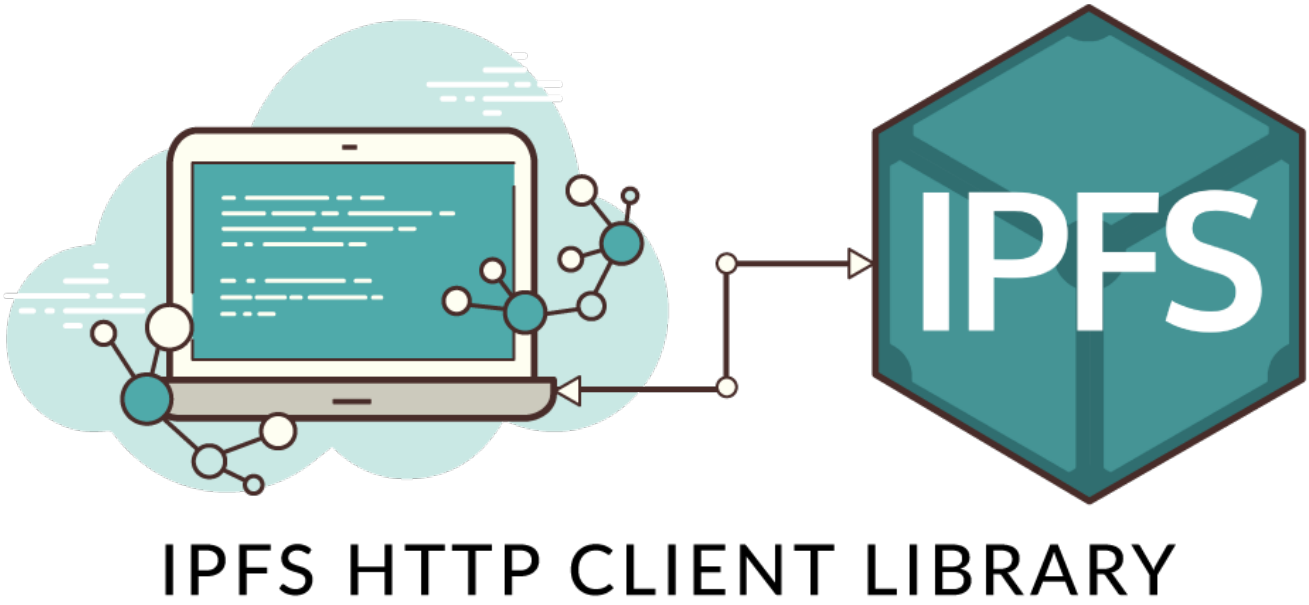
0004 -【IPFS + 区块链 系列】 入门篇 - IPFS + Ethereum （中篇）-js-ipfs-api - IPFS图片上传与下载

