Virtual File System & Management

这是操作系统的第三次课程设计项目，我开发的是一个基于 Vue 的文件管理系统和基于 NodeJS 的虚拟文件系统。

TIPS：由于编码问题，现在只支持非中文的文件和文件夹名，但是文件内容已使用 UTF-8，支持所有语种

使用方式

0. 首先确保电脑上安装了 npm >= v7.5.0
1. 启动 server 端：

```
$ npm run server
```

2. 启动静态文件 server 端，系统默认浏览器会被自动打开：

```
$ npm start
```

代码结构

程序由四个部分组成：

- 前端界面层：
  - 基于 Vue、ElementUI 和 es6
  - 手动管理数据状态
- 数据链接层：
  - 使用 HTTP 二进制字节流传输文件
  - 实现了简单的 MIME Parser，具备文件转码能力
  - **支持所有格式的文件**
- 虚拟文件系统层：
  - 在内存中开辟空间来模拟硬盘
  - 采用 FAT 来实现文件信息的储存
  - 采用红黑树来实现文件数据的储存
  - 空闲空间采用多级链表，保证先整体划分空间，再分开存储，充分利用系统缓存
  - 采用实时持久化系统，防止程序突然中止所造成的数据丢失，同时也保证了内存和硬盘上的数据一致性
- 数据持久化层：
  - 将文件从内存中的 Buffer 状态转码成相应的 type
  - 调用系统的接口将文件 **递归** 地写入硬盘，保证内存和硬盘上的拓扑一致性
  - 程序一启动就会将持久化的数据 **递归** 地读入内存

核心代码

虚拟文件系统基类：

```
const dirEntry = _.struct([
    _.struct('Name', [
        _.char('filename',8),
        _.char('extension',3)
    ]),
    _.struct('Attr', [
        _.bool('readonly'),
        _.bool('hidden'),
        _.bool('system'),
        _.bool('volume_id'),
        _.bool('directory'),
        _.bool('archive'),
        _.ubit('reserved', 2)
    ].reverse()),
    _.byte('NTRes', 1),
    _.uint8('CrtTimeTenth'),
    _.struct('CrtTime', [time]),
    _.struct('CrtDate', [date]),
    _.struct('LstAccDate', [date]),
    _.uint16le('FstClusHI'),
    _.struct('WrtTime', [time]),
    _.struct('WrtDate', [date]),
    _.uint16le('FstClusLO'),
    _.uint32le('FileSize')
]);
```

FAT 类与 Mark Code：

```
const fatField = {
    'fat12': _.struct('Status', [
        _.ubit('field0bc', 8),
        _.ubit('field1c', 4),
        _.ubit('field0a', 4),
        _.ubit('field1ab', 8),
    ]),
    'fat16': _.uint16le('Status'),
    'fat32': _.uint32le('Status')
};

const fatPrefix = {
    'fat12': 0xF00,
    'fat16': 0xFF00,
    'fat32': 0x0FFFFF00
};

const fatStat = {
    free: 0x00,
    _undef: 0x01,
    rsvMin: 0xF0,
    bad: 0xF7,
    eofMin: 0xF8,
    eof: 0xFF
};
```

红黑树类：

```
// 节点基类
function Node(key, value) {
        this.key = key;
        this.value = value;
        this.color = RED;
        this.left = null;
        this.right = null;
        this.parent = null;
    };

// 指针类
function Cursor(tree, start, end) {
        this.tree = tree;
        this.start = start;
        this.end = end;

        var self = this;
        this.walk = function walk(node, iterator) {
            if (node === null) return;

            if (start !== undefined && node.key < start) {
                walk(node.right, iterator);
            } else if (end !== undefined && node.key > end) {
                walk(node.left, iterator);
            } else {
                walk(node.left, iterator);
                iterator(node.value, node.key, self.tree);
                walk(node.right, iterator);
            }
        };
    };

// 树类
function Tree() {
        this.root = null;
        this.balancer = new Balancer(this);
    };

// 平衡组合类
function Balancer(tree) {
        this.tree = tree;
    };
```

HTTP 文件 Parser类：

```
// 文件流转化类
function PartStream (input, headers) {
  this.headers = headers;
  this.readable = true;
  this.input = input;
  this.events = [];
}

// 文件解析类
function Parser (input, boundary) {
  this.boundary = boundary;
  this.input = input;
  this.state = BOUNDARY_START;
  this.offset = 0;
  this.index = 0;
  this.chunks = [];
  input.on('data', this.onData.bind(this));
  input.on('end', this.onEnd.bind(this));
}
```

