## 1.axios

axios 使用 post 发送数据时，默认是直接把 json 放到请求体中提交到后端的。也就是说，我们的 Content-Type 变成了 application/json;charset=utf-8 ,这是axios默认的请求头content-type类型。但是实际我们后端要求的 'Content-Type': 'application/x-www-form-urlencoded' 为多见，这就与我们不符合。所以很多同学会在这里犯错误，导致请求数据获取不到。明明自己的请求地址和参数都对了却得不到数据。

我们现在来说说post请求常见的数据格式（content-type）

1. Content-Type: application/json ： 请求体中的数据会以json字符串的形式发送到后端
2. Content-Type: application/x-www-form-urlencoded：请求体中的数据会以普通表单形式（键值对）发送到后端
3. Content-Type: multipart/form-data： 它会将请求体的数据处理为一条消息，以标签为单元，用分隔符分开。既可以上传键值对，也可以上传文件。

## 2.mvc和mvvm

mvc

m:model 数据层

v:view 视图层

c:controller 控制层，响应用户的操作，并将model更新到view上。

所带来的问题

1. 大量的dom操作会频繁使dom树重构，影响客户端效率。
2. model频繁发生变化，开发者需要主动更新到view层上，当用户的操作导致model层发生变化，开发者同样需要将变化的数据更新到model层中，这样会导致工作繁琐，而且很难维护复杂多变的数据状态。

Mvvm

m:数据层

v:视图层

vm:viewModel是一个同步view和model的对象

v层和m层并不能直接交互，而是通过viewModel进行交互，通过数据的双向绑定将view和model连接了起来，并且同步的工作是自动进行的，因此开发者只用关心业务逻辑，不用关注dom，不需要关注数据状态的问题，这些由viewmodel统一管理。

Mvvm的前端框架vue

Vue2通过Object.defineProperty实现数据的双向绑定

Vue3通过Proxy里的get与set函数 实现数据的双向绑定