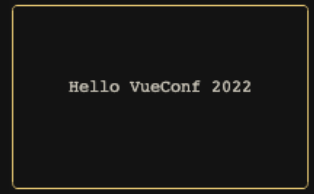
```ts
// renderer.ts
import {
  createRenderer
} from '@vue/runtime-core'
import Yoga from 'yoga-layout-prebuilt'


export const renderer = createRenderer({
  createElement() {
    Yoga.Node.create()
  },
  insert(childNode, node) {
    node.yogaNode?.insertChild(
      childNode.yogaNode,
      node.yogaNode?.getChildCount(),
    )
  },
  patchProp(el,key,_, { margin }) {
    if (key === 'margin') {
      el.yogaNode.setMargin(margin)
    }
  },
  // ...
})
```

```ts
// main.ts
import {
  renderer
} from './renderer.ts'
import App from './App.vue'
renderer.createApp(App)

// App.vue
<script setup lang="ts">
import { TuiBox, TuiText } from 'vue-termui'
</script>

<template>
  <TuiBox
    :width="30"
    :height="10"
    border-style="round"
    border-color="yellow"
    justify-content="center"
    align-items="center"
  >
    <TuiText>Hello VueConf 2022</TuiText>
  </TuiBox>
</template>
```



核心功能

- 内置组件 (Components)
- 组合式函数 (Composables)

内置组件 (Components)

- Text
- Box
- Newline
- Link
- ProgressBar
- Input
- ...

组合式函数 (Composables)

- onKeyData
- onInputData
- onMouseData
- useFocus
- …

开发者体验

- Dev Server & HMR
- 开箱即用能力
- Vue Devtools

Vite Node

Vite的Node.js运行时

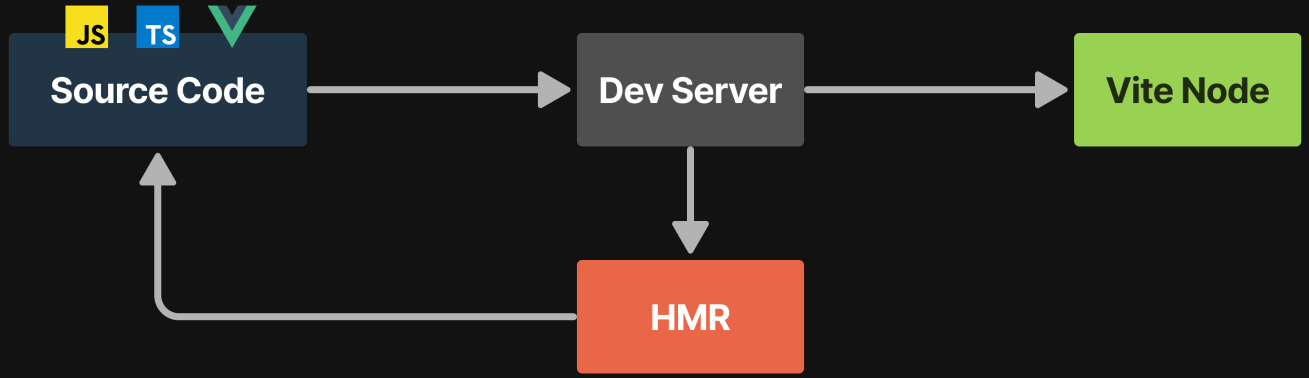
- 按需执行
- 复用Vite配置与插件
- 开箱即用的ESM与TypeScript支持
- HMR支持

