



学会 Substrate 区块链应用开发

4.3 - 使用 Front-end template


Jimmy Chu




jimmy.chu@parity.io







内容

- 介绍及准备
- 模块化设计
- 设计模式: 用 React Context 提供 Substrate API
- 设计模式: 提交交易


介绍



 ALICE   1.152M Unit

bob_stash	5HpG9w8EBLe5XCrbczpqw5TSXvedjrBGCwqxK1IQ7qUsSWFc		1.152M Unit
charlie	5FLSigC9HGRKvHb9FIEo4Y3koPsNmBmLJbpXg2mp1hXcS59Y		0
dave	5DAAnrj7VHTznn2AWBemMuyBwZW66FNfjdyVXUeYum3PTXFy		0
eve	5HGjWAeFDfFCWPsjFQdVv2Msvz2XtMktvgocEZcCj68kUMaw		0
ferdie	5CIPpSeXPECbkjWCa6MnjNokrgYjMqmkndv2rSnekmSK2DJL		0
Personal (polkadot-js)	5EU7GGmwiVMdsc9zAHduRe18C1LNiC6rUnK6LoPxnWQk1mYu		0

Transfer

 1 Unit = 1000000000000

To

Amount

Upgrade Runtime

Wasm File No file chosen

Pallet Interactor

Interaction Type ☒ Extrinsic ☐ Query ☐ RPC ☐ Constant

Unsigned


 or

Signed

 or

SUDO

Events



- 如你是以 React 作前端开发, Front-end Template 对你有帮助意义
- 演示

准备

```
git clone  
https://github.com/substrate-developer-hub/substrate-front-end-template  
cd substrate-front-end-template  
yarn install  
yarn start
```

模块化设计

- 从 App.js 可看出每个模块对应的代码
- 容易找到如何在 React 里实现该模块的功能

```
function Main () {
  //...
  return (
    <div ref={contextRef}>
      <Sticky context={contextRef}>
        <AccountSelector
          setAccountAddress={setAccountAddress} />
      </Sticky>
      { //... }
    </Container>
    <Grid stackable columns='equal'>
      <Grid.Row stretched>
        <Transfer accountPair={accountPair} />
        <Upgrade accountPair={accountPair} />
      </Grid.Row>
    </Grid>
  </Container>
</div>
);
}

export default function App () {
  return (
    <SubstrateContextProvider>
      <Main />
    </SubstrateContextProvider>
  );
}
```

设计模式：用 React Context 提供 Substrate API

参看：

- [React Context Hook](#)
- `src/substrate-lib/SubstrateContext.js`
- `src/substrate-lib/useSubstrate.js`
- `src/App.js`

设计模式：用 React Context 提供 Substrate API

SubstrateContext.js

- 定义 context 提供的值: 包括 socket, keyring, api, 及他们的状态
- 状态转换函数 `reducer = (...) => { ... }`

useSubstrate.js

初始设置

- 连到去 substrate 節點
`const provider = new WsProvider(socket);`
- 读取钱包帐户
`let allAccounts = await web3Accounts();`

设计模式：用 React Context 提供 Substrate API

App.js

- 把你的应用包在
`<SubstrateContextProvider>` 之内
- 这样所有组件都可轻松调用 API

```
const { api, keyring } =  
useSubstrate();
```


设计模式：提交交易

参看

- `src/Interactor.js`
- `src/substrate-lib/components/TxButton.js`

设计模式：提交交易

- Interactor 提供介面设定要呼唤的模块, 函数, 及要输入的参数
- <TxButton> 可以处理不同类型与 pallet 互动的类型:
 - QUERY - 存储查询
 - RPC - 远程过程调用
 - SIGNED-TX - 具签名交易
 - UNSIGNED-TX - 不具签名交易
 - SUDO-TX - 超管交易
 - CONSTANT - 常量查询

```
TxButton.propTypes = {
  accountPair: PropTypes.object,
  setStatus: PropTypes.func.isRequired,
  type: PropTypes.oneOf([
    'QUERY', 'RPC', 'SIGNED-TX',
    'UNSIGNED-TX', 'SUDO-TX',
    'CONSTANT' ]).isRequired,
  attrs: PropTypes.shape({
    palletRpc: PropTypes.string,
    callable: PropTypes.string,
    inputParams: PropTypes.array,
    paramFields: PropTypes.array
  }).isRequired
};
```

<https://github.com/paritytech/substrate>

- star
- watch
- fork

官网文档: substrate.dev

知乎专栏: parity.link/zhihu

jimmy.chu@parity.io