# 实验10：Nodejs调试与测试

实验背景

利用已掌握的知识，完成实例。如图所示：

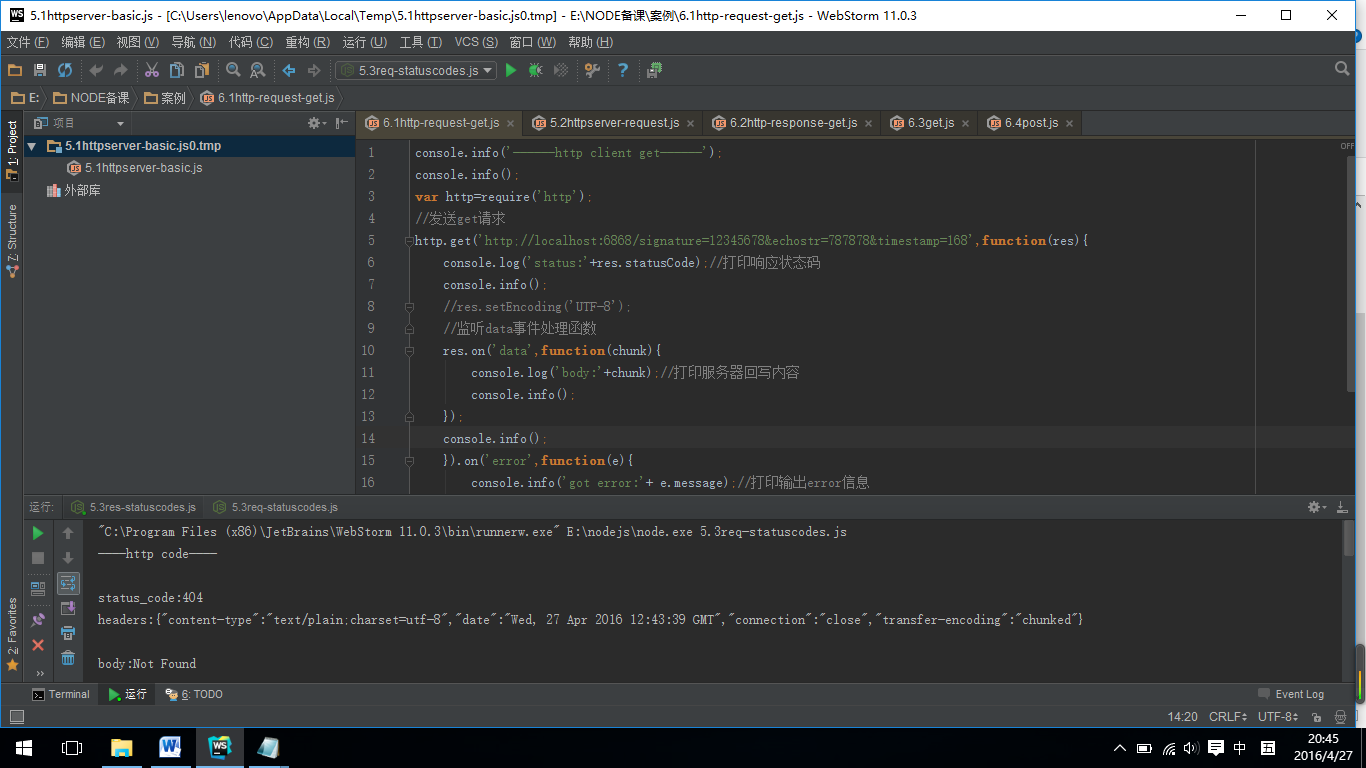


图1-1

实验目的

1．掌握nodejs调试

2．掌握nodejs测试

实验分析

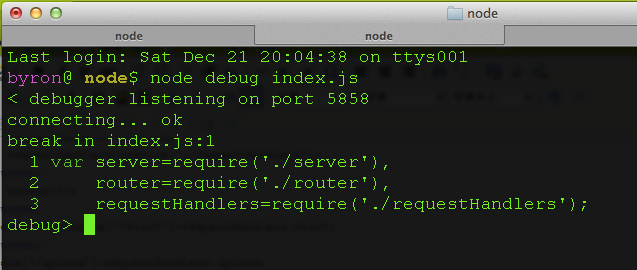
本次实验要能够熟练掌握nodejs调试与测试。

实验步骤

1、原生控制台调试

在启动服务的时候添加debug 选项

node debug index.js



这时候输入一些指令就可以单步调试、到断点监视局部变量等，看个命令图，很多命令都有其缩写形式

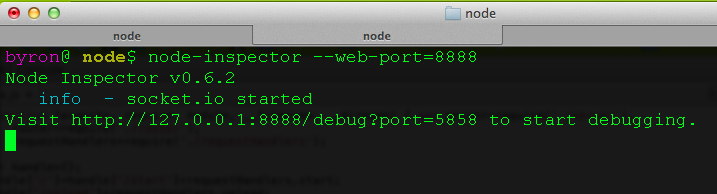
|  |  |
| --- | --- |
| node.js调试命令 | |
| 命令 | 功能 |
| run | 执行脚本,在第一行暂停 |
| restart | 重新执行脚本 |
| cont, c | 继续执行,直到遇到下一个断点 |
| next, n | 单步执行 |
| step, s | 单步执行并进入函数 |
| out, o | 从函数中步出 |
| setBreakpoint(), sb() | 当前行设置断点 |
| setBreakpoint(‘f()’), sb(...) | 在函数f的第一行设置断点 |
| setBreakpoint(‘script.js’, 20), sb(...) | 在 script.js 的第20行设置断点 |
| clearBreakpoint, cb(...) | 清除所有断点 |
| backtrace, bt | 显示当前的调用栈 |
| list(5) | 显示当前执行到的前后5行代码 |
| watch(expr) | 把表达式 expr 加入监视列表 |
| unwatch(expr) | 把表达式 expr 从监视列表移除 |
| watchers | 显示监视列表中所有的表达式和值 |
