

换个角度看 Vue 组件

唐金州

校宝在线 技术专家

极客邦科技 会议推荐2019

5月

QCon 北京

全球软件开发大会

大会: 5月6-8日
培训: 5月9-10日

QCon 广州

全球软件开发大会

培训: 5月25-26日
大会: 5月27-28日

6月

GTLC
GLOBAL
TECH LEADERSHIP
CONFERENCE

上海

技术领导力峰会

时间: 6月14-15日

GMTC 北京

全球大前端技术大会

大会: 6月20-21日
培训: 6月22-23日

7月

ArchSummit 深圳

全球架构师峰会

大会: 7月12-13日
培训: 7月14-15日

10月

QCon 上海

全球软件开发大会

大会: 10月17-19日
培训: 10月20-21日

11月

GMTC 深圳

全球大前端技术大会

大会: 11月8-9日
培训: 11月10-11日

AiCon 北京

全球人工智能与机器学习大会

大会: 11月21-22日
培训: 11月23-24日

12月

ArchSummit 北京

全球架构师峰会

大会: 12月6-7日
培训: 12月8-9日

重学前端

每天10分钟，重构你的前端知识体系

你将获得

告别零散技术点，搭建前端知识体系

打通JS、HTML、CSS、浏览器4大脉络

40+前端重难点完全解答

大厂前端工程实战演练



作者：winter (程劭非)

前手机淘宝前端负责人



扫码立即参与

到手价 **¥69** ~~原价¥99~~ (仅限 **48** 小时)

参与拼团，结算时输入【GMTC用户专享优惠口令】：**2qianduan**

换个角度看 Vue 组件

唐金州

校宝在线 技术专家

自我介绍

Ant Design Vue 作者

极客时间 Vue 专栏讲师

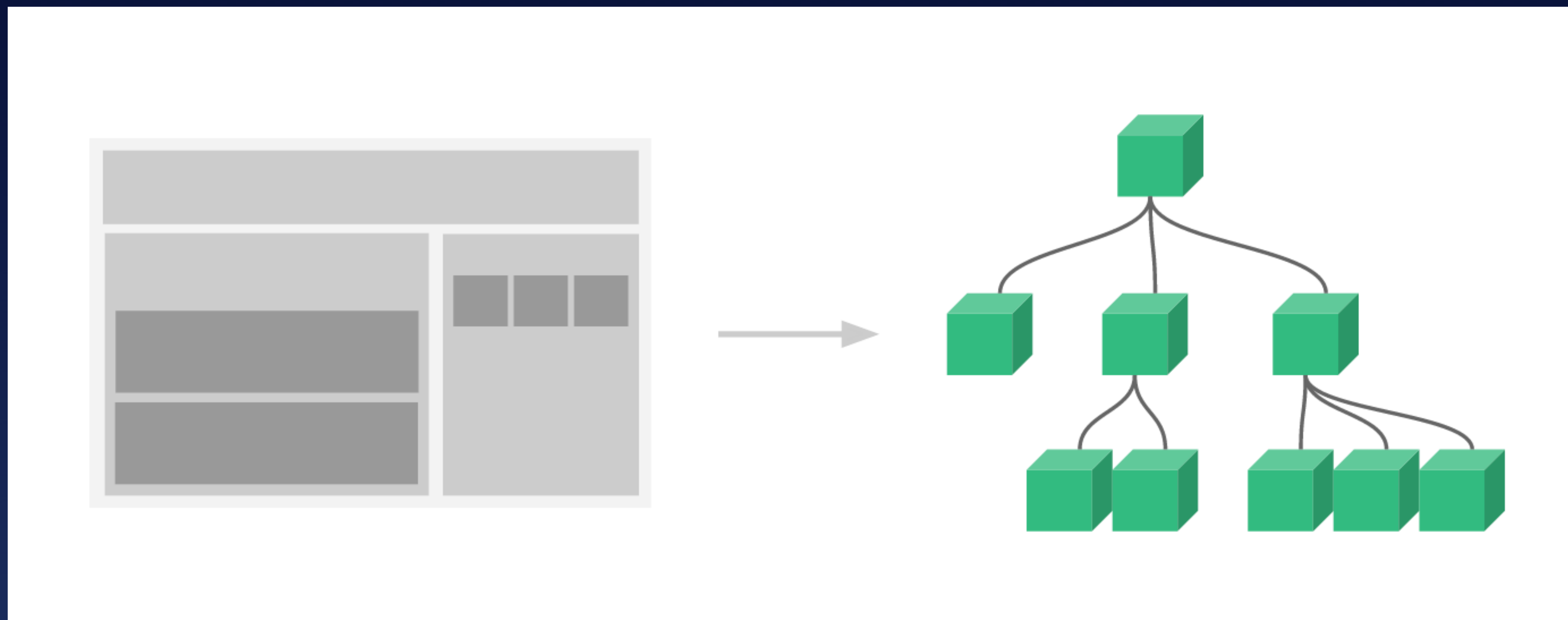
曾就职于 搜狗、蚂蚁金服、一点资讯

目录

- 1、什么是组件
- 2、又爱又恨的函数式组件
- 3、当插槽碰上函数式组件
- 4、指令的另外一种用途
- 5、简约而不简单的 Vue

什么是组件

什么是组件



什么是组件



什么是组件

```
function fn () {  
  .....  
  .....  
  .....  
  .....  
  成百上千行...  
}
```

```
function fn1 () {  
  .....  
}  
  
function fn2 () {  
  .....  
}  
  
function fn () {  
  fn1()  
  fn2()  
  fn1()  
}
```

什么是组件

```
<div>  
  <span>hello vue</span>  
  .....  
  成百上千个标签  
</div>
```

```
const custom1 = {  
  render () {  
    return <div>.....</div>  
  }  
}
```

```
const custom2 = { ..... }
```

```
<div>  
  <span>hello vue</span>  
  <custom1 />  
  <custom2 />  
  <custom1 />  
</div>
```

什么是组件

```
const custom1 = {  
  render () {  
    return <div>.....</div>  
  }  
}
```

```
const custom2 = { ..... }
```

```
<div>  
  <span>hello vue</span>  
  <custom1 />  
  <custom2 />  
  <custom1 />  
</div>
```

```
function custom1 (props) {  
  return createElement({  
    return <div>.....</div>  
  }, { prop })  
}
```

```
function custom2 () { return ... }
```

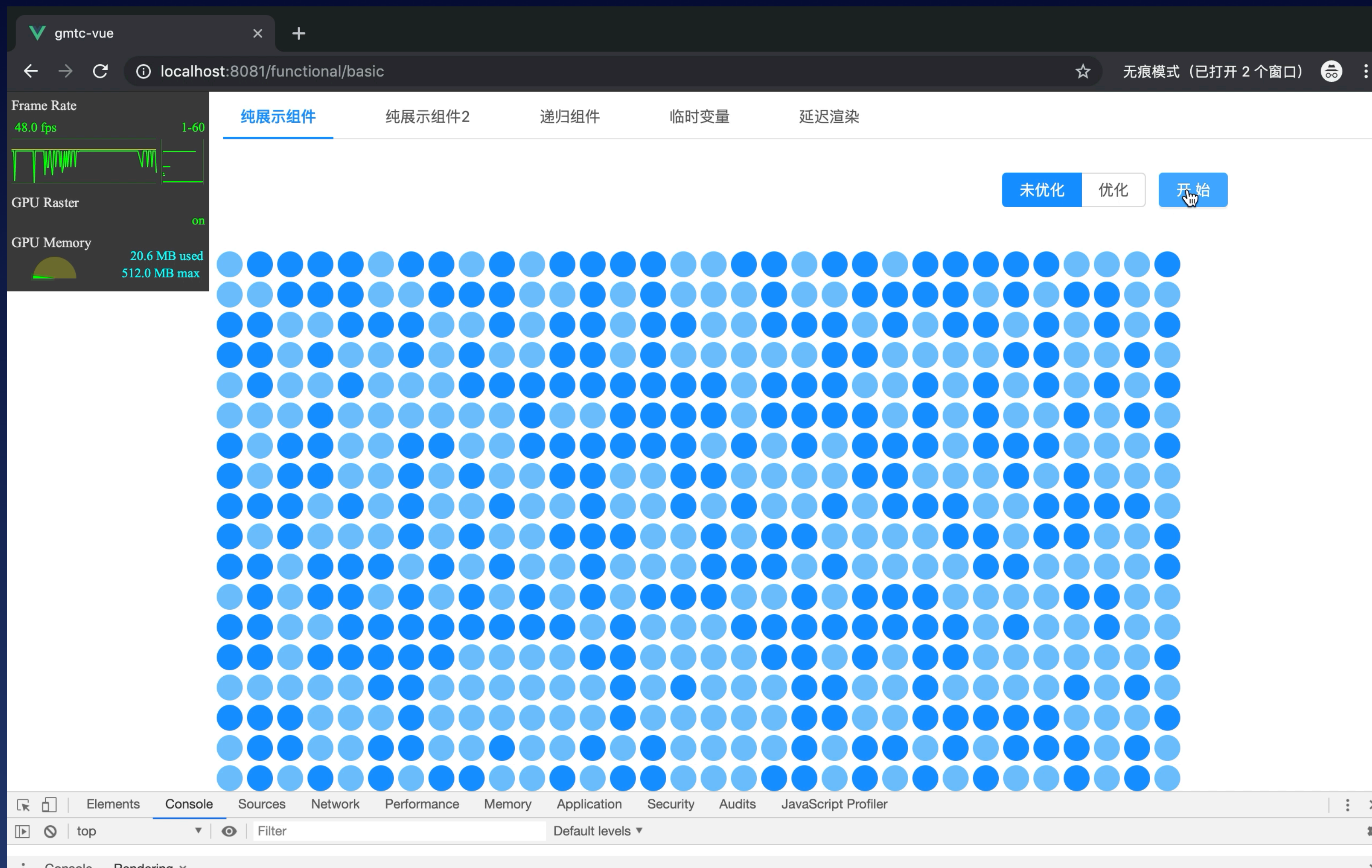
```
<div>  
  <span>hello vue</span>  
  {{ custom1(props) }}  
  {{ custom2() }}  
  {{ custom1() }}  
</div>
```