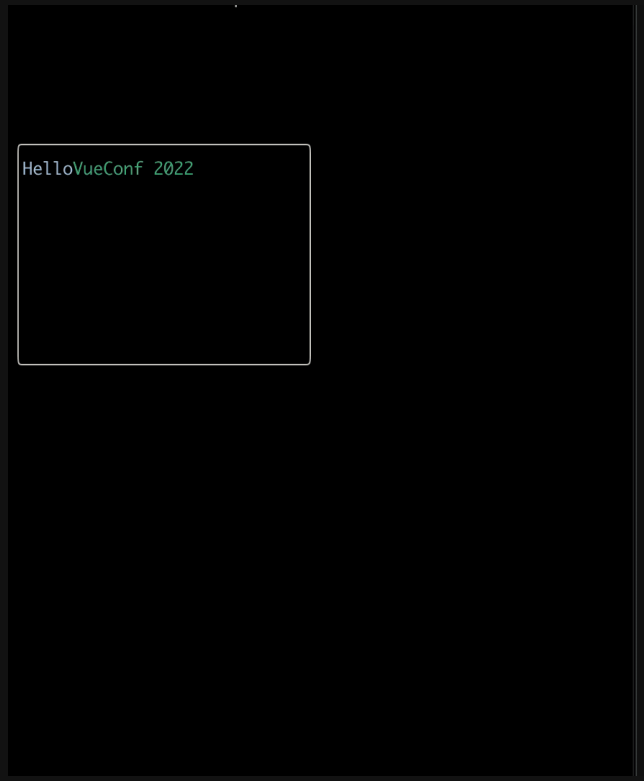
Vite Node

Vite的Node.js运行时

- 按需执行
- 复用Vite配置与插件
- 开箱即用的ESM与TypeScript支持
- HMR支持

Dev Server & HMR




HelloVueConf 2022
VueConf2022.vue M
packages > playground > src > VueConf2022.vue > {} template > div > He
setup sugar ☑ | ref sugar ☐
<script setup lang="ts">
import Hello from './Hello.vue'
const n = ref(0)
</script>
<template>
<div
:width="30"
:height="10"
:margin-top="5"
border-style="round"
>
<Hello />
bfansplz [11 days ago] • no message

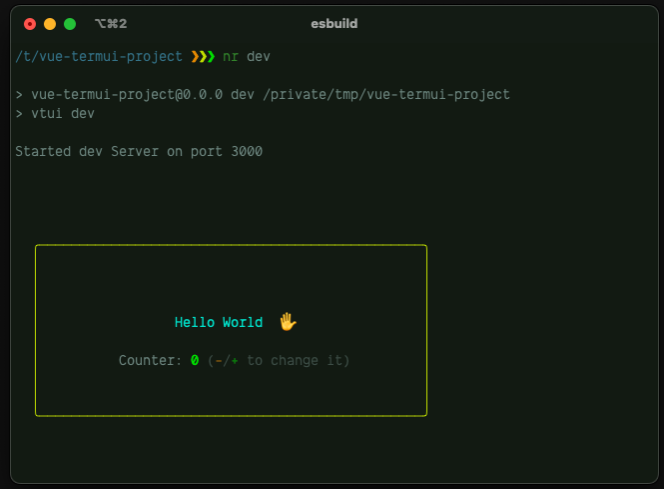
</template>

开箱即用能力

- 起手式模版
- CLI支持
- Vite插件
- "原子化Style"

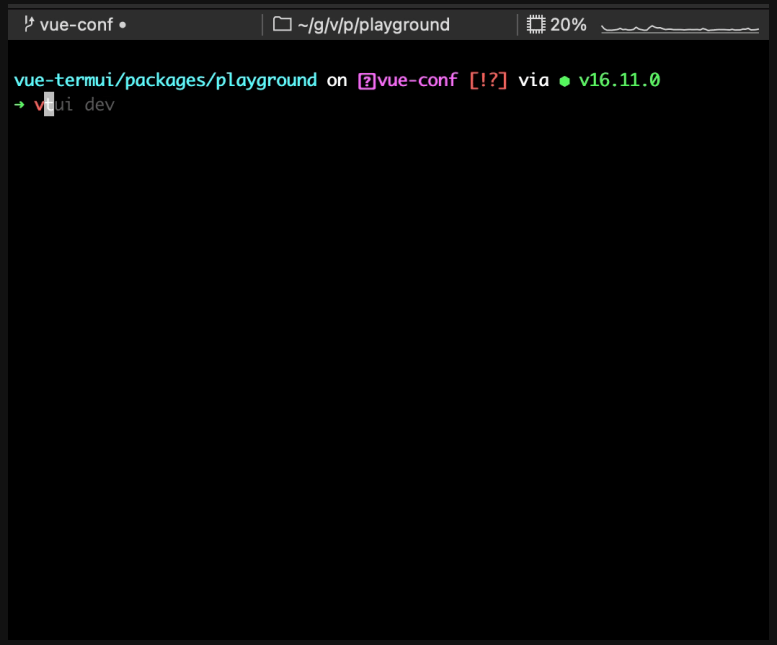
起手式模板

- Quick Start
- npm create vue-termui



CLI支持

- 启动开发服务
- 构建生产版本



Vite插件

- 基础配置集成
- 组件与API自动导入
- 组件别名

```vue
<script setup>
import { TuiBox, TuiText, TuiNewLine } from 'vue-termui'
</script>

<template>
  <TuiBox
    border-style="double"
    border-color="#213547"
    flex-direction="column"
    align-items="center"
   >
    <TuiText color="#aac8e4">Hello</TuiText>
    <TuiNewLine />
    <TuiText color="#42b883">VueConf 2022</TuiText>
  </TuiBox>
</template>
```