Ebay项目



目录

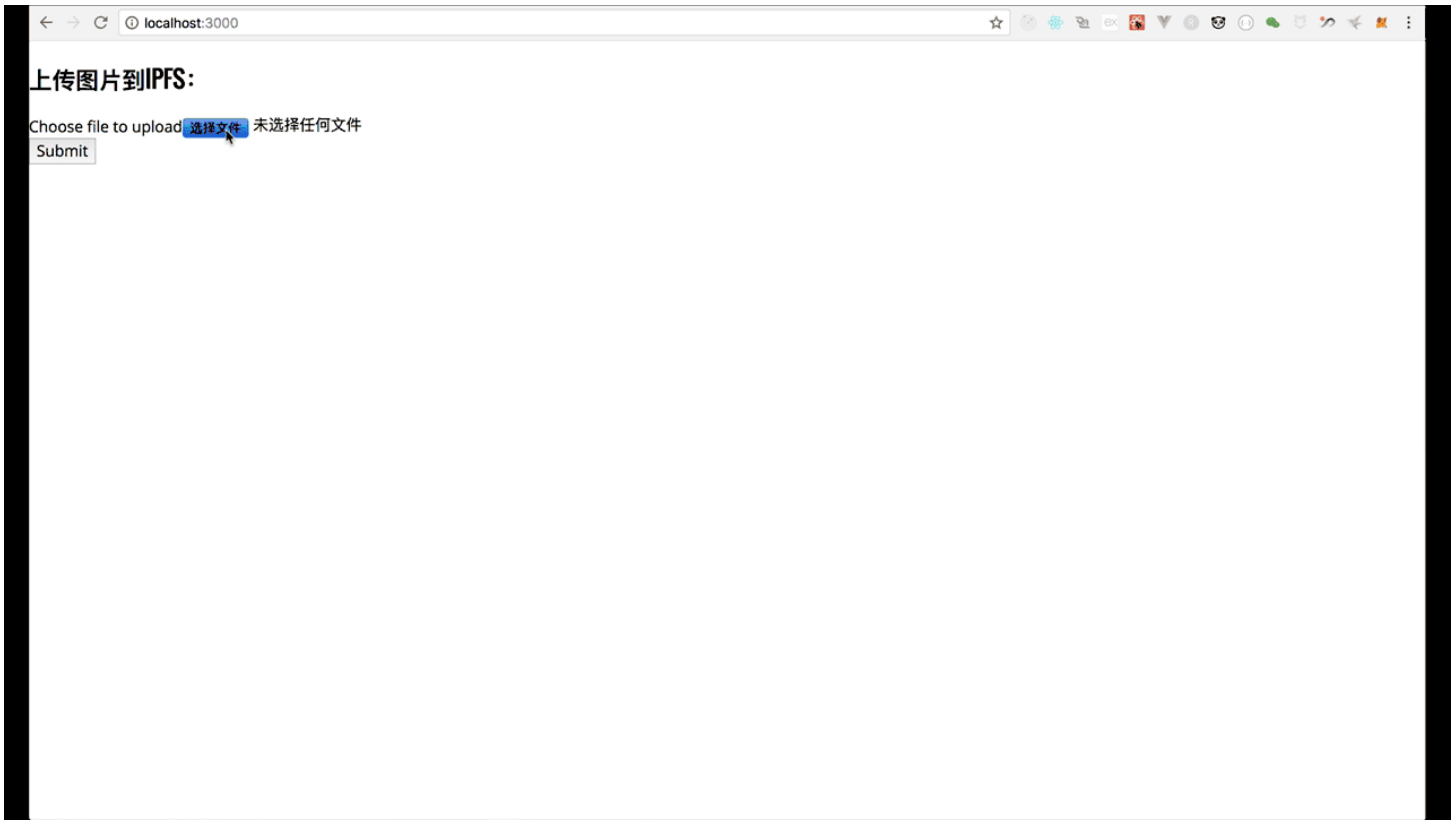
系列文章

- 【IPFS + 区块链 系列】 入门篇 - IPFS环境配置
- 【IPFS + 区块链 系列】 入门篇 - IPFS+IPNS+个人博客搭建
- 【IPFS + 区块链 系列】 入门篇 - IPFS + Ethereum （上篇）-js-ipfs-api - 数据上传到IPFS