又爱又恨的函数式组件

函数式组件

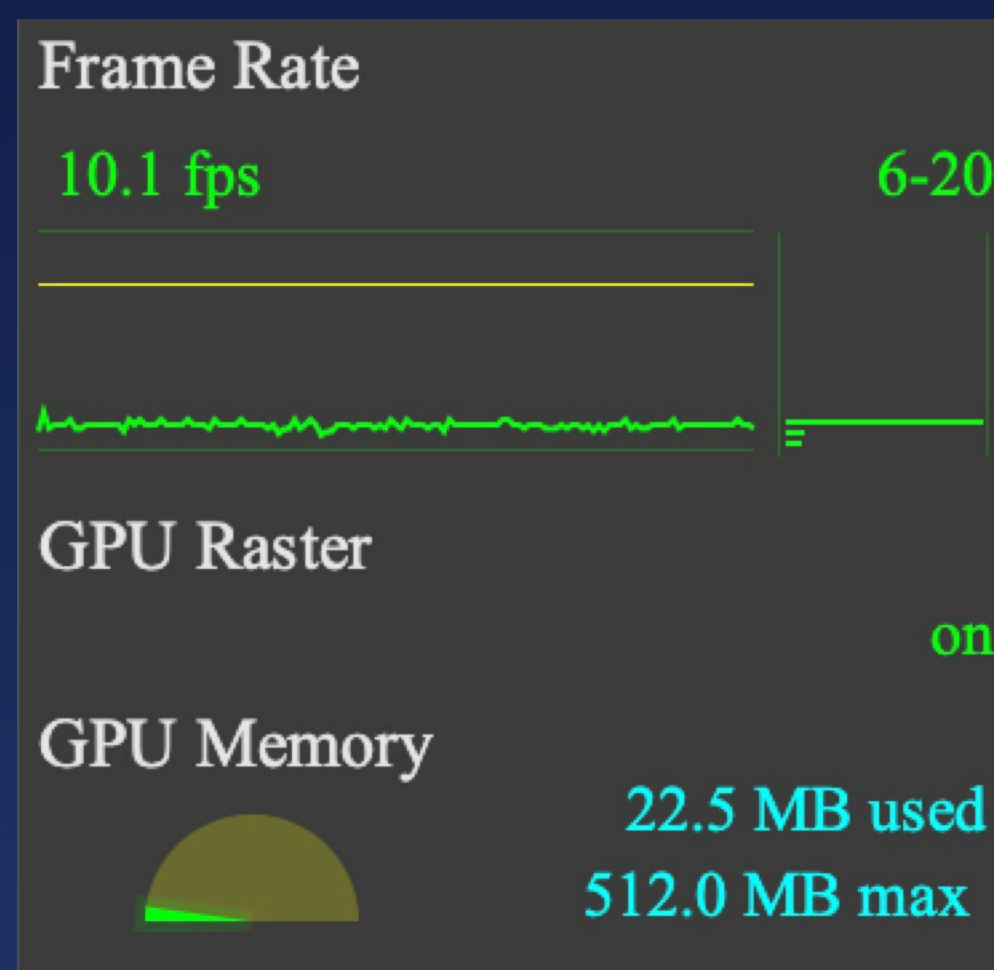
- 1、无状态(没有响应式数据)
- 2、无实例(没有this上下文)

