1. vue3为什么要用proxy替换defineProperty —— 响应式优化?

a. defineProperty 局限性最大的愿意是它只能针对对象属性监听。 vue2中需要对data的属性做遍历，及递归遍历，为每个属性设置getter和setter 因此只能对data中预定义过的属性作出响应。 所以新增的属性无法监听。数组方法无法监听。

b. proxy是针对对象进行拦截的，可以直接代理对于这个对象的操作。外界对这个对象的访问，都必须先经过这层拦截。 data属性不需要都提前预定义 解决了数组方法。

c. 响应是惰性的

- vue2中，对于一个深层属性嵌套的对象，要劫持它内部层次的变化，就需要递归遍历这个对象
- vue3中，使用proxy不能监测到对象内部深层次的属性变化，因此它的处理方式是在getter中去递归响应式，这样在真正访问的内部属性才会变成响应式，可以实现按需实现响应式，减少性能消耗。

2. vue3编译做了哪些优化?

a. 标记动态节点，dom diff只针对变化的节点 b. slot编译优化

3. setup

- 执行时机是在 beforeCreate 之前执行
- 参数 prop context($attr,slot,$emit等)

4. reactive(引用类型)、ref（基础和引用）、toRefs(解构)、toRef

5. 生命周期钩子

6.watch和watchEffect用法

watch函数侦听特点的数据源，并在回调函数中执行副作用。默认是惰性的，只有侦听到源数据变更才执行回调。 watch(source, callback, [options]) // 修改age值时会触发 watch的回调 watch(() => state.age, (curAge, preAge) => { console.log("新值:", curAge, "老值:", preAge); }); watchEffect setup() { const state = reactive({ nickname: "xiaofan", age: 20 }); let year = ref(0)

```
setInterval(() =>{
    state.age++
    year.value++
},1000)
```

```
watchEffect(() => {
    console.log(state);
    console.log(year);
  }
);

return {
    ...toRefs(state)
}
```

}, watchEffect不需要手动传入依赖，会自动手机依赖，只需要指定一个回调函数，在组件初始化时，会先执行一次来收集依赖，然后当收集到的数据发生变化时，就会再次执行回调函数。无法获取到变化前的值。

v-model改变

- vue2 对一个prop双向绑定

1. v-model 相当于value @input 可以定义model选项自定义 model: { prop: 'title', event: 'change' }, 相当于 <ChildComponent :title="pageTitle" @change="pageTitle = $event" />
2. 使用v-bind.sync 使用update:myPropName抛出事件。 this.$emit('update:title', newValue) 父组件监听该事件，并更新本地data <ChildComponent :title="pageTitle" @update:title="pageTitle = $event" /> 为了方便起见，可以用.sync修饰符来缩写：

- vue3.0 在3.x中，自定义组件上的v-model相当于传递了modelValue prop并接收抛出的update:modelValue事件 相当于 <ChildComponent :modelValue="pageTitle" @update:modelValue="pageTitle = $event" />

若需要修改model名称，作为组件内model选项的替代，现在我们可以将一个argument传递给model：<ChildComponent :title="pageTitle" @update:title="pageTitle = $event" />

这也可以作为.sync修饰符的替代，而且允许我们在自定义组件上使用多个v-model

<ChildComponent :title="pageTitle" @update:title="pageTitle = $event" :content="pageContent" @update:content="pageContent = $event" />

除了像 .trim 这样的 2.x 硬编码的 v-model 修饰符外，现在 3.x 还支持自定义修饰符：

异步组件

要让 TypeScript 正确推断 Vue 组件选项中的类型，需要使用 defineComponent