1. 项目效果图

选择图片，点击`Submit`将图片提交到`IPFS`，并且返回`IPFS`的`HASH`值，再通

过 HASH 从 IPFS 读取图片。



2. 创建React项目

具体不知道怎么操作的，请移步到【IPFS + 区块链 系列】 入门篇 - IPFS + Ethereum （上篇）-js-ipfs-api。

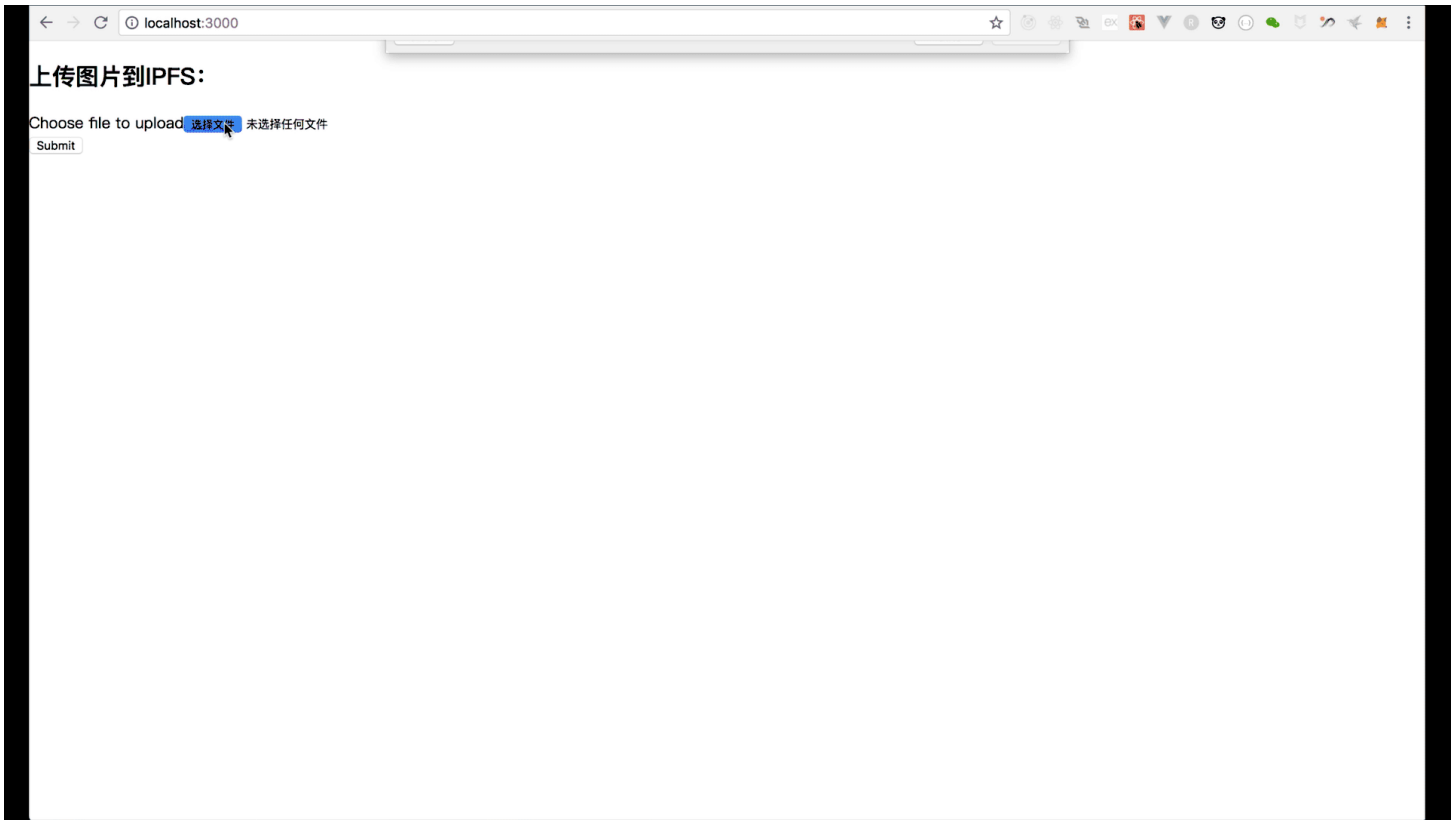
```
$ create-react-app ipfs_img
```

3. 完成 UI 逻辑

将下面的代码拷贝替换掉 App.js 里面的代码。

```
import React, {Component} from 'react'

class App extends Component {
  constructor(props) {
    super(props)

    this.state = {
      imgSrc: null
    }
```

```
  }


  render() {
    return (<div className="App">

      <h2>上传图片到IPFS: </h2>
      <div>
        <label id="file">Choose file to upload</label>
        <input type="file" ref="file" id="file" name="file" multiple="multip
le"/>
      </div>
      <div>
        <button onClick={() => {
            var file = this.refs.file.files[0];
            var reader = new FileReader();
            // reader.readAsDataURL(file);
            reader.readAsArrayBuffer(file)
            reader.onloadend = (e) => {
              console.log(reader);
              // 上传数据到IPFS
            }

          }}>Submit</button>
      </div>
      {
        this.state.imgSrc
          ? <div>
              <h2>{"http://localhost:8080/ipfs/" + this.state.imgSrc}</h2>
              <img alt="区块链部落" style={{
                  width: 1600
                }} src={"http://localhost:8080/ipfs/" + this.state.imgSrc}/>
            </div>
          : <img alt=""/>
      }
    </div>);
  }
}

export default App
```