函数式组件 - 纯展示组件

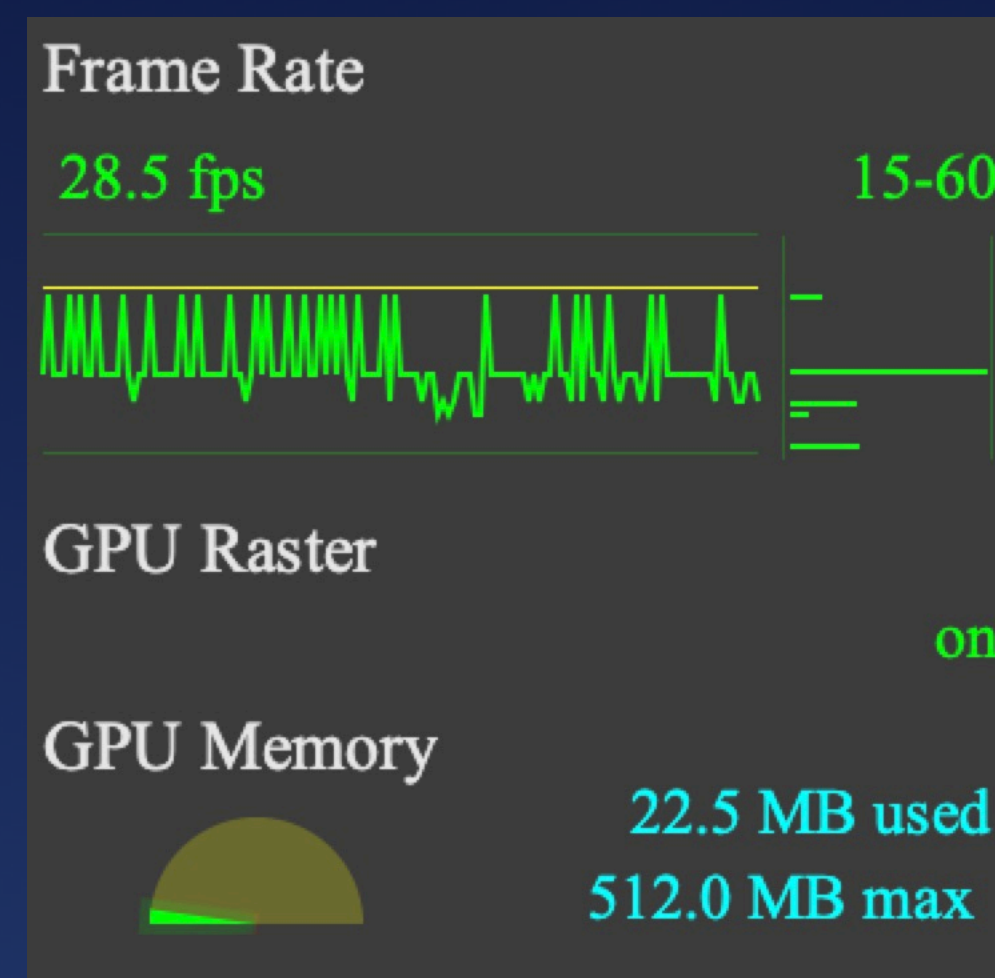


函数式组件 – 纯展示组件

```
<template>  
  <div>{{value}}</div>  
</template>
```

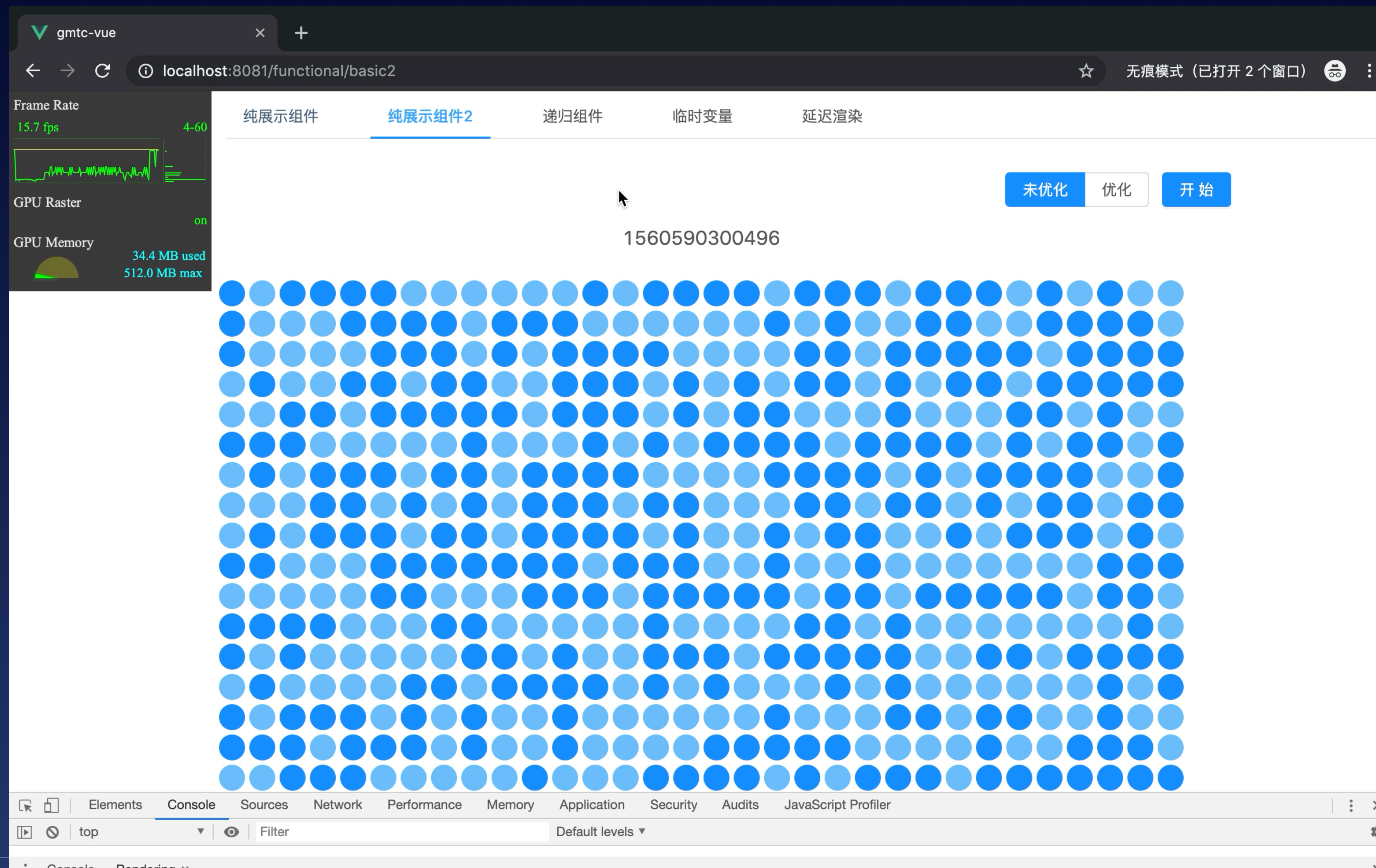


```
<template functional>  
  <div>{{props.value}}</div>  
</template>
```



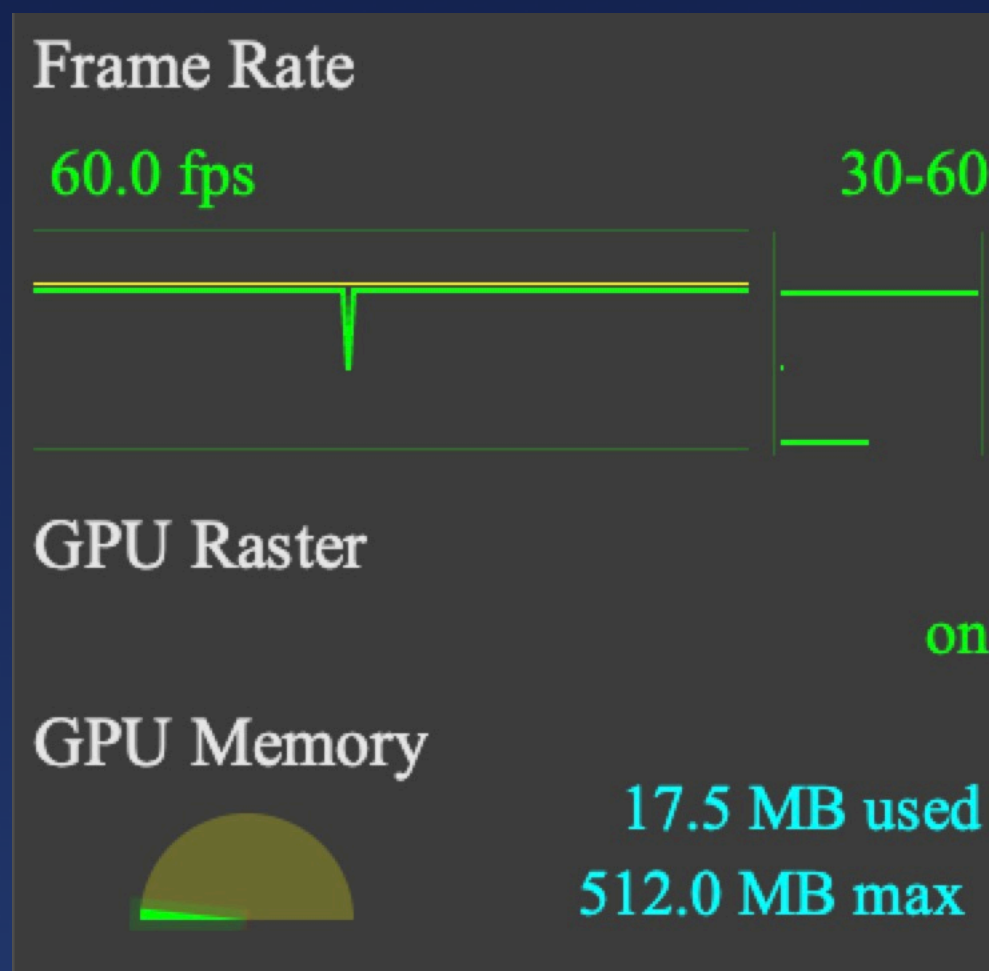
但是

妙用函数式组件 – 纯展示组件

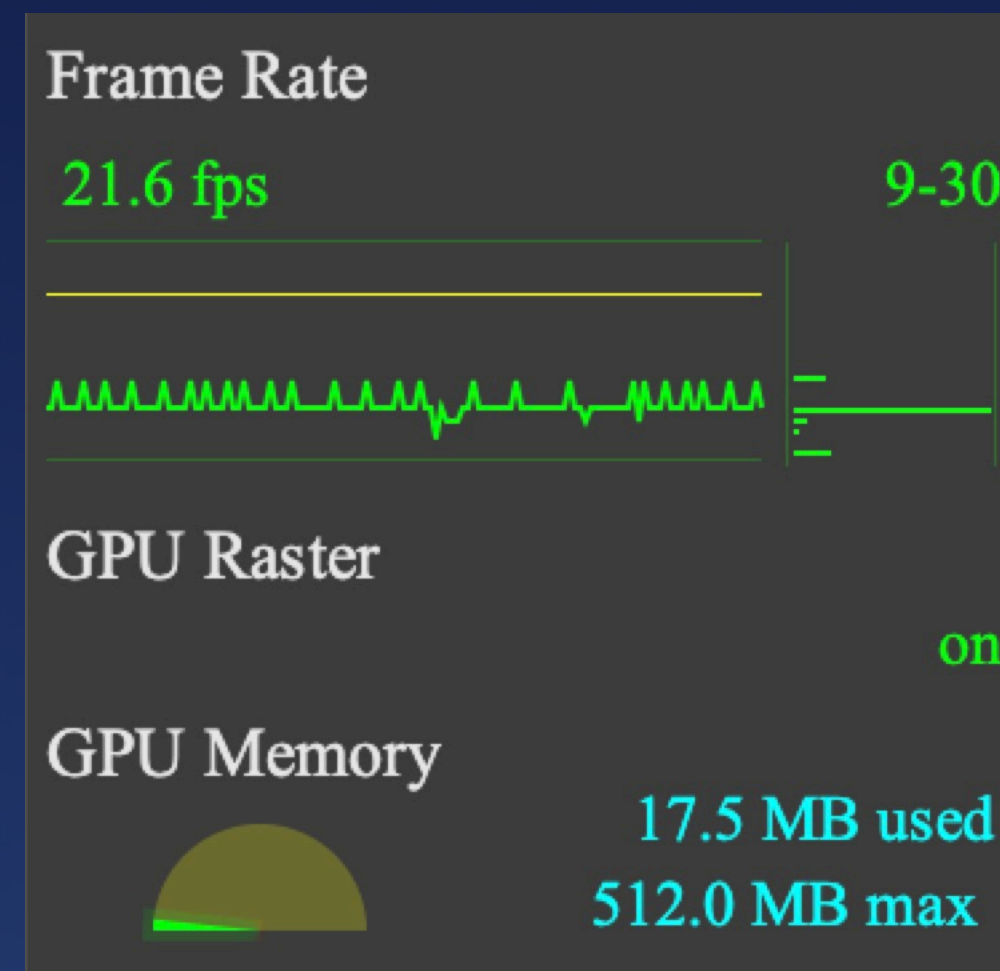


函数式组件 – 纯展示组件

```
<template>  
<div>  
  {{timer}}  
  <Component :value= "value" />  
</div>  
</template>
```



```
<template>  
<div>  
  {{timer}}  
  <FunctionalComponent  
    :value= "value"  
  />  
</div>  
</template>
```