Vite插件

- 基础配置集成
- 组件与API自动导入
- 组件别名

```vue
<script setup>
import { TuiBox, TuiText, TuiNewLine } from 'vue-termui'
</script>

<template>
  <TuiBox
    border-style="double"
    border-color="#213547"
    flex-direction="column"
    align-items="center"
   >
    <TuiText color="#aac8e4">Hello</TuiText>
    <TuiNewLine />
    <TuiText color="#42b883">VueConf 2022</TuiText>
  </TuiBox>
</template>
```

"原子化Style"

```
<template>
  <div
    :width="30"
    :height="10"
    border-style="double"
    border-color="green"
    flex-direction="column"
    align-items="center"
    justify-content="center"
  >
    <span color="cyan">Hello</span>
    <br />
    <span color="yellow">VueConf 2022</span>
  </div>
</template>
```

```
<template>
  <div class="
    w-30
    h-10
    border-double
    border-green
    flex-col
    items-center
    justify-center
    ">
    <span text-cyan>Hello</span>
    <br />
    <span text-yellow>VueConf 2022</span>
  </div>
</template>
```

"原子化Style"

```
<template>
  <div
    :width="30"
    :height="10"
    border-style="double"
    border-color="green"
    flex-direction="column"
    align-items="center"
    justify-content="center"
  >
    <span color="cyan">Hello</span>
    <br />
    <span color="yellow">VueConf 2022</span>
  </div>
</template>
```

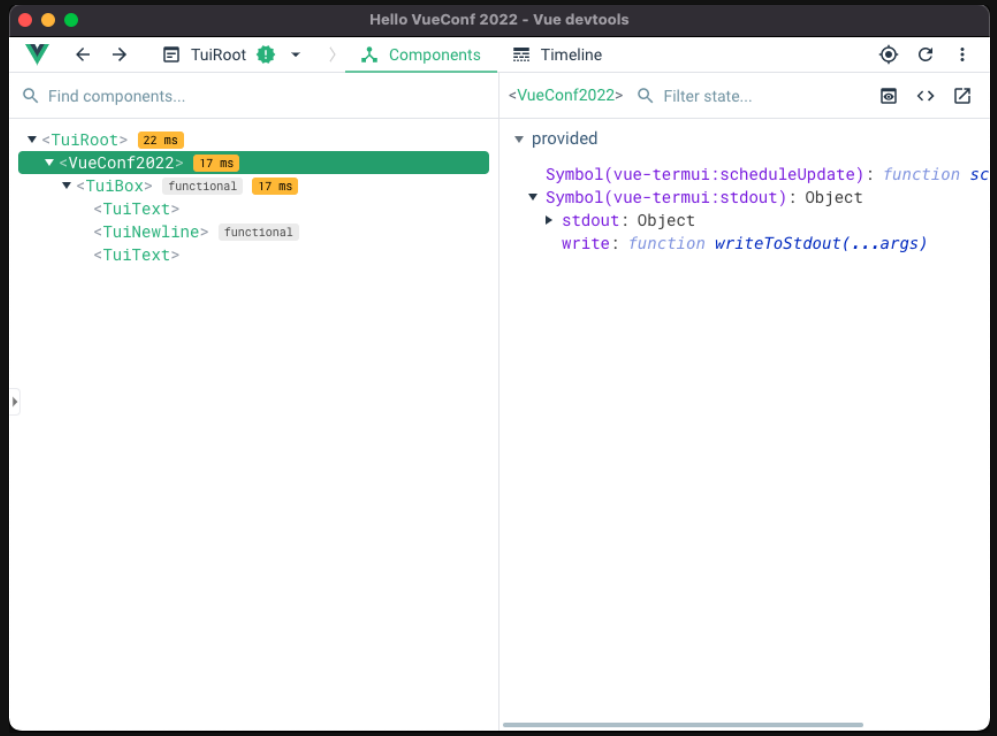
```
<template>
  <div class="
    w-30
    h-10
    border-double
    border-green
    flex-col
    items-center
    justify-center
    ">
    <span text-cyan>Hello</span>
    <br />
    <span text-yellow>VueConf 2022</span>
  </div>
</template>
```

