高级node第二单元（5.29）

2020年5月29日 8:17

目标：编写属于自己的命令行程序

看懂node官方文档

学习node的核心模块

Process

- beforeExit事件
- exit事件
- Exit()
- Abort()
- Arch
- Argv
- Argv0
- Cwd()
- Chdir
- Env
- exitCode
- Stdin
- stdout

process模块

1. 不需要引用可以直接使用
2. 进程，一个程序从运行开始到结束的过程中所有相关的信息会被记录在进程对象（process）中,当程序全部执行完毕后，进程就会退出

```
index.js
1.process学习 > index.js > ...
1  console.log('程序开始-测试内部的process就记录了程序运行过程中的所有信息')
2
3  // 当前运行的工作目录是
4  console.log(process.cwd());
5
6  console.log('进程还没有结束，等待一个延时器')
7
8  setTimeout(() => {
9    console.log('进程真正结束了')
10 }, 5000)
```

3. Exit() 强制退出进程
4. abort（）强制退出进程，但是生成core文件
5. Arch 返回一个当前cpu的架构字符串
6. Argv 用于接受命令行传递的参数
7. Argv0 返回当前环境的字符串
8. Cwd() 返回的是当前的工作目录
9. chdir（）用于去改变当前的工作目录
10. Env 用来获取当前系统的环境变量，也可以随意设置

Process.stdin和process.stdout

1. Process.stdin表示一个输入流，代表从命令行中输入的内容指向内存中的变量
2. Process.stdout表示一个输出流，代表的是从内存中的变量指向命令行的显示

```
let count = 0;

// 把数据通过流输出到控制台
process.stdout.write('请输入你的用户名：')
// 数据从命令行中流过来
process.stdin.on('data', chunk => {
    count++;
    if (count >= 2) {
        // 退出程序
        process.exit();
    }
    // 拿到用户名之后要求输入密码
    process.stdout.write('请输入你的密码：')
})
```

退出事件和状态码

```
console.log('进程开始')

process.exitCode = 300;
process.on('exit', (exitCode) => {
    console.log(exitCode)
    console.log('exit');
})
process.on('beforeExit', (exitCode) => {
    console.log(exitCode)
    console.log('beforeExit')
})
```

退出码来自于exit()的参数或者设置的exitCode的属性值

程序退出的时候触发