函数式组件

```
function custom1 (props) {  
  return createElement(  
    return <div>.....</div>  
    }, { prop })  
}
```

```
function custom1 (props) {  
  return <div>.....</div>  
}
```

函数式组件

```
<div>  
  <span>hello vue</span>  
  .....  
  成百上千个标签  
</div>
```

```
function custom1 (props) {  
  return <div>.....</div>  
}  
  
function custom2 () { return ... }
```

```
<div>  
  <span>hello vue</span>  
  {{ custom1(props) }}  
  {{ custom2() }}  
  {{ custom1() }}  
</div>
```

函数式组件

```
diff  
div  
  span  
  custom1  
  custom2  
  custom1
```

```
diff  
div  
  span  
  div  
    ...  
  div  
    ...  
  div  
    ...
```

函数式组件 or 组件式函数

是“组件” 更是“函数”

(props) => VNodes

函数式组件

Vue的Template的图灵完备性

实际上Vue的组件系统具有图灵完备性

使用Vue计算阶乘

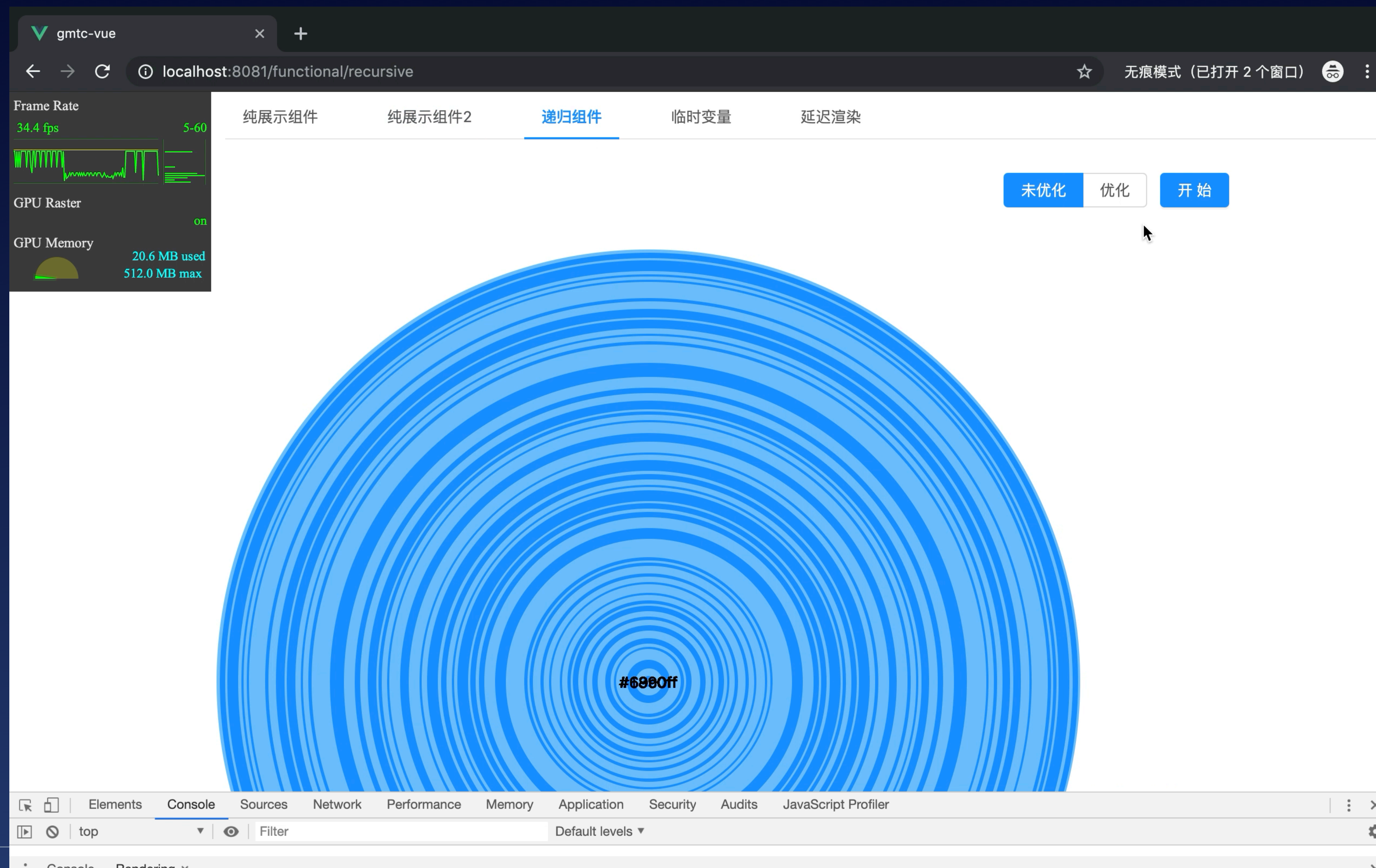
使用Vue计算斐波那契数列

-- winter 计算机之子 2018 vue conf

函数式组件

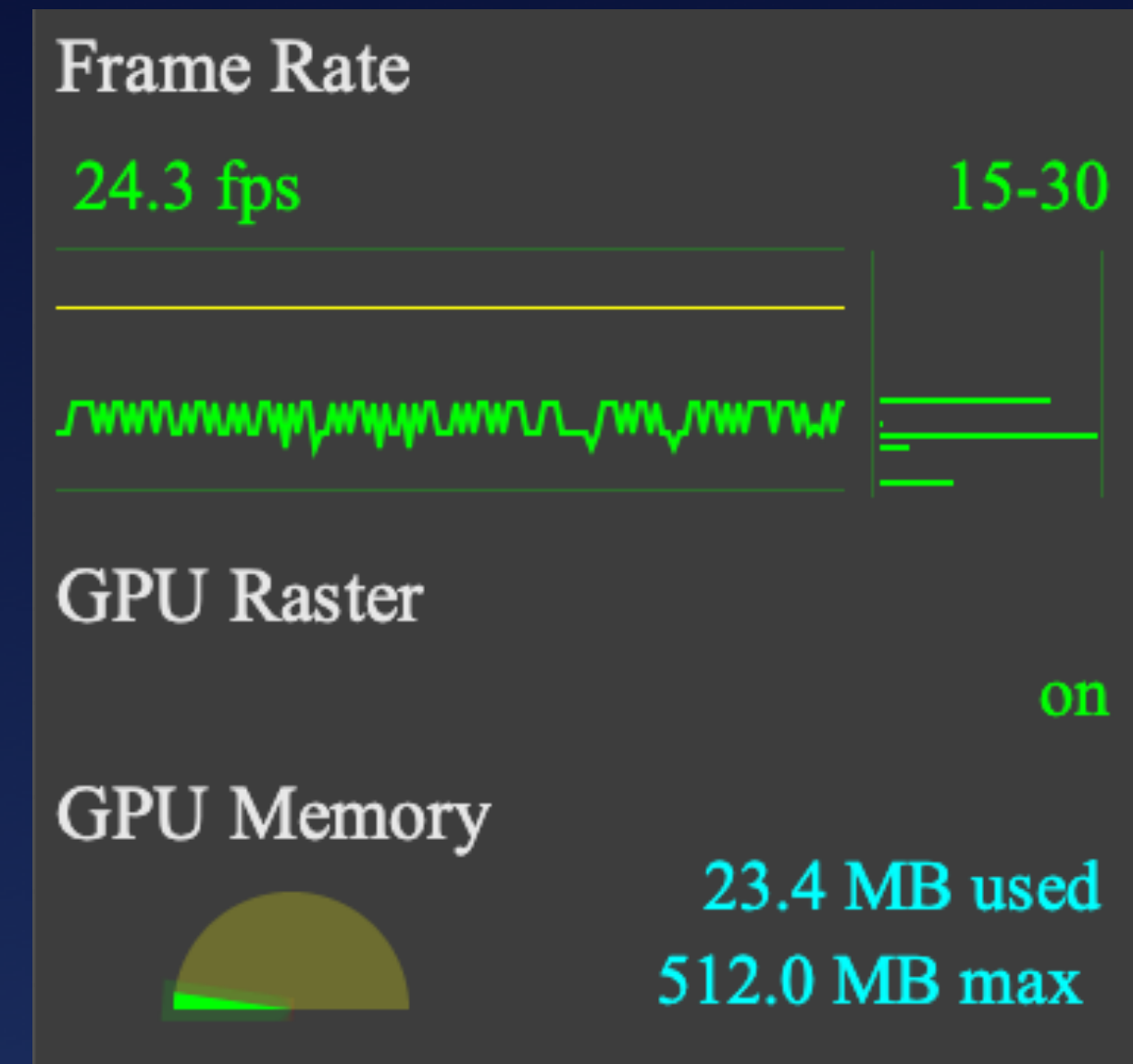
```
function fn (value) {  
  return (  
    <div>  
  
      ...  
      {value > 1 ? fn(value - 1) : ...}  
    </div>  
  )  
}
```