项目亮点

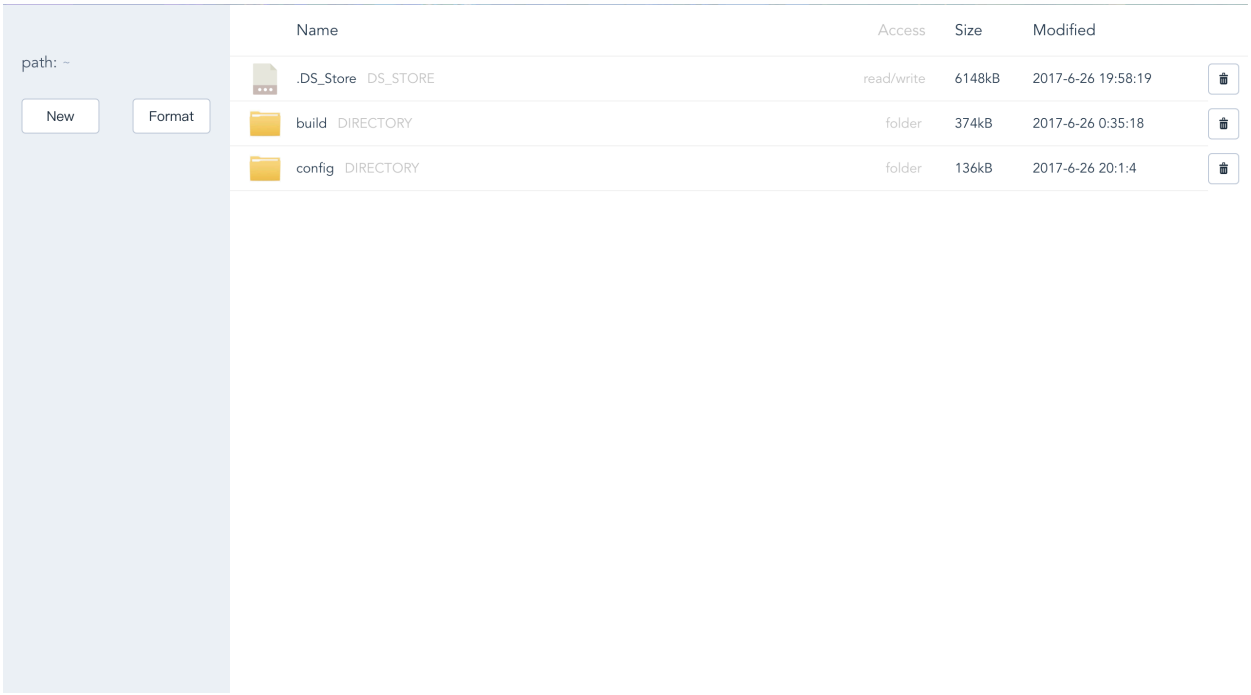
1. 结构清晰，层次划分明确
2. 页面简洁，实现了简单的过渡动画，交互较为友好
3. 支持所有类型的文件
4. 实现了硬盘和内存中数据和拓扑的一致性
5. 支持所有要求的文件操作
6. 实现了真实的文件储存

项目不足

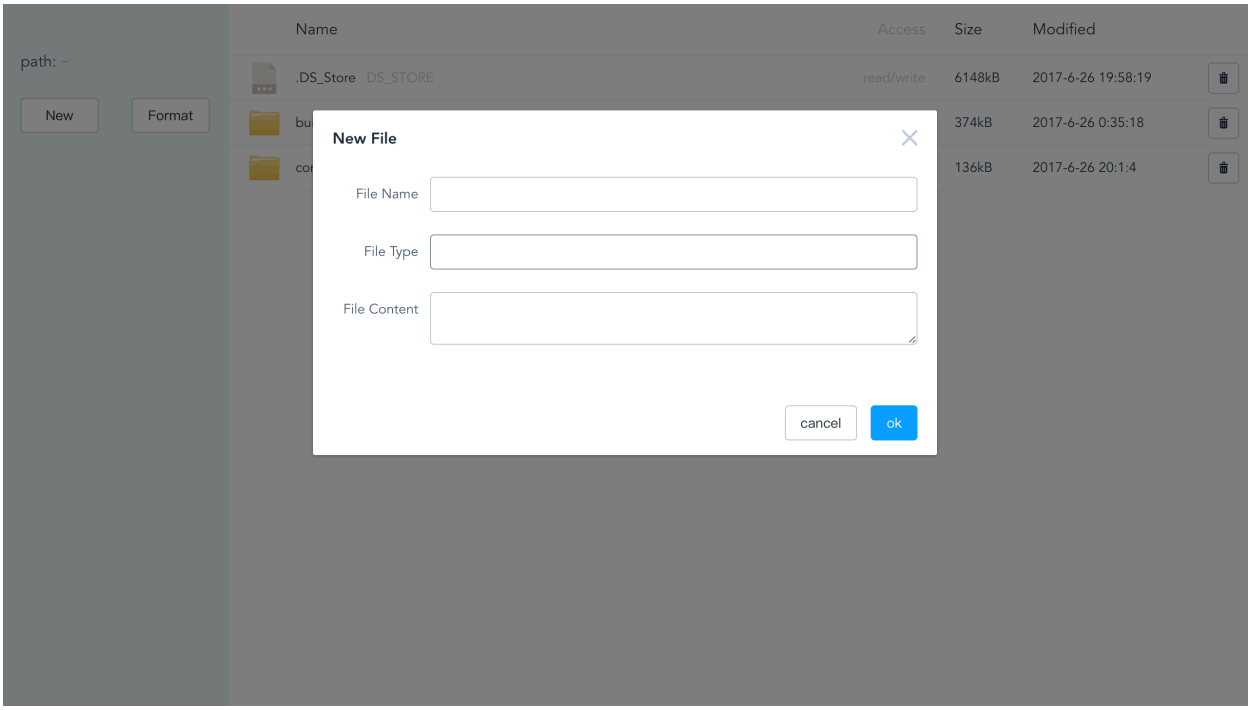
1. 没有实现流媒体（IMG、MP3、MP4等）的z展示交互功能
2. 中文仍有编码问题
3. 暂不支持 copy 交互操作（但是后端接口已完成）
4. 为了实现真实读取、存储文件，不得不递归地调用 IO 函数，性能上有一定损耗

