第三次补充内容

config的两种写法

1. fcl.config().put
2. import {config} from "@onflow/fcl"

transaction(mutate) 和 script(query)的参数问题

```
fcl.query({

    cadence: ``,

    args: (arg,t) => [arg(param,T)]

})
```

作业讲解

一 在测试网上注册一个测试账号并配置到flow.json中，将BookShelf合约部署到测试网环境下，并尝试script和transaction的操作（录制视频）

二 完成下列操作：

1. 添加fcl库
2. 初始化fcl配置
3. 添加fcl的授权登陆
4. 在配置中添加BookShelf的合约
5. 添加查询book数量的脚本
6. 执行该脚本

三 尝试分析一下fcl的实现和其他插件钱包（如MetaMask）的优势和劣势（问答题）

```
flow keys generate
```

之后跟上节课一样部署合约（flow deploy --network testnet），发送transaction(flow transactions send xx param --network testnet --signer xxx)和调用脚本(flow scripts execute xx param --network testnet)

```
npm install @onflow/fcl --save
```

config

- fcl.config().put("accessNode.api", "https://rest-testnet.onflow.org")
- Import {config} from "@onflow/fcl"

config({

"accessNode.api", "https://rest-testnet.onflow.org"

})

在index.js中import "./config"

```
fcl.authenticate();
fcl.unauthenticate();
```

- fcl.query({

  cadence: ``,

  args: (arg,t) => [arg(param,T)]

})

2. fcl对用户更友好，不需要浏览器拥有插件同样能够使用，而MetaMask必须有插件才能够进行一些交互，这让用户的使用有了门槛。 fcl对开发者也友好，不需要考虑钱包内部逻辑，只需要引入钱包这个功能，然后使用fcl的逻辑实现transaction，script等操作就ok。劣势的话，可能就是MetaMask更细粒度一些，开发者能够有更多跟钱包交互的功能