1、Vue 3.0 性能提升主要是通过哪几方面体现的？

* l  Tree-shaking 支持
* l  Composition API
* l  Fragment、Teleport、Suspense
* l  更好的 TS 支持
* l  自定义渲染API
* l  编译时对VDom的性能优化

2、Vue 3.0 所采用的 Composition Api 与 Vue 2.x使用的Options Api 有什么区别？

1>在vue2中如何组织代码的，我们会在一个vue文件中methods，computed，watch，data中等等定义属性和方法，共同处理页面逻辑，我们称这种方式为Options API。**缺点：** 一个功能往往需要在不同的vue配置项中定义属性和方法，比较分散，项目小还好，清晰明了，但是项目大了后，一个methods中可能包含20多个方法，**你往往分不清哪个方法对应着哪个功能。**

**2>在vue3 Composition API 中**，我们的代码是**根据逻辑功能来组织的**，一个功能所定义的所有api会放在一起（更加的高内聚，低耦合），这样做，即时项目很大，功能很多，我们都能快速的定位到这个功能所用到的所有API，而不像vue2 Options API 中一个功能所用到的API都是分散的，需要改动功能，到处找API的过程是很费劲的

3>为什么要使用 Composition API：

Composition API 是根据逻辑相关性组织代码的，提高可读性和可维护性

基于函数组合的 API 更好的重用逻辑代码（在vue2 Options API中通过Mixins重用逻辑代码，容易发生命名冲突且关系不清）

3、Proxy 相对于 Object.defineProperty 有哪些优点？

Proxy 的优势如下：

* Proxy 可以直接监听数组的变化
* Proxy 可以直接监听对象而非属性
* Proxy 有 13 种拦截方法，比 Object.defineProperty 要更加丰富的多

Object.defineProperty 的优势如下：

·

兼容性好，支持 IE9，而 Proxy 的存在浏览器兼容性问题,而且无法用 polyfill 磨平，因此 Vue 的作者才声明需要等到下个大版本( 3.0 )才能用 Proxy 重写。

 Object.defineProperty

(obj, prop, descriptor) 的问题主要有三个：

无法监听数组的变化

必须遍历对象的每个属性

必须深层遍历嵌套的对象

4、Vue 3.0 在编译方面有哪些优化？

通过在编译阶段优化编译的结果，实现运行时patch过程的优化

通过编译阶段对静态模版的分析，编译生成了block tree

block tree是一个将模版基于动态节点指令切割的嵌套区块，每个区块内部的节点结构是固定的，每个区块只需要以一个Array来追踪自身包含的动态节点

借助block tree,vue.js将vnode更新性能由与模版整体大小相关提升为与动态内容的数量相关

5、Vue.js 3.0 响应式系统的实现原理？

Vue3 使用 Proxy 对象重写响应式系统，这个系统主要有以下几个函数来组合完成的：

1、reactive:

接收一个参数，判断这参数是否是对象。不是对象则直接返回这个参数，不做响应式处理

创建拦截器对象 handler, 设置 get/set/deleteProperty

get

收集依赖（track）

返回当前 key 的值。

如果当前 key 的值是对象，则为当前 key 的对象创建拦截器 handler, 设置 get/set/deleteProperty

如果当前的 key 的值不是对象，则返回当前 key 的值

set

设置的新值和老值不相等时，更新为新值，并触发更新（trigger）

deleteProperty

·

当前对象有这个 key 的时候，删除这个 key 并触发更新（trigger）

返回 Proxy 对象

2、effect: 接收一个函数作为参数。作用是：访问响应式对象属性时去收集依赖

3、track:

接收两个参数：target 和 key

如果没有 activeEffect，则说明没有创建 effect 依赖

如果有 activeEffect，则去判断 WeakMap 集合中是否有 target 属性，

WeakMap 集合中没有 target 属性，则 set(target, (depsMap = new Map()))

WeakMap 集合中有 target 属性，则判断 target 属性的 map 值的 depsMap 中是否有 key 属性

depsMap 中没有 key 属性，则 set(key, (dep = new Set()))

depsMap 中有 key 属性，则添加这个 activeEffect

4、trigger:

判断 WeakMap 中是否有 target 属性

WeakMap 中没有 target 属性，则没有 target 相应的依赖

WeakMap 中有 target 属性，则判断 target 属性的 map 值中是否有 key 属性，有的话循环触发收集的 effect()