# 网易篇——

**1、组件间通信的分类可以分成哪些？**

父组件传子组件，子组件使用 poros 进行接受
子组件传父组件，子组件使用$emit+事件对父组件传值
组件中使用$parent 和$children 获取到父组件实例和子组件实例，进而获取数据
使用$refs 获取组件实例，进而获取数据
使用 Vuex 进行状态管理
使用 eventBus 进行跨组件触发事件，进而传递数据
使用浏览器本地储存，例如 localStorage

**2、Vue 为什么在 HTML 中监听事件？**

首先要明确一点，VUE 并不是在 HTML 中监听事件，VUE 底层是通过动态注册事件来实现事件监听的。VUE 只不过是在 HTML 中记录每个节点都绑定了哪些具体的事件。
因为 Vue 会动态的插入 dom 节点，为了在增加 dom 节点时，仍然保留 dom 节点上绑定的事件，所以 VUE 需要记录节点上绑过的每个具体事件。在节点新增成功后，再把事件一一的绑定到对应的节点上。
VUE 是基于模板语言来实现数据的双向绑定，根据其设计，通过把事件记录在 HTML 中无疑是一个比较好的解决方案。

**3、JS 中的 Array.splice() 和 Array.slice() 方法有什么区别？**

splice（）返回符合条件的新数组，且会改变原数组；slice（）返回符合条件的数组，并不会改变原数组

**4、web 常见的攻击方式有哪些？如何防御？**

xss 和 csrf，这两个与前端有关，也比较常见；
具体基本上都可以百度到，我这边就当温故而知新了；
xss（跨域脚本攻击）（这里也仅针对 script 的攻击，当然实际上远远不止）

主要利用了浏览器会自动加载 script 标签内的代码而产生的攻击；攻击范围的话，js 代码能做到的，基本上都能实现，比如用 document.cookie 获取 cookie 信息；

一般直接分为两大类：反射型、储存型；（至于 dom 那个，基本上也属于反射型，也没必要太过于较真再去细分）
所谓的反射型 xss，一般有两种攻击方式：
①通过 url 上的参数，在参数上直接添加脚本代码，这样服务器可以不需要处理，直接反射到浏览器，浏览器直接执行；（这种信任链接也能实现，只要服务器没有做对应处理）
②通过 url 上的参数，让服务器根据参数返回特定的 script 脚本代码；（这种需要陌生链接才能实现）

储存型，其实就是将恶意的 script 脚本给保存进数据库了，比如保存一篇文章、一条评论；这种一般在有用户输入功能的网站可以做到；
如何防范：服务器要对 url 参数、保存文本内容进行防范和检查、前端则需要对特殊字符进行编码等；

csrf （跨域请求伪造）
简单的说，就是利用你在信任网站的登录信息，去伪造一个身份，然后去信任网站发起带有恶意的请求；
原理大概是：假设我登录了 a 页面，在没有登出的前提下（浏览器的 cookie 是跨页面的），访问了 b 页面（恶意页面），b 页面收集到 cookie 等敏感信息后，会利用这些信息去访问 a 页面，进行一些恶意请求； a 页面虽然不知道请求的主题是谁，但是因为有 cookie，a 页面还是会授于相应的权限，从而达到 b 页面的攻击目的；
防范：敏感操作多重验证，验证请求来源，cookie 非跨域设置等；

**5、Vue 中组件的 data 为什么是一个函数？而 new Vue 实例里，data 可以直接是一个对象？**

因为组件可能在应用中多次被使用而被多次实例化，组件的是个函数可以确保每个实例化后的实例有独立的作用域，从而确保组件实例化后的 data 中的数据不会相互污染。而 new Vue 生成的是根应用实例，只有一个。