4. 安装 `ipfs-api`

```
localhost:1126 yuechunli$ cd ipfs_img/
localhost:ipfs_img yuechunli$ ls
README.md   package.json    src
node_modules    public      yarn.lock
localhost:ipfs_img yuechunli$ npm install --save-dev ipfs-api
```

5. `App.js` 导入IPFS

```
const ipfsAPI = require('ipfs-api');
const ipfs = ipfsAPI({host: 'localhost', port: '5001', protocol: 'http'});
```

6. 实现上传图片到IPFS的Promise函数

```
let saveImageOnIpfs = (reader) => {
  return new Promise(function(resolve, reject) {
    const buffer = Buffer.from(reader.result);
    ipfs.add(buffer).then((response) => {
      console.log(response)
```

```
      resolve(response[0].hash);
    }).catch((err) => {
      console.error(err)
      reject(err);
    })
  })
}
```

7. 上传图片到IPFS

```
var file = this.refs.file.files[0];
var reader = new FileReader();
// reader.readAsDataURL(file);
reader.readAsArrayBuffer(file)
reader.onloadend = function(e) {
  console.log(reader);
  saveImageOnIpfs(reader).then((hash) => {
    console.log(hash);
    this.setState({imgSrc: hash})
  });
```

- `reader.readAsDataURL(file);` 上传图片路径。
- `reader.readAsArrayBuffer(file)` 上传图片内容
- 上传图片

```
saveImageOnIpfs(reader).then((hash) => {
    console.log(hash);
    this.setState({imgSrc: hash})
  });
```

`hash` 即是上传到 `IPFS` 的图片的 `HASH` 地址， `this.setState({imgSrc: hash})` 将 `hash` 保存到状态机变量 `imgSrc` 中。