函数式组件 - 递归组件



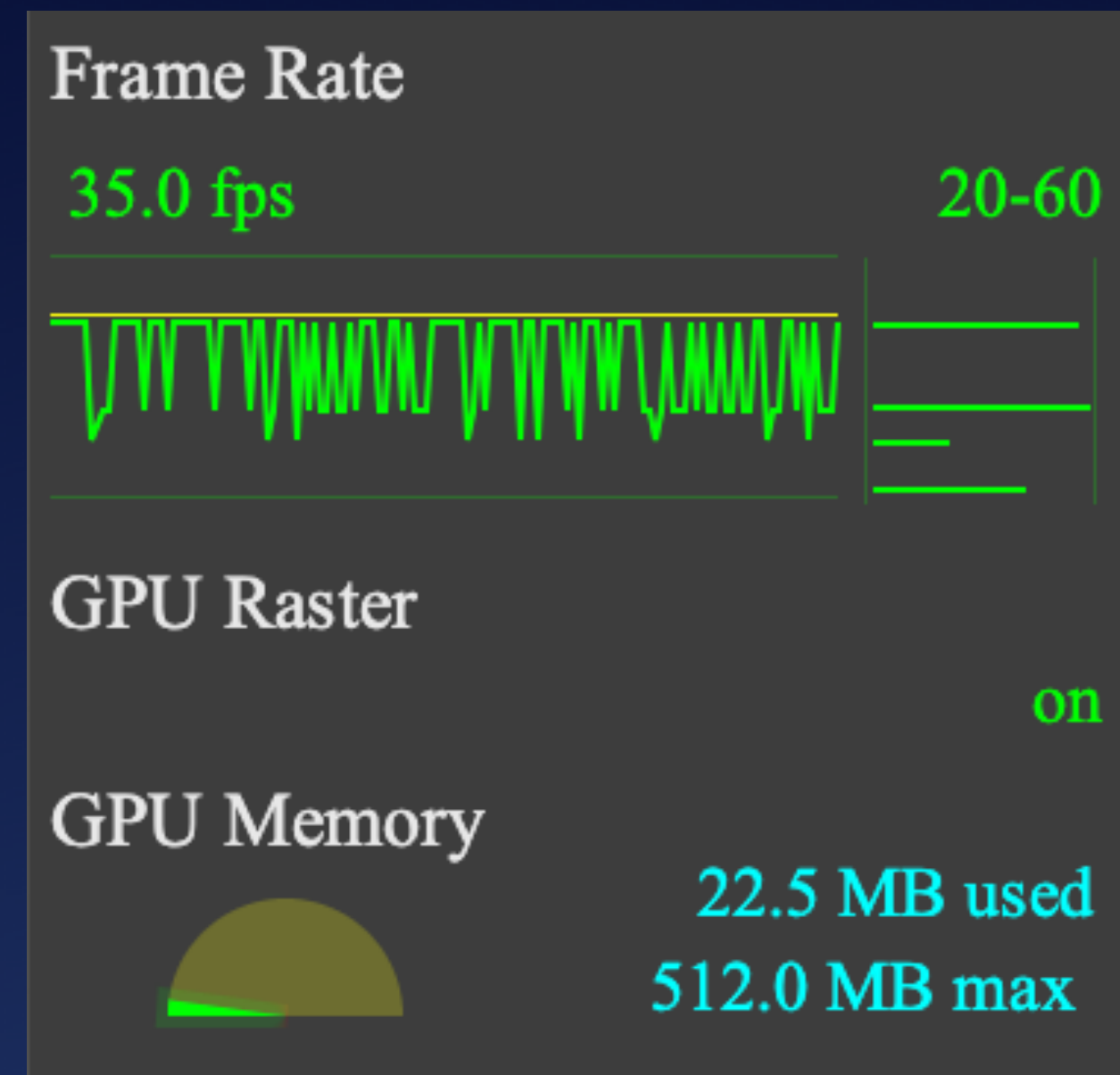
函数式组件 – 递归组件

```
<template>
  <div
    class="cell"
    :style="{
      width: `${value}px`,
      height: `${value}px`,
      backgroundColor: randomColor()
    }"
  >
    <span style="position: absolute">
      {{ randomColor() }}
    </span>
    <FunctionalOff v-if="value > 5" :value="value - 5" />
  </div>
</template>
```



函数式组件 – 递归组件

```
<template functional>
  <div
    class="cell"
    :style="{
      width: `${props.value}px`,
      height: `${props.value}px`,
      backgroundColor: parent.randomColor()
    }"
  >
    <span style="position: absolute">
      {{ parent.randomColor() }}
    </span>
    <Functional0n v-if="props.value > 5" :value="props.value - 5" />
  </div>
</template>
```



当插槽碰上函数式组件

插槽

```
<Com v-bind= "props" >  
  <template v-slot="slotProps" >  
    <div class="item" > {{ slotProps.item.text }} </div>  
  </template>  
</Com>
```

```
<slot v-bind= "slotProps" ></slot>
```

插槽

```
<Com v-bind= "props" >  
</Com>
```

```
props.getItem = (slotProps) =>  
<div class="item" > { slotProps.item.text } </div>
```

```
{{ getItem(slotProps) }}
```

插槽

```
<template>  
  <div>  
    <VNodes :vnodes= "getItem(slotProps)" />  
  </div>  
</template>
```

```
{  
  name: ' VNodes' ,  
  functional: true,  
  render: (h, ctx) => ctx.props.vnodes  
}
```


变量

```
<Com v-for="(n, index) of list" :key="index" :value="n">  
  {{ result(list.length) }}  
</Com>
```

变量优化

临时变量

计算属性

作用域插槽

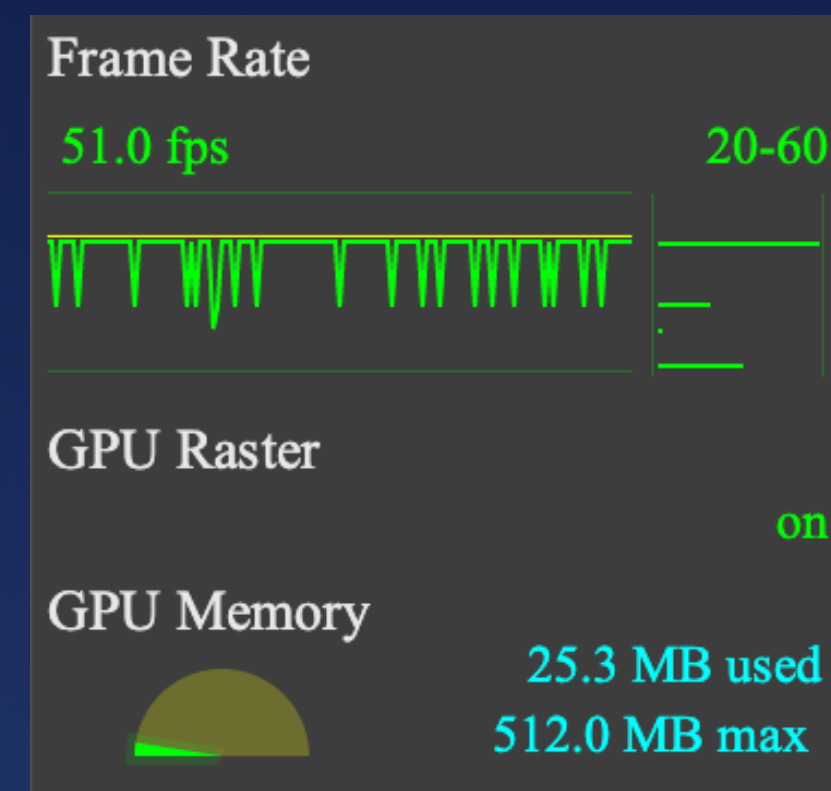
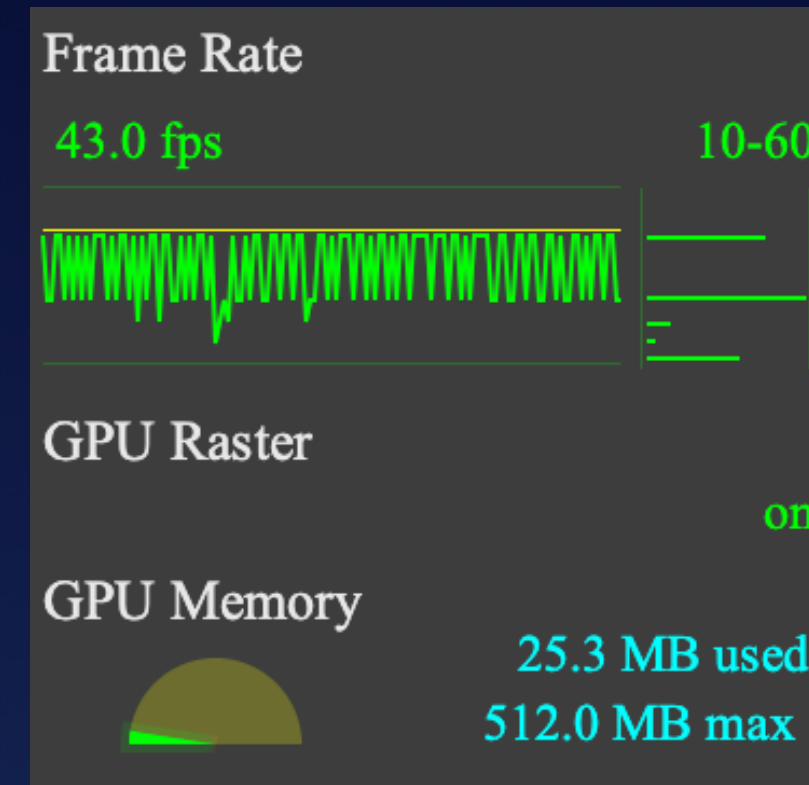
```
<Com v-bind= "prop" >  
  <template v-slot="slotProps">  
    <div class="item"> {{ slotProps.item.text }} </div>  
  </template>  
</Com>
```

```
function com (prop, slot) {  
  return slot(slotProps)  
}
```

