1. Nuxt常用方法（服务端方法都可以接收到context对象，分清楚服务端方法和客户端vue的方法）：layout：设置布局文件，head：设置文件头部属性（meta）
2. Nuxt：2.1.3版本后的nuxt配置component为true可以不需要显式导出导入组件即可使用
3. Nuxt根据pages目录自动生成路由
4. 手动修改配置和页面结构等等
5. React和vue都是把组件挂载到某个节点上的，只是项目中的根HTML（模板）的body标签只有一个div，也就是整个页面只显示一个组件，所以组件的页面结构就是页面结构。也可以在模板HTML文件中修改页面结构。组件化开发一般不修改模板HTML文件
6. React、vue等JavaScript框架中函数传参时都可以使用解构赋值解构出来参数的某个属性
7. Nuxt的服务端方法都可以接收context对象作为参数，自定义方法（vue method中的方法）运行在客户端，不能接受context对象
8. Vue本身是客户端渲染
9. 不要固定思维，框架是灵活变化的，但是思想不会变，知道error等其它页面在哪定义的就行
10. Nuxt的pages等文件夹下的vue文件会被打包到.nuxt目录下，layout下的布局文件本质上是从打包后的文件中导入的，所以error文件也会使用布局文件
11. ！！！！！Nuxt的根标签是HTML文件中的标签，布局文件只是把组件统一管理，类似react中的App组件
12. ！！！！！Nuxt流程：pages、components等文件夹下的文件写页面，webpack打包到.nuxt目录下，再集中到layout中，不同的layout使用时挂载到根HTML文件的标签上。只是自己写的页面不需要导出导入和配置路由了（新版本和旧版本会有一些差异，但思想不会变化）
13. ！！！！！服务端渲染的根HTML文件的根标签上挂载的组件可以变化，拼接成HTML文件后发给前端，所以页面的title可以变化。客户端渲染不行
14. Nuxt没有layouts文件夹就自己加上
15. Nuxt要注意区分服务端钩子函数和客户端钩子函数。服务端钩子函数可以接收context对象，其中的app属性是服务端对象。客户端钩子函数直接使用this获取客户端vue实例
16. AsyncData在组件加载之前调用，可以合并data的数据
17. Vue使用第三方库需要绑定到实例对象中，作为插件，使用this调用，nuxt同理，通过$调用
18. Async虽然返回一个promise对象，但是也可以当作函数用，主要是用await关键字等待异步操作的结果
19. 拦截器就是一种钩子函数
20. 服务端渲染和客户端渲染的区别应该是路由在哪处理。客户端渲染是把路由管理交给前端，服务端渲染是在服务端处理路由
21. Vue使用插件绑定时可以用vue.use等方式，不需要配置插件路径。Nuxt引用第三方插件时只比vue多了一步配置插件路径。自定义插件时不用vue.use
22. 使用第三方插件配置插件文件即可，自定义插件配置是哪一端的插件，分别用于不同端的方法中
23. Vue组件的script和style是全局有效的。
24. Es6定义对象新方式：只写变量名，键名默认为变量名，值是变量的值
25. 使用第三方UI库，类名最好有独特标识，以免和库的类名重复
26. ！！！！！重要内容：使用elementUI等第三方UI组件库，不要只在组件添加类名，要看浏览器渲染之后的标签是什么样的，有些组件内置了类名，有些转换为了其它标签，基本都有类名
27. Nuxt不需要手动配置store
28. Redux和vuex区别点：vuex把state独立于mutation，而redux把state分散在reducer中
29. Nuxt注意在客户端方法中使用客户端对象，在服务端方法中使用服务端对象
30. ！！！！！nuxt使用vuex虽然方便，但是要求必须把mutations和actions名称为mutations和actions不能改动，除非自己定义store对象
31. Nuxt服务端渲染，服务端的方法意思是在拼接成html文档前调用，即vue实例对象生成前调用，所以获取不到this对象（vue实例）
32. Nuxtserverinit是每次发送请求时触发，在生成vue实例前触发
33. ！！！！！cookie只是存到了浏览器中，在一个浏览器登录后，另一个浏览器不是登录状态
34. Nuxt使用子路由，父级路由名称不显示？