| repl | 在当前上下文打开即时求值环境 |
| kill | 终止当前执行的脚本 |
| scripts | 显示当前已加载的所有脚本 |
| version | 显示v8版本 |

2、使用node-inspector调试

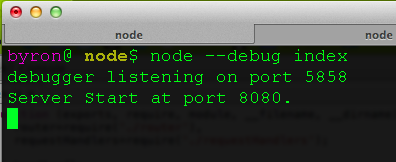
①首先使用npm把其安装在全局环境中

npm install -g node-inspector

node-inspector是通过websocket方式来转向debug输入输出的。因此，我们在调试前要先启动node-inspector来监听node.js的debug调试端口。默认情况下node-inspector的端口是8080，可以通过参数--web-port=[port]来设置端口。



在启动node-inpspector之后，我们可以通过--debug或--debug-brk来启动node.js程序。



3、单元测试

npm install -g mocha

mkdir test

$EDITOR test/test.js

var assert = require("assert");

describe('Array', function() {

describe('#indexOf()', function() {

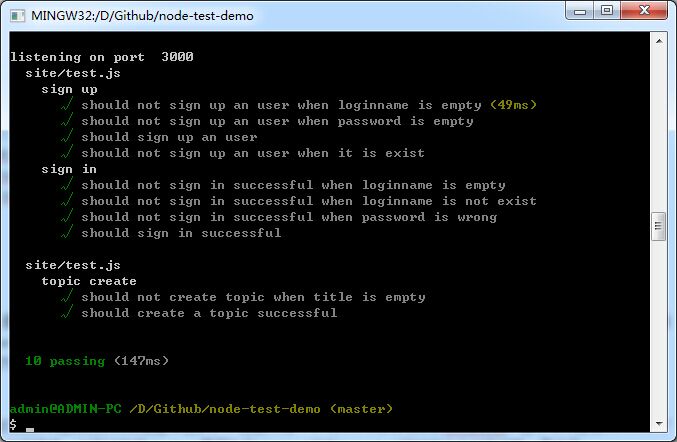
it('should return -1 when the value is not present', function() {

assert.equal(-1, [1,2,3].indexOf(5)); assert.equal(-1, [1,2,3].indexOf(0));

}); }); });

mocha

✔ 1 test complete (1ms)



扩展练习

1.思考并实现，搜索第三方测试工具。

实验思考

1.思考并实现，使用第三方测试工具。