妙用函数式组件 – 临时变量

```
<Com v-for="(n, index) of list" :key="index" :value="n">
  {{ result(list.length) }}
</Com>
```

```
<TempVar :res="result(list.length)">
  <template v-slot="{ res }">
    <Com v-for="(n, index) of list" :key="index" :value="n">
      {{ res }}
    </Com>
  </template>
</TempVar>
```



妙用函数式组件 - 临时变量

```
<script>
export default {
  functional: true,
  render: (h, ctx) => {
    return ctx.scopedSlots.default && ctx.scopedSlots.default(ctx.props || {})
  }
};
</script>
```

指令的另外一种用途

v-ref

v-decorator

ref

```
<!-- `vm.$refs.p` will be the DOM node -->
```

```
<p ref="p">hello</p>
```

```
<!-- `vm.$refs.child` will be the child component instance -->
```

```
<child-component ref="child"></child-component>
```

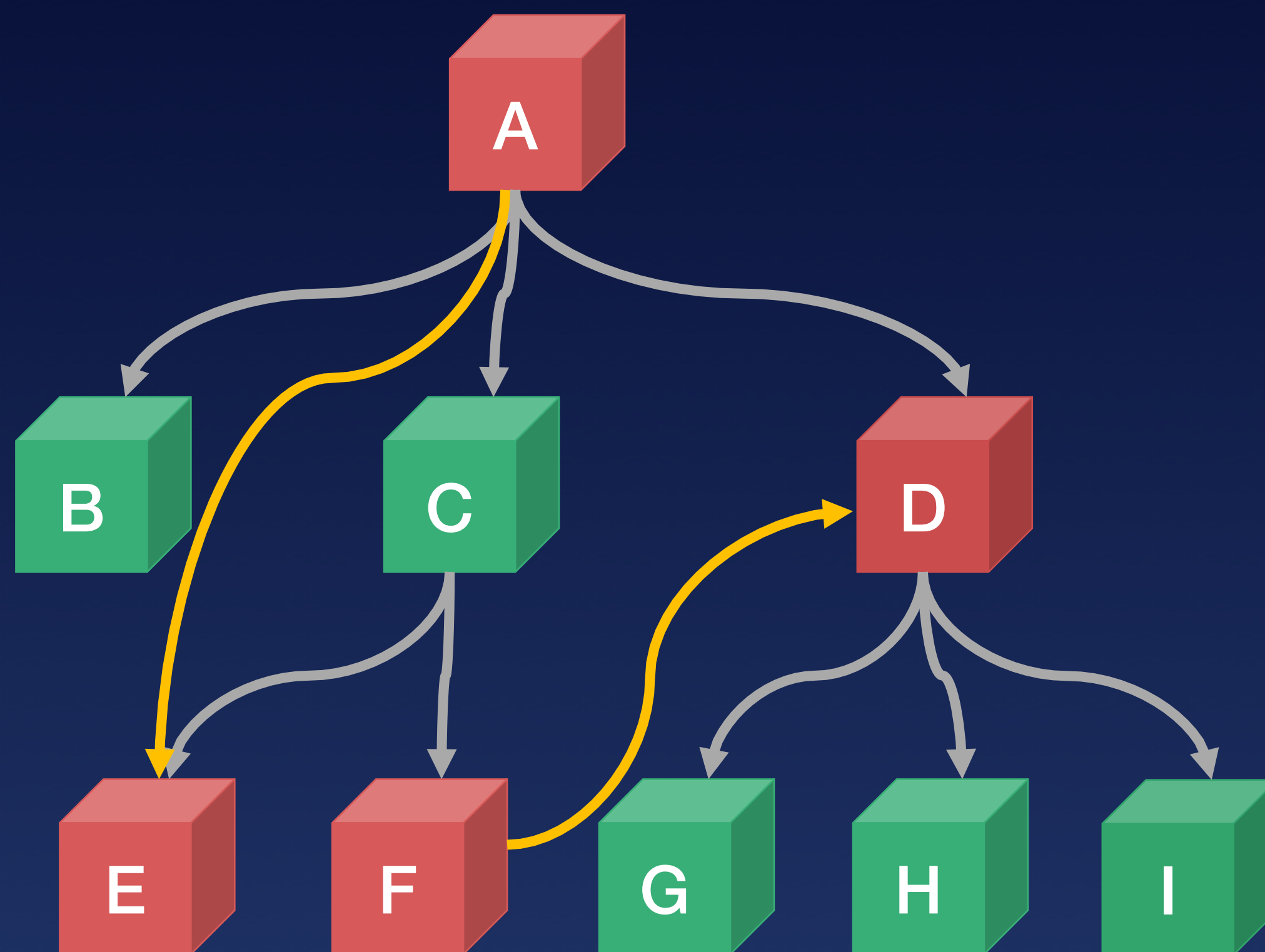
v-ref

```
<!-- vm.dom will be the DOM node -->  
<p v-ref="c => this.dom = c">hello</p>
```

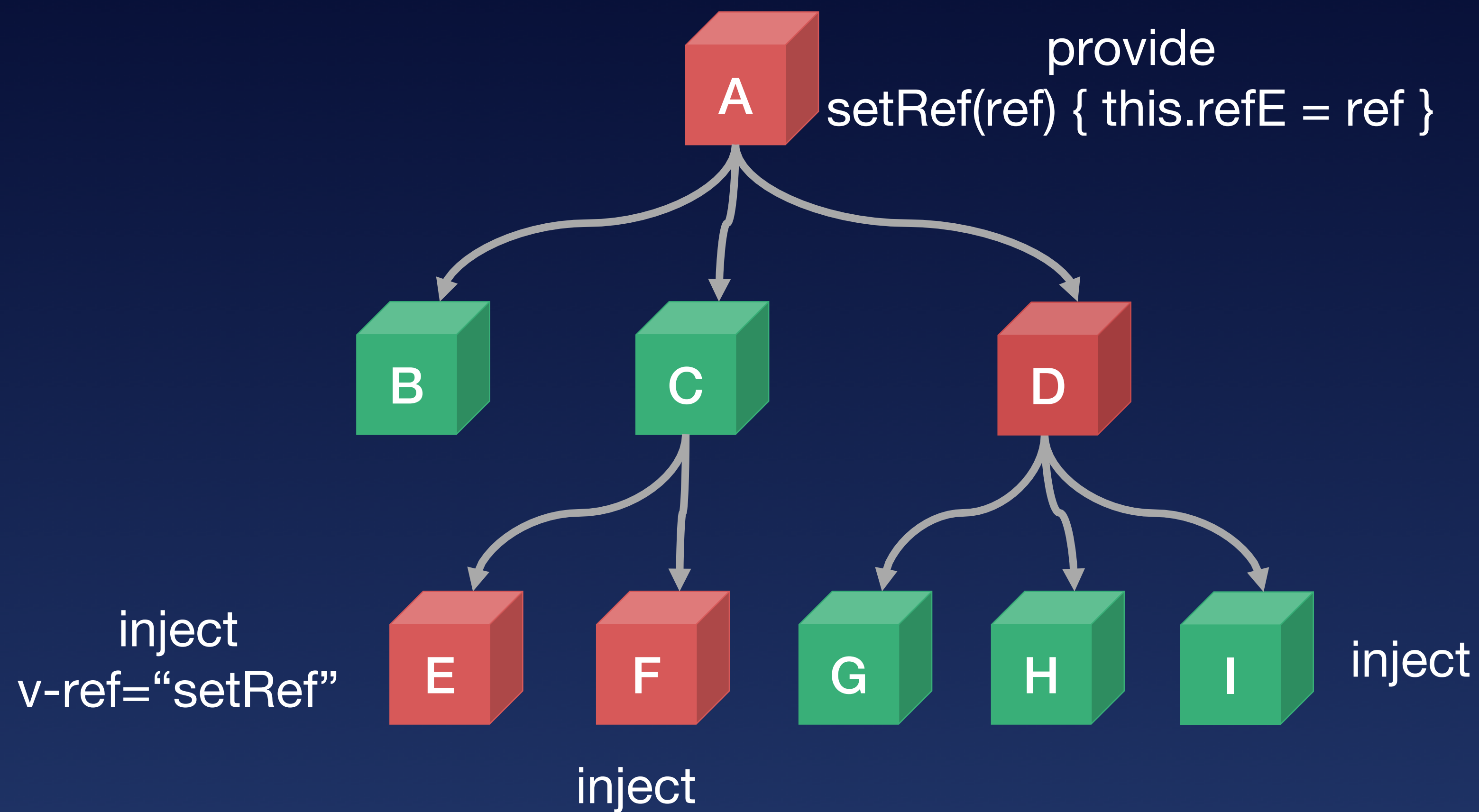
```
<!-- vm.child will be the child component instance -->  
<child-component v-ref="c => this.child = c"></child-component>
```

```
<span v-for="n in 10" :key="n" v-ref="(c, key) => {...}">{{ n }} </span>
```