Vue Devtools

调试体验

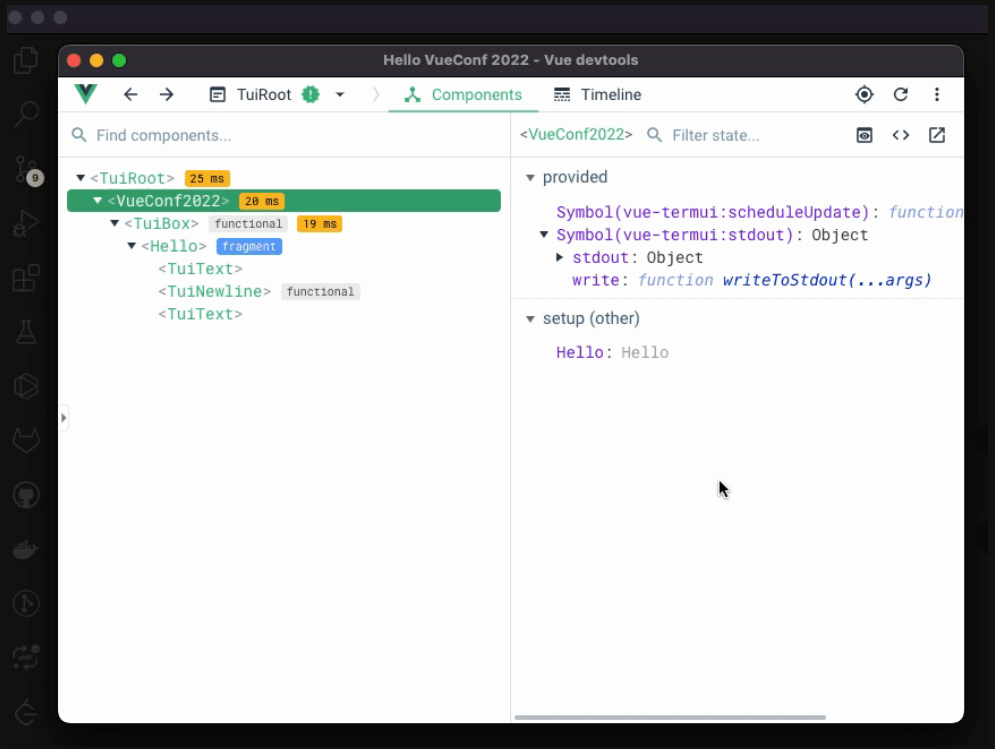
Vue Devtools

调试体验



Vue Devtools

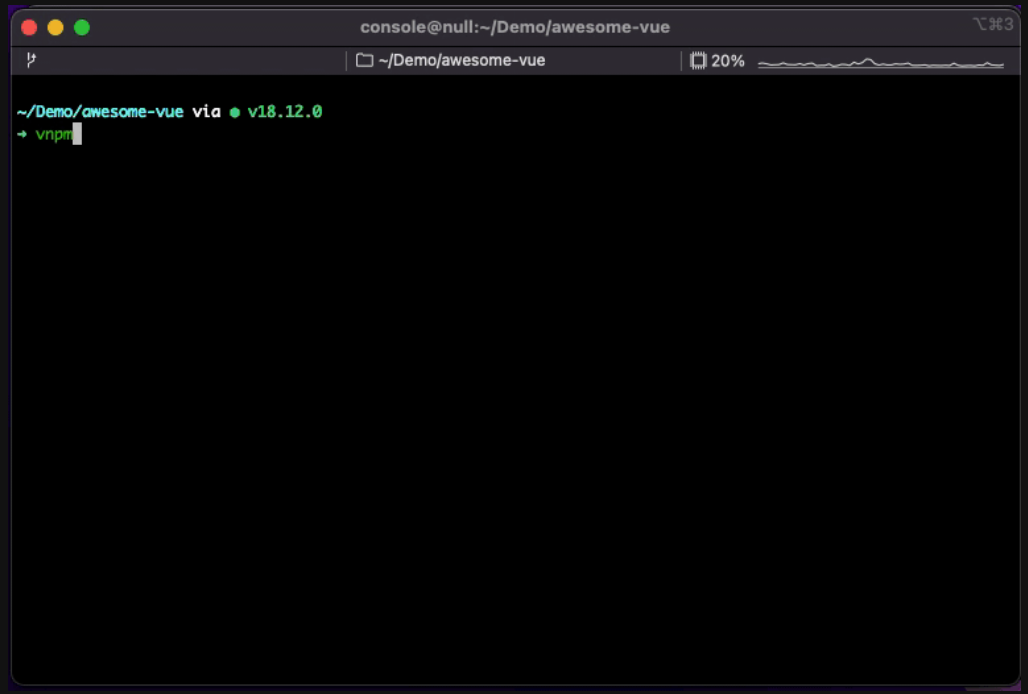
调试体验



实践分享

交互式命令行工具

搜索和安装各种 JS 的 npm 模块 (https://github.com/webfansplz/vtui-npm)



Node.js交互式解释器

在命令行界面上写代码 (https://github.com/webfansplz/vtui-repl)

Vue TermUI交互式文档

(https://github.com/vue-terminal/docs)

命令行贪吃蛇游戏

(https://github.com/webfansplz/temir-snake-game)

Thanks

Slidev