8. 完整代码

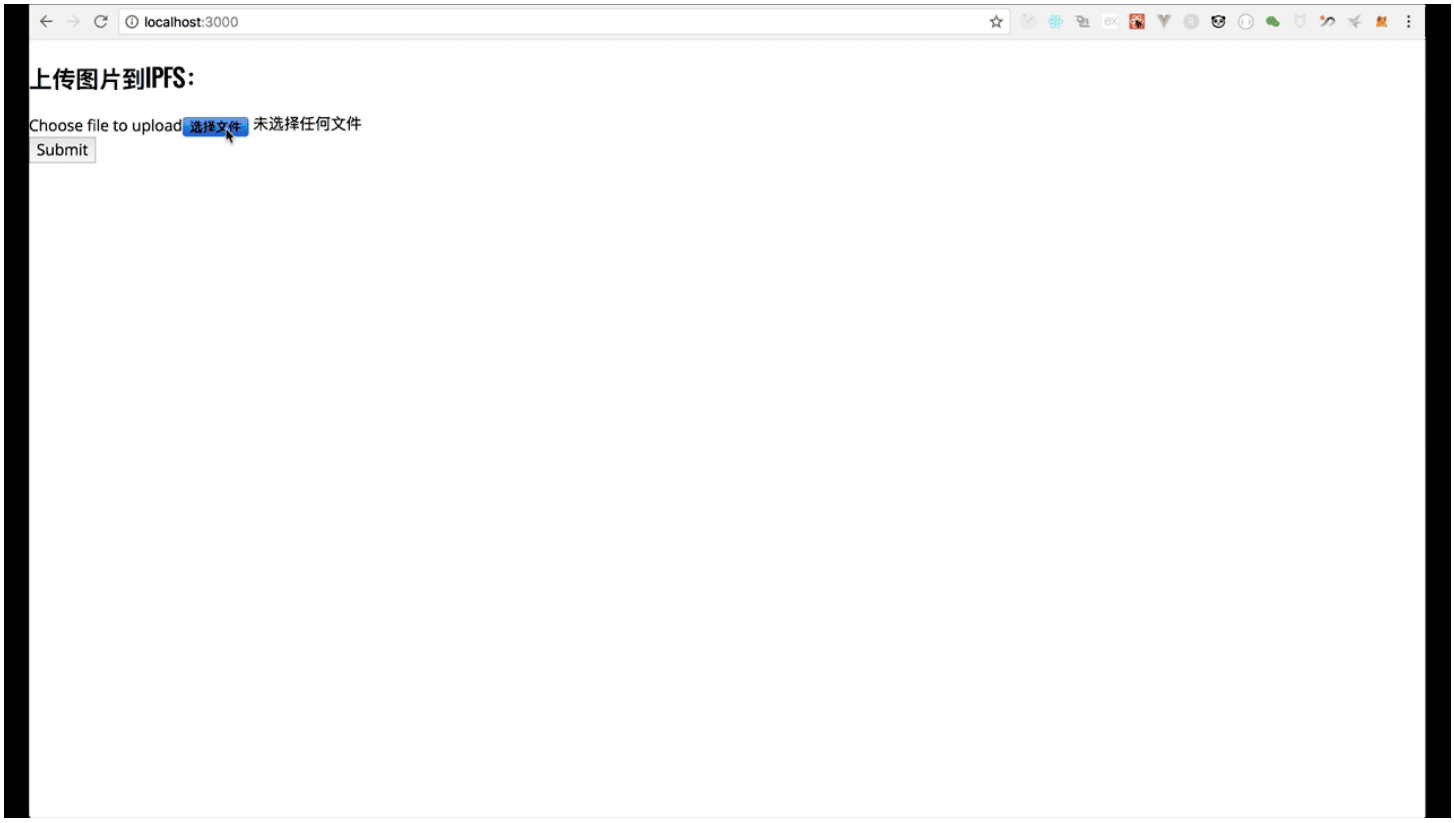
```
import React, {Component} from 'react'

const ipfsAPI = require('ipfs-api');
const ipfs = ipfsAPI({host: 'localhost', port: '5001', protocol: 'http'});
```

```
let saveImageOnIpfs = (reader) => {
  return new Promise(function(resolve, reject) {
    const buffer = Buffer.from(reader.result);
    ipfs.add(buffer).then((response) => {
      console.log(response)
      resolve(response[0].hash);
    }).catch((err) => {
      console.error(err)
      reject(err);
    })
  })
}

class App extends Component {
  constructor(props) {
    super(props)

    this.state = {
      imgSrc: null
    }
  }

  render() {
    return (<div className="App">

      <h2>上传图片到IPFS：</h2>
      <div>
        <label id="file">Choose file to upload</label>
        <input type="file" ref="file" id="file" name="file" multiple="multip
le"/>
      </div>
      <div>
        <button onClick={() => {
            var file = this.refs.file.files[0];
            var reader = new FileReader();
            // reader.readAsDataURL(file);
            reader.readAsArrayBuffer(file)
            reader.onloadend = (e) => {
              console.log(reader);
              // 上传数据到IPFS
              saveImageOnIpfs(reader).then((hash) => {
                console.log(hash);
                this.setState({imgSrc: hash})
              });
            }

          }}>Submit</button>
      </div>
```

```
      {
        this.state.imgSrc
          ? <div>
              <h2>{"http://localhost:8080/ipfs/" + this.state.imgSrc}</h2>
              <img alt="区块链部落" style={{
                  width: 1600
                }} src={"http://localhost:8080/ipfs/" + this.state.imgSrc}/>
            </div>
          : <img alt=""/>
      }
    </div>);
  }
}

export default App
```

9. 运行项目

- `npm start`
- 新建终端，启动节点服务`ipfs daemon`



10. 总结

这本文章主要复习如何创建React项目，如何安装`ipfs-api`，如何编写上传图片

到 `IPFS` 的 `Promise` 函数，下一篇我们将介绍，如何将图片 `hash` 存储到区块链，如何从区块链取到 `hash`，并且通过 `hash` 从 `ipfs` 拿到图片。

11. 技术交流

- 区块链技术交流QQ群： `348924182`
- 进微信群请加微信： `liyc1215`
- 「区块链部落」官方公众号