项目截图

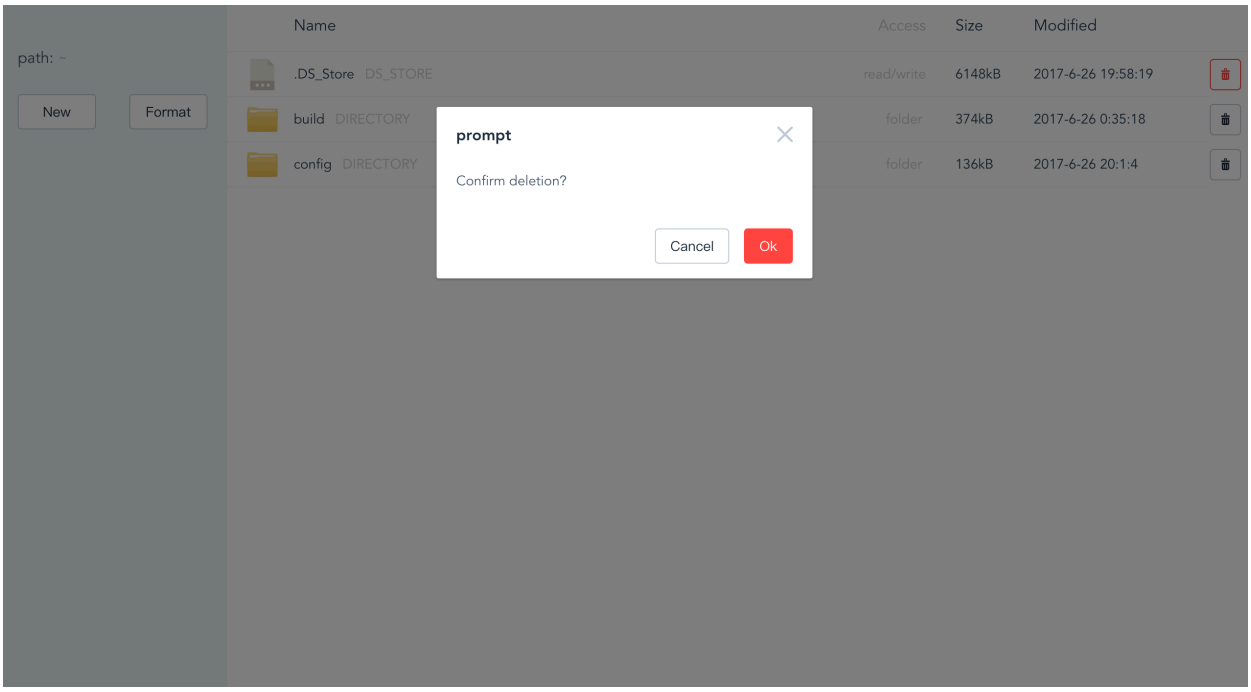
主页


path: ~
New
Format
Name
Access
Size
Modified
.DS_Store DS_STORE
read/write
6148kB
2017-6-26 19:58:19
build DIRECTORY
folder
374kB
2017-6-26 0:35:18
config DIRECTORY
folder
136kB
2017-6-26 20:1:4

新建文件


path: ~
New
Format
Name
Access
Size
Modified
.DS_Store DS_STORE
read/write
6148kB
2017-6-26 19:58:19
374kB
2017-6-26 0:35:18
136kB
2017-6-26 20:1:4
New File
File Name
File Type
File Content
cancel
ok

删除文件夹


path: ~
New
Format
Name
Access
Size
Modified
.DS_Store DS_STORE
read/write
6148kB
2017-6-26 19:58:19
build DIRECTORY
folder
374kB
2017-6-26 0:35:18
config DIRECTORY
folder
136kB
2017-6-26 20:1:4
prompt
Confirm deletion?
Cancel
Ok

LICENSE

MIT