组件通信 - 访问实例



组件通信 - 访问实例



v-decorator

Ant Design Vue form 表单进化史

- 1、 `getFieldDecorator('userName')(<a-input />)`
- 2、 `<a-form-control name= "userName" >
 <a-input />
</ a-form-control >`
- 3、 `<a-input v-decorator= "['userName']" />`

v-decorator

```
Vue.directive('decorator', {});
```

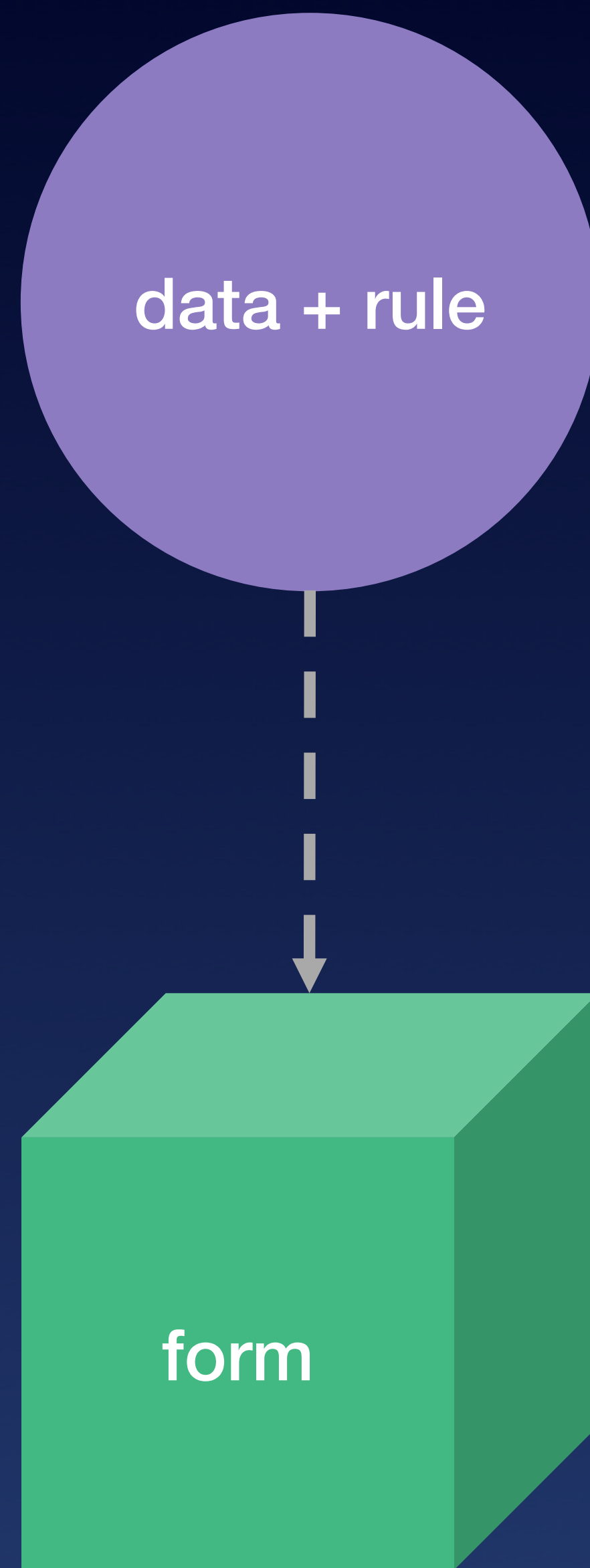
```
<a-form-item>
```

```
  <a-input v-decorator= "[ 'userName' ]" />
```

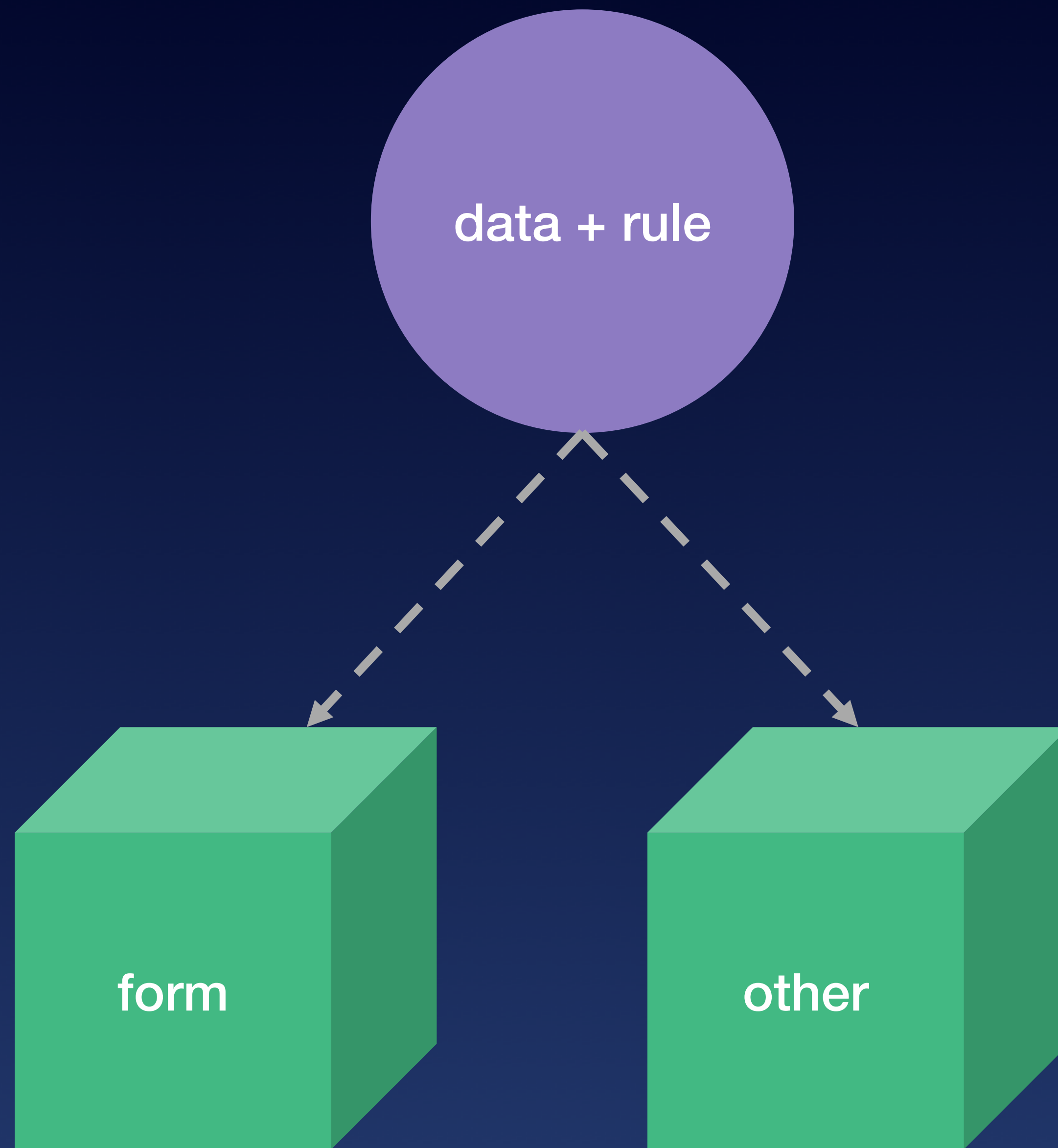
```
</a-form-item>
```



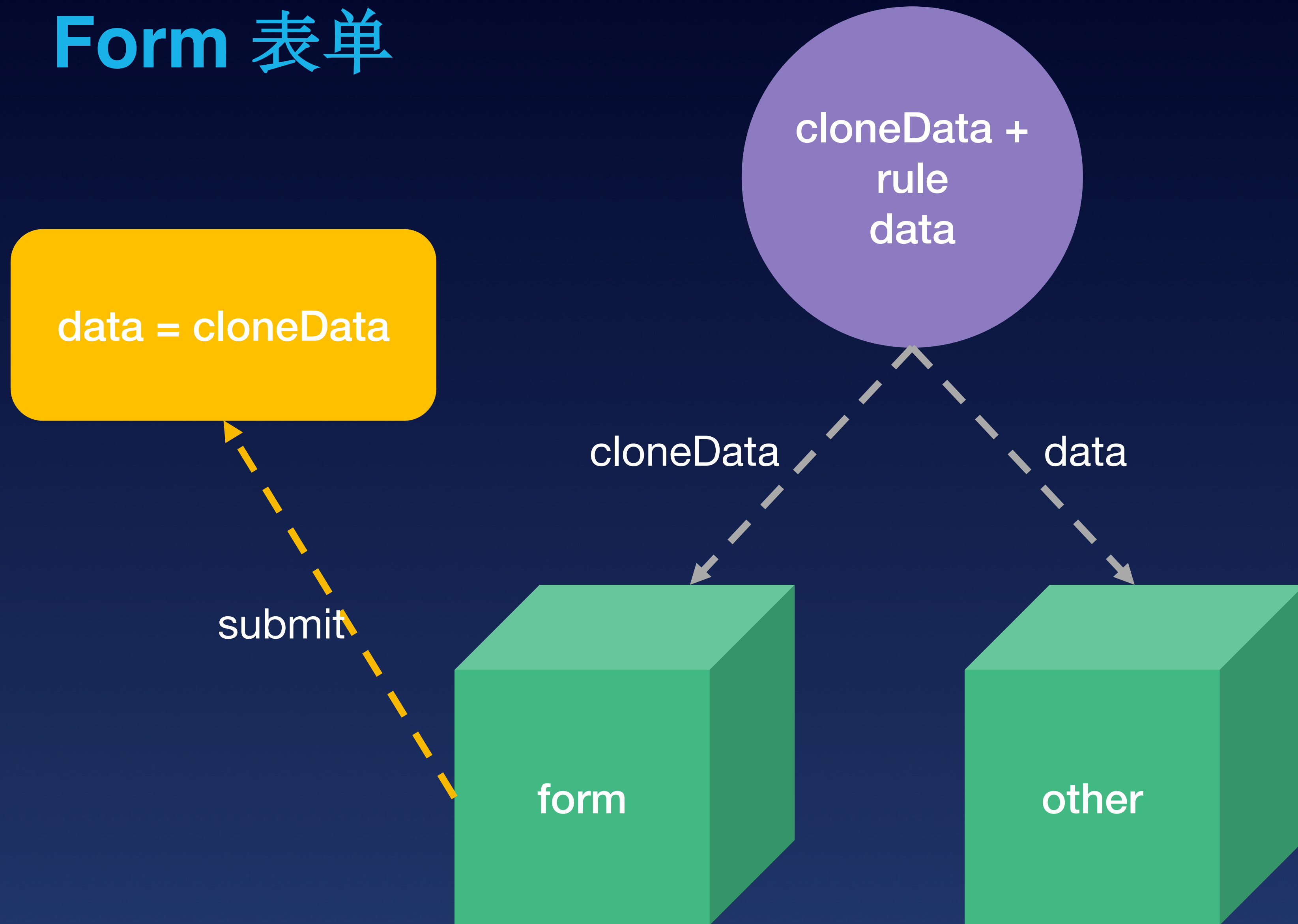
Form 表单



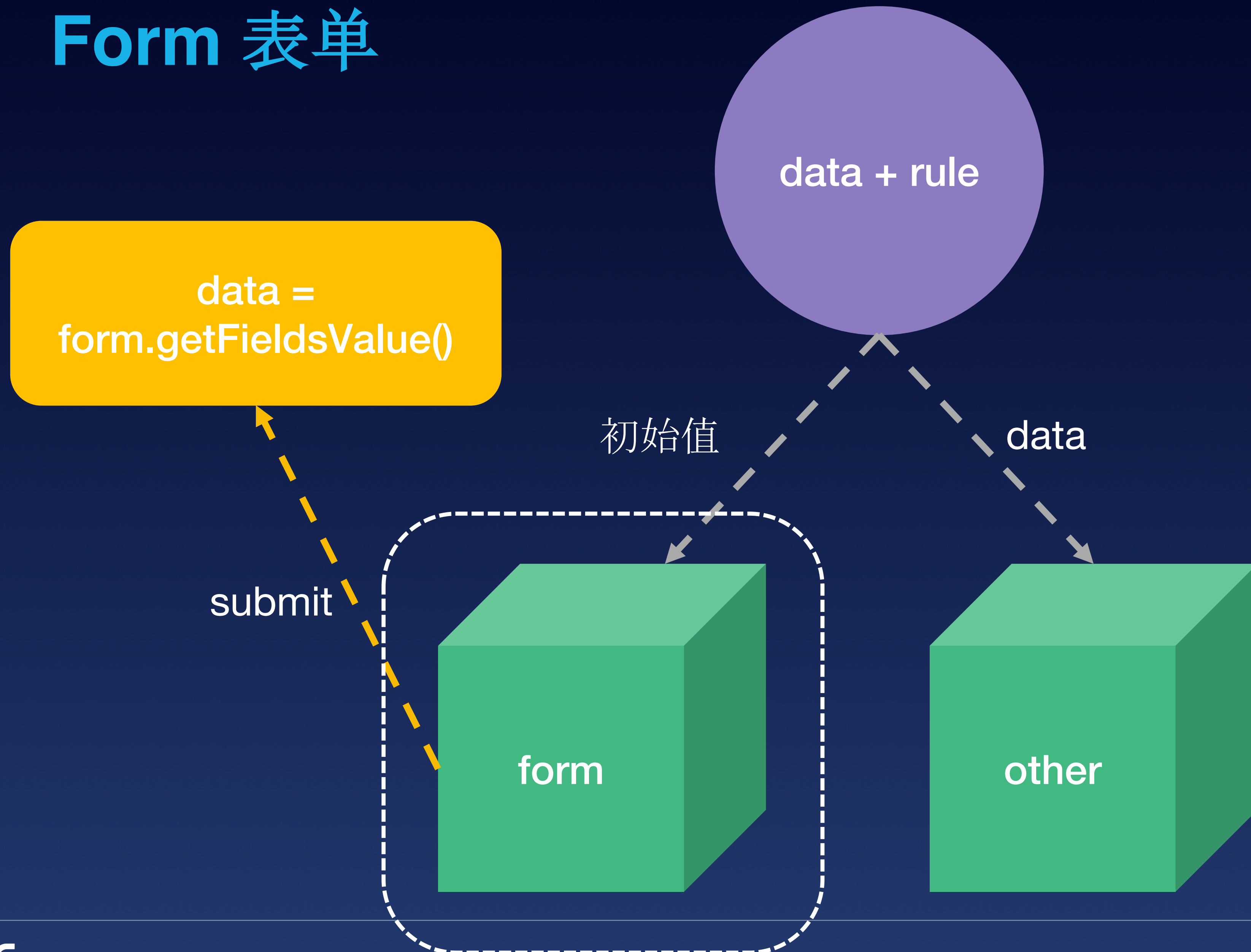
Form 表单



Form 表单



Form 表单



但是

Form 表单

即将支持双向绑定的表单收集校验

敬请期待!!!

总结 – 简约而不简单的 Vue

1、Vue 简单？

2、Vue 灵活度低？

3、不要被固有的概念，限制了我们的思维

InfoQ官网 全新改版上线

促进软件开发领域知识与创新的传播



关注InfoQ网站
第一时间浏览原创IT新闻资讯



免费下载迷你书
阅读一线开发者的技术干货

TGO 鲲鹏会

汇聚全球科技领导者的高端社群

🏠 全球12大城市

👤 850+ 高端科技领导者

使命

Mission

为社会输送更多优秀的
科技领导者

愿景

Vision

构建全球领先的有技术背景
优秀人才的学习成长平台



扫描二维码，了解更多内容

视频课程 

Vue 开发实战

从 0 开始搭建大型 Vue 项目

输入**优惠口令**：MYLOVEVUE

到手价 **¥89**，立省 ¥40

~~原价 ¥129~~